UNIVERSITY OF MELBOURNE

Identification of translational regulatory elements in mRNAs

Author:
Jiangrong OUYANG

Supervisor:
Dr. Heejung SHIM

A thesis submitted in fulfilment of the requirements for the degree of Master of Science (Mathematics and Statistics)

in the
Faculty of Science
School of Mathematics and Statistics

May 2019

Declaration

Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree at any university.

Jiangrong OUYANG

Acknowledgements

My sincere thanks go to my supervisor Dr. Heejung Shim for her invaluable advice, feedback and mentorship.

I would like to express gratitude to our research group: Qian Feng for her patience on teaching me how to use computer clusters, Yupei You for pre-processing such complicated data, Brendan Law for proof-reading at the final stage of writing this thesis, Denny Shin for helping me with my presentation skills and Ziwei Liu for supporting and encouraging me when I was stressed.

I would like to thank my family in particular, my parents, for their love and financial support.

UNIVERSITY OF MELBOURNE

Abstract

Faculty of Science
School of Mathematics and Statistics

Master of Science

by Jiangrong OUYANG

Unlike the transcriptional regulatory mechanisms in the central dogma, much less is known about translational regulatory mechanisms. A particularly intriguing one involves upstream Open Reading Frames (uORFs). This project develops the statistical methods that enable comprehensive identification of the locations of the translated uORF and downstream main coding region within the messenger RNA (mRNA). It will facilitate our understanding of translational regulatory mechanisms.

We begin by studying the structure and properties of ribosome profiling data. Then we construct a hidden Markov model for the ribosome profiling data around translated uORFs and downstream main coding regions within mRNAs. We implement an R package for this model, called *riboHMM2*. The package includes estimation of model parameters via the Expectation-Maximisation (EM) algorithm and identification of locations of translated uORFs and downstream main coding regions in mRNAs via the Viterbi algorithm. We conduct a simulation study on three cases to show the model's ability to identify the translated uORFs and downstream main coding regions within mRNAs.

Contents

Acknowledgements **i**

Abstract **ii**

Contents **iii**

1 Introduction **1**

1.1 Background 1

1.1.1 Translation mechanism 2

1.1.2 Ribosome profiling 3

1.1.3 Upstream Open Reading Frame 4

1.2 riboHMM 5

1.3 Objective 6

2 Method **7**

2.1 How we construct the model 7

2.1.1 Characteristic structure of ribosome footprints 7

2.1.2 Construction of hidden Markov model 9

2.2 Model set-up 12

2.3 Some features of emission distribution 14

2.4 Likelihood 15

2.5 Expectation-Maximization algorithm 17

2.5.1 Expectation step 17

2.5.2 Maximization step 20

2.6 Viterbi Algorithm 23

3 R package **25**

3.1 Implementation 25

3.1.1 Estimation 26

3.1.2 Inference 27
3.2 Data sets 27
3.2.1 Counts generated from model 28
3.2.2 Counts generated based on what real ribosome footprints look like 29
3.3 Vignette 30

4 Simulation studies **35**
4.1 Case 1: use counts generated from our model 35
4.2 Case 2: use counts generated based on what real ribosome footprints look like 36
4.3 Case 3: use counts with noise in mRNA sequences 37

5 Future study **39**
5.1 Real data analysis 39
5.2 Model modification 40

A Algorithms and functions used in R package **41**

B R code for simulation studies **45**

Bibliography **51**

Chapter 1

Introduction

1.1 Background

Proteins are complex molecules that play essential roles in the body. They are made up of hundreds or thousands of smaller units called amino acids. Even though there are only 20 types of amino acids, the order of combining amino acids and the different way of amino acids folding to three-dimensional structures enable protein to have various functions.

The synthesis of proteins occurs in the process of translation, which is explained in the central dogma. The central dogma is a two-step theory which concerns the flow of gene information [1]. The information in genes is first passed from DNA to messenger RNA (mRNA) via transcription. Then the mRNA is decoded by ribosome in the process of translation, and the information in genes is expressed by proteins synthesized. The graphical flow of gene information is shown in Figure 1.1.

However, unlike the transcriptional regulatory mechanisms in the central dogma, much less is known about translational regulatory mechanisms.

The rate of translation is an essential factor of protein expression level. An intriguing translational regulatory mechanism involves the upstream Open Reading Frames (uORFs). The translation at uORFs has been known to affect translation of coding regions during important biological processes [2] [3] [4], such as hypoxia condition [5] and stress response [6].

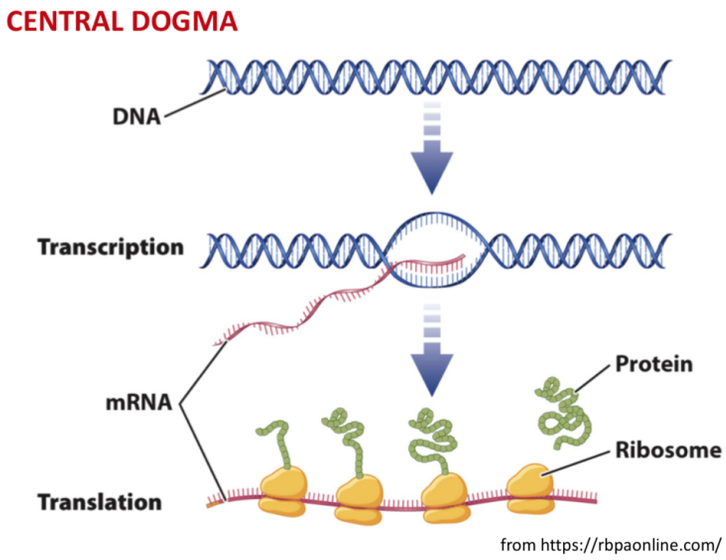


Figure 1.1: Information in genes flows into proteins via transcription and translation

Hence, we want to develop the statistical methods that enable comprehensive identification of the location of the translated uORFs and downstream main coding regions within mRNAs, which will facilitate our understanding of translational regulatory mechanisms.

In the following sections, we provide some biological background needed for this thesis. The process of translation in the central dogma is introduced, followed by a new technique, ribosome profiling, that is used to study locations where translation occurs. Then we introduce the upstream open reading frame, a region in mRNA, whose location will be the main focus of this project.

1.1.1 Translation mechanism

Our project focuses on the process of translation. In a cell, the messenger RNA (mRNA) carries the information for making protein, wandering in cytoplasm. As shown in Figure 1.2, at first, a small subunit of ribosome attaches on one end of mRNA, scanning the sequence of mRNA. When it finds a start codon, e.g., AUG, a large subunit of ribosome comes and combines with the small subunit, and the two units are assembled into a ribosome. Then the ribosome moves along the mRNA, decoding 3 bases (a codon) at a time. For every codon decoded, one corresponding amino acid is brought by transport RNA (tRNA), and combines together to form an amino acid chain (also known as polypeptide). When the ribosome meets a stop

codon, e.g., UAG, the amino acid chain is released, and the ribosome comes apart and is released as well. The released amino acid chain later folds into a protein. The process of synthesizing a protein is known as translation.

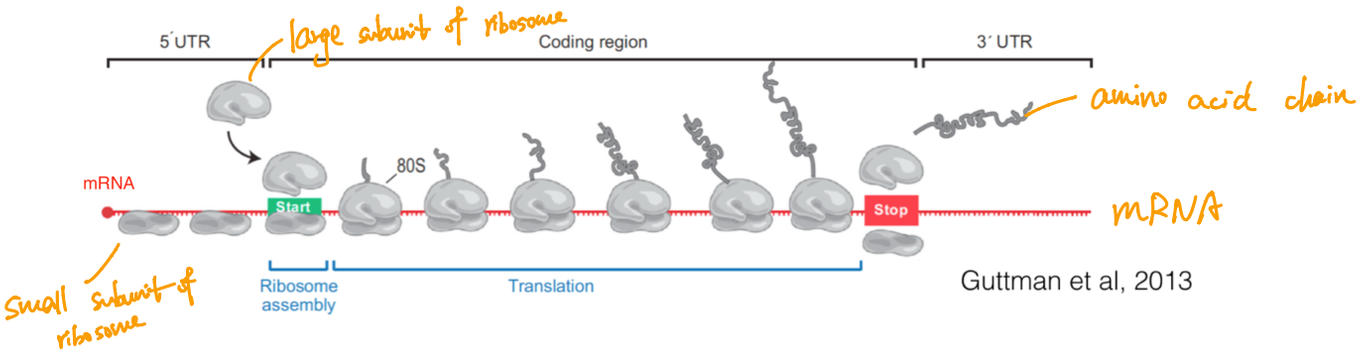


Figure 1.2: Translation mechanism. Adapted from [7].

The region where mRNA is translated to form an amino acid chain is called the coding region. The region before translation is called 5' untranslated region (5' UTR), while the region after translation is called 3' untranslated region (3' UTR).

As the functionality of protein is determined by the sequence of the coding region in mRNA, identifying the locations of translated coding regions is important. GENCODE provides the annotations for protein coding regions [8], which relies on manual annotation from the Human and Vertebrate Analysis and Annotation (HAVANA) group [9]. However, this collection of coding regions only includes coding genes that have only one long and conserved coding regions. Hence, a method for comprehensive identification of translated coding region is needed.

limitation of GENCODE

1.1.2 Ribosome profiling

An mRNA sequence contains four letters: A, C, U, and G, which represent four bases: adenine, cytosine, uracil and guanine. Based solely on an mRNA sequence, we cannot identify where the translated coding region is. However, since the the translation requires ribosome, we could use ribosome occupancy to study the coding information hidden in mRNA.

Ribosome profiling, or Ribo-Seq, is a new technique based on deep sequencing of ribosome-protected mRNA fragments. Purification and sequencing of these fragments provide information about which part of mRNAs were translated [10].

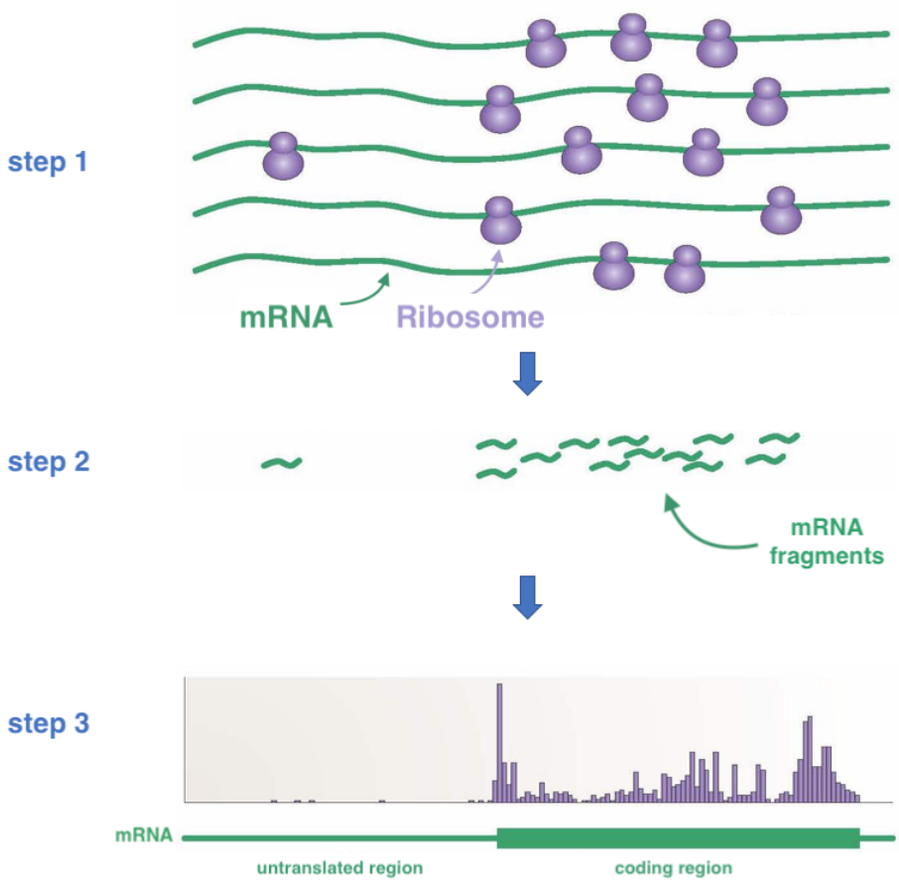


Figure 1.3: The process of ribosome profiling. Adapted from [11].

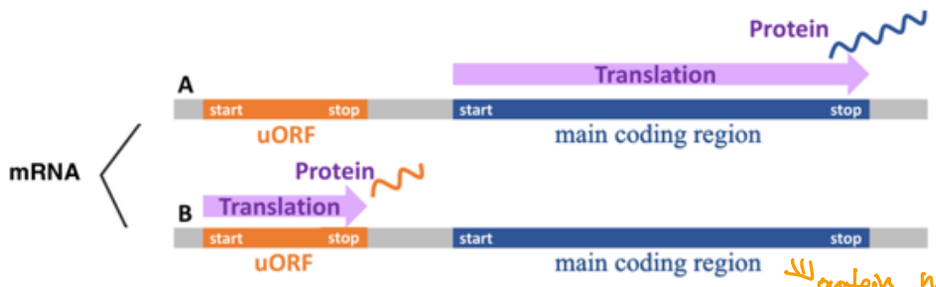
Figure 1.3 shows a graphical illustration of ribosome profiling. Take a set of mRNAs that are being translated (step 1). Isolate the part of mRNA occupied by ribosome using nucleases that degrade unprotected mRNA regions (step 2). Then we get many mRNA fragments. Map these mRNA fragments back to a reference mRNA based on the sequence of mRNA, and count the number of fragments on each base (step 3). Finally, after some technical analysis, a sequence of numbers for each base along an mRNA is obtained, called a ribosome footprint [11].

1.1.3 Upstream Open Reading Frame

As shown in Figure 1.2, the coding region is used to produce protein. The 5' UTR and 3' UTR are usually not translated. However, there is a small region within 5' UTR that might be translated, called upstream open reading frame (uORF).

As there are two regions which might be translated, we called the long coding region the main coding region (see Figure 1.4 (1)). This project focuses on identifying the locations of uORFs and main coding regions within mRNAs.

(1) Two possible translated regions in mRNA.



protein not synthesized
disease

(2) Regulation by uORF

Figure 1.4: (1): There are two regions which might be translated, uORF and main coding region. (2): Two examples of uORF regulatory mechanism. Case A: translation at main coding region produces functional protein. Case B: translation at uORF suppresses the translation at main coding region.

Translation at uORFs has been known to affect translation at downstream main coding regions [2] [3] [4]. As shown in Figure 1.4 (2), the translation process recognises and translates the uORF. If translation termination of the uORF is efficient, the downstream main coding region is not translated. Then the corresponding protein is not synthesized, which may cause diseases. It is still possible that both regions are translated. No translation at both region could also occur.

However, systemic investigations of the uORF regulatory mechanism has been challenging, mainly due to a lack of appropriate annotations of translated uORFs. Hence, it is critical to identify the locations of translated uORFs.

1.2 riboHMM

There has been recent (2016) work done around finding the locations of translated regions within mRNA [12]. This is introduced below.

Raj et al [12] developed a mixture hidden Markov models, and produced a software

package, called *riboHMM*, for identifying translated regions using ribosome profiling data. They discovered 7273 novel translated coding regions, including 2442 translated uORFs.

The model in *riboHMM* is designed to find one translated region within one mRNA. Hence, the identification of translated uORFs and main coding region is separated.

1.3 Objective

To allow identification of the locations of translated uORFs and main coding regions within mRNAs at the same time, we have decided to develop a new model that can jointly capture the structure in ribosome profiling data around translated uORFs and downstream main coding regions.

Chapter 2

Method

In this chapter, we first describe how the model is constructed based on the characteristic structure of ribosome footprints. Then we present the model set-up, followed by the derivation of our model.

2.1 How we construct the model

In this section, we begin with the characteristic structure of ribosome footprints. Then based on this structure, we show how we construct the new model used in this thesis.

2.1.1 Characteristic structure of ribosome footprints

A ribosome profiling footprint is represented as a sequence of counts along bases in an mRNA, which measures how frequently each base has been translated. More abundant counts within translated uORFs and main coding regions compared to the 5' and 3' UTRs are expected [13].

Data pre-processing on ribosome footprint data was performed by Raj et al. [12]. They check patterns in ribosome footprint data around the start and stop sites of the main coding regions by making aggregate plots. The plot in Figure 2.1 involves the following steps:

- Step 1: Consider a set of mRNA transcripts whose main coding regions are annotated in GENECODE. Take the ribosome footprints on these mRNA transcripts.

- Step 2: Choose a fixed length as display window size for the aggregate plot. Extract 2 subsequences of data with this length centering at the start and stop sites of the main coding region from each mRNA transcript. Then we obtain 2 sets of equal-length subsequences of ribosome footprints. One set centers at the start site and the other centers at the stop site.

- Step 3: For each set of subsequences of ribosome footprints, compute the proportions of count numbers for each base.

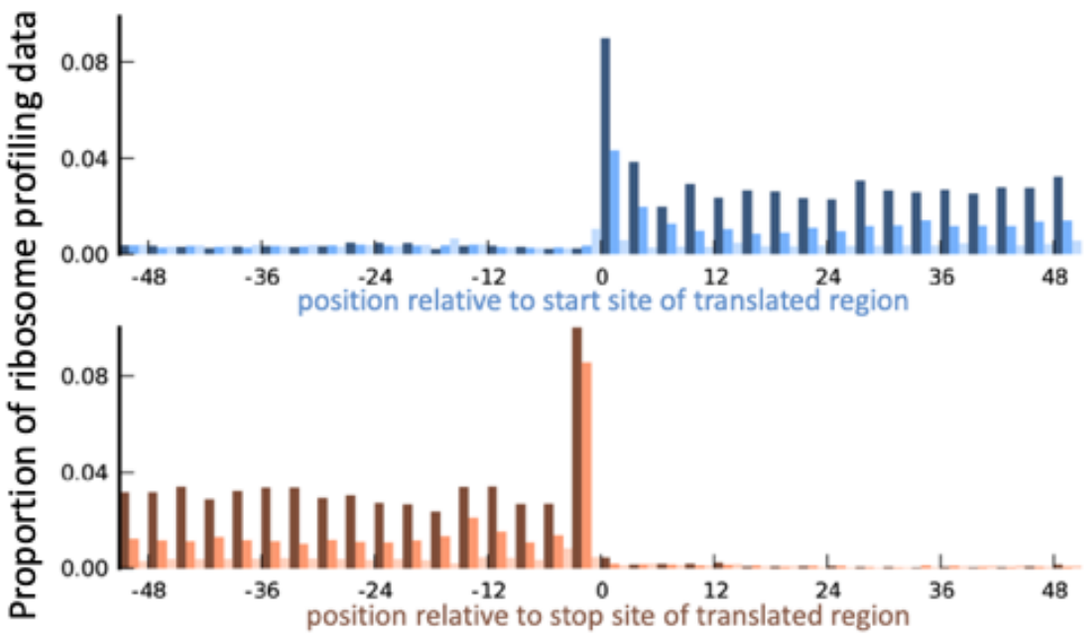


Figure 2.1: Proportion of ribosome profiling counts aggregated across highly expressed annotated coding transcripts, centered at the start site of the translation start (blue) and stop (orange) sites. In aggregate, ribosome profiling count data have higher abundance within the translated region than the untranslated regions and exhibit a 3-base periodicity within the translated region. Reproduced from [12].

The proportion of ribosome footprints centering at the start and stop site of the main coding region is given in Figure 2.1. In the figure on the top, the position is relative to the annotated start site of main coding region. The right part is the first 50 bases of main coding region. In line with their expectations, a higher abundance of counts within the coding regions were observed. Moreover, the ribosome footprints exhibit

a 3-base periodicity within the main coding region. This is because the ribosome decodes 3 bases (a codon) at a time, and spends more time for decoding rather than moving between codons during translation. The same structure is shown in the figure on the bottom. We also notice that the counts at start and stop sites are higher than the rest of counts in the main coding region.

We develop a model that capture this characteristic structure of ribosome footprints, and use the model to infer the locations of translated regions (either uORFs or main coding regions) in mRNA transcripts.

2.1.2 Construction of hidden Markov model

We observe the ribosome footprint count at each base of an mRNA, which carries unobserved information about translation. Based on the ribosome footprint counts, we wish to know the locations of translated regions in an mRNA. This situation fits the framework of hidden Markov model [14] [15].

Consider an mRNA being translated. We can form a sequence of states regarding the information of translation. As the information of each base in translation is unknown, this sequence corresponds to a sequence of hidden states. As for ribosome footprints, we have the number of counts for each base that depends on the state of this base in translation.

Next, we construct the model by defining the hidden state space, the transition probabilities for the underlying Markov chain and the emission distributions for the observed ribosome footprints.

Hidden state space

As shown in Figure 2.2, we consider an mRNA whose uORF and main coding region are translated. We first partition bases of mRNA into 2 large groups, an untranslated group (grey stripes) and a translated group (colored stripes). The untranslated group contains 3 states: 5'U, 5'U2, and 3'U standing for front-end, middle, and back-end untranslated regions respectively. The translated group has 2 parts, translated uORF and main coding region. Due to the differences in counts

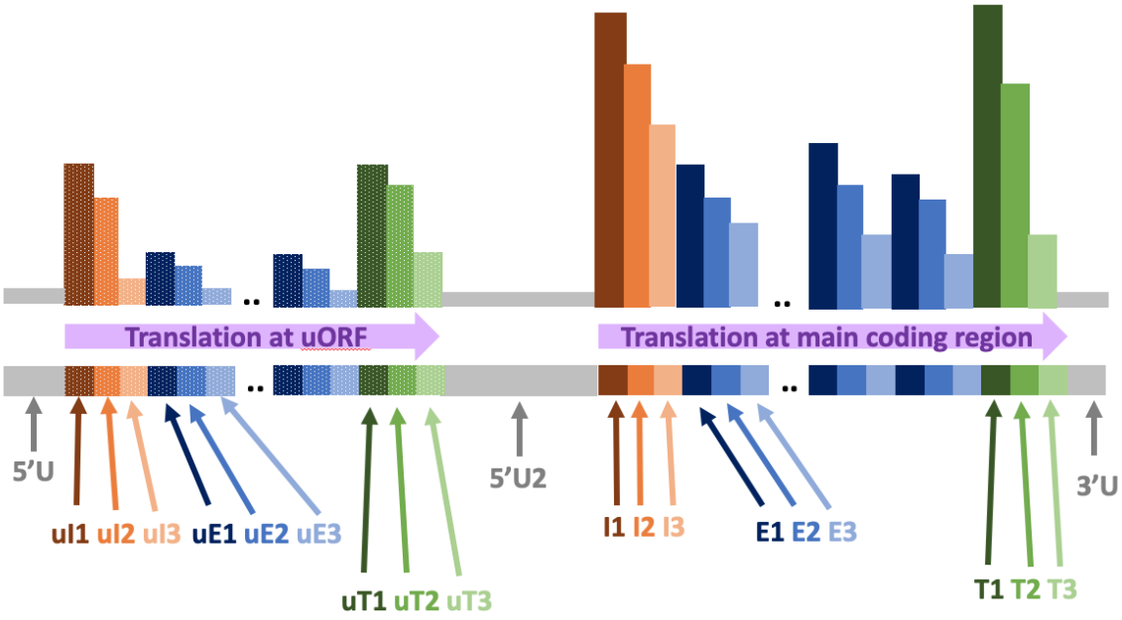


Figure 2.2: Hidden states of an mRNA whose uORF and main coding region are translated

at the start, middle and stop sites of the translated region, we divide the main coding region into 3 chunks: initiation (I), elongation (E) and termination (T), which consists of the first 3 bases (orange stripes), the bases in the middle (blue stripes) and the last 3 bases (green stripes). Because of three-base periodicity, we put 3 states for each chunk. Hence, we have 9 states for main coidng region, denoted by I1, I2, I3, E1, E2, E3, T1, T2, and T3. Similarly, we use uI1, uI2, uI3, uE1, uE2, uE3, uT1, uT2 and uT3 to denote the states in translated uORF. A summary of state space is shown in Figure 2.3.

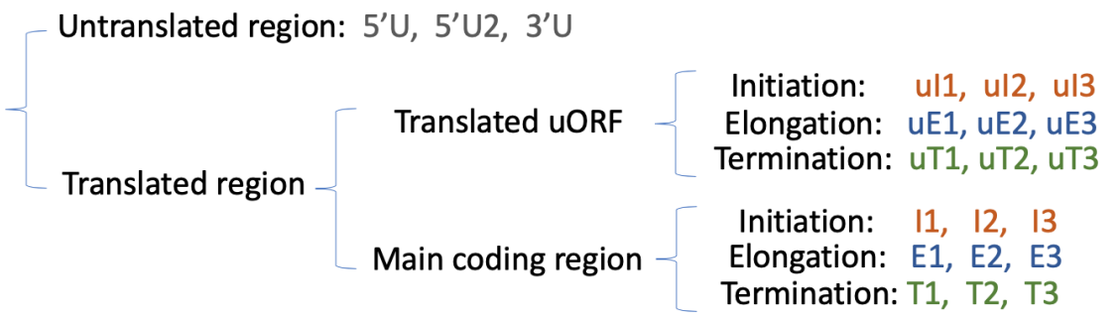


Figure 2.3: Summary of state space.

Underlying Markov chain

To allow each mRNA transcript to have 1) both translated uORF and main coding region, 2) only a translated uORF, 3) only a translated main coding region, or 4) none of them, we consider possible transitions between hidden states as shown in Figure 2.4.

Even though we have many states for the bases of mRNA, most of the movements between states are deterministic. A grey arrow without a parameter attached, in the transition diagram, indicates the transition with probability 1. For example, within the translated region, if the current state is uI1, then it will move along the sequence uI2, uI3, uE1, uE2 and uE3 with probability 1.

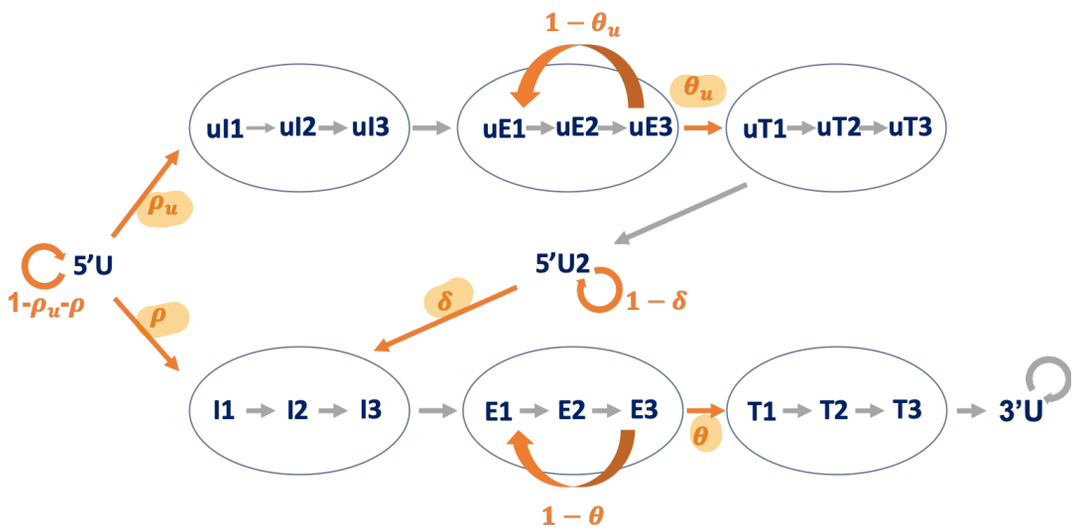


Figure 2.4: Transition diagram for the hidden Markov chain

As shown in the Figure 2.4, we only have 5 parameters for transition probabilities. In this thesis, we assume that translation terminates at the first in-frame canonical stop codon, so we model parameters θ_u and θ as functions of the mRNA sequence. At any base, if the next three bases form a canonical stop codon, one of {UAG, UAA, UGA}, then $\theta_u = \theta = 1$. Otherwise, $\theta_u = \theta = 0$. The other three parameters ρ_u, ρ and δ are constants.

notation is a bit unclear. two translations are seperated. whenever meet stop codon. current $\theta_u / \theta = 1$

Assumption: An mRNA is assumed always to start with 5'U. It will stay at 5'U for some amount of time, then jump to uI1 or I1. For either translated region, the base would experience a 3-state initiation sequence, a multiple of 3-state elongation sequences and a 3-state termination sequence. If the Markov chain jumps to translated uORF, it will leave

uORF and enter 5'U2. After staying at 5'U2 for a while, it then enters the main coding region. Finally, the Markov chain enters the absorbing state 3'U, marking the end of the translation process.

Emission distribution

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

In our case, the observed data is ribosome footprint count data. A natural choice for modelling this data is with a Poisson distribution. However, a limitation of the Poisson distribution is that its mean and variance are the same. Count data that violates this restriction leads to the issue of over-dispersion, if a Poisson distribution is used. We avoid this issue by using a random rate for the Poisson distribution.

Suppose at each base, given the hidden state, the distribution of ribosome footprint counts follows a Poisson distribution with a random rate that follows a gamma distribution. Then the variance of ribosome footprint counts is allowed to be different to its mean.

We correct for differences in ribosome footprint abundance across transcripts due to differences in transcript expression levels, by including a transcript-specific normalisation factor in the random rate of Poisson distribution.

2.2 Model set-up

Consider a set of N mRNA transcripts where the nth transcript has length of $T^{(n)}$. Our data consist of 3 types of data: ribosome footprint counts $\boldsymbol{X} = (\boldsymbol{X}^{(n)})_{n=1}^{N}$, e.g. (5, 2, 1, ⋯) mRNA sequences $\boldsymbol{S} = (\boldsymbol{S}^{(n)})_{n=1}^{N}$ e.g. (A, C, G, ⋯) and gene expression levels $E = (E^{(n)})_{n=1}^{N}$, where $\boldsymbol{X}^{(n)}$ and $\boldsymbol{S}^{(n)}$ are vector quantities and $E^{(n)}$ is a scalar.

Let $\boldsymbol{X}^{(n)} = (X_1^{(n)}, X_2^{(n)}, \cdots, X_{T^{(n)}}^{(n)})$ and $\boldsymbol{S}^{(n)} = (S_1^{(n)}, S_2^{(n)}, \cdots, S_{T^{(n)}}^{(n)})$, where $X_t^{(n)}$ and $S_t^{(n)}$ denotes the ribosome footprint count and the mRNA sequence at the tth position in the nth transcript, respectively.

As mentioned in previous section, we model the footprint data using a hidden

Why can f: no partition into 3-base

Markov model that incorporates mRNA sequences and gene expression levels. Assuming independence across transcripts, the probability of footprint data $\boldsymbol{X}$ given mRNA sequences $\boldsymbol{S}$ and gene expression levels E is written as

$$\mathbb{P}(\boldsymbol{X}|\boldsymbol{S}, E, \theta) = \sum_{n=1}^{N} \mathbb{P}(\boldsymbol{X}^{(n)}|\boldsymbol{S}^{(n)}, E^{(n)}, \theta),$$

where θ denotes the set of model parameters.

e.g. (5'U, ⋯)

For the nth transcript, we introduce a length $T^{(n)}$ sequence, $\boldsymbol{Z}^{(n)} = (Z_1^{(n)}, Z_2^{(n)}, \cdots, Z_{T^{(n)}}^{(n)})$, where $Z_t^{(n)}$ denotes the hidden state for the tth position in the nth transcript. The transition probabilities are set up as in previous section. Recall that transition probabilities θ_u and θ depend on mRNA sequence, so the underlying Markov chain is not time-homogeneous, i.e., the transition probabilities depend on position t.

To be convenient, we label the states in order by numbers 1, 2, $\cdots$, 21, as in Table 2.1.

State	5'U	uI1	uI2	uI3	uE1	uE2	uE3	uT1	uT2	uT3	
Label	1	2	3	4	5	6	7	8	9	10	
State	5'U2	I1	I2	I3	E1	E2	E3	T1	T2	T3	3'U
Label	11	12	13	14	15	16	17	18	19	20	21

Table 2.1: Label the states by numbers.

By doing so, we avoid writing $Z_t^{(n)} = 5'\text{U}$ and $Z_{t+1}^{(n)} = \text{uI1}$, but use $Z_t^{(n)} = 1$ and $Z_{t+1}^{(n)} = 2$ instead. In this case, the corresponding transition probability can be written as $a_{Z_t^{(n)}, Z_{t+1}^{(n)}} = a_{1,2}$.

$P(X_t^n | \lambda_{z_t}^{(n)}, z^{(n)})$.
$P(\lambda_{z_t}^{(n)} | \nu_z, m_z)$.
$P(X_t^{(n)} | E^{(n)}, \nu_z, m_z)$
$= \int P(X | \lambda E^{(n)}, \nu) P(\lambda | \nu, m)$

The emission distribution is modeled as follows.

$$X_t^{(n)} | Z_t^{(n)} = z \sim \text{Poi}(\lambda_{z,t}^{(n)} E^{(n)}),$$
$$\lambda_{z,t}^{(n)} \sim \text{Gamma}(\nu_z, m_z),$$

where ν_z and m_z are shape and mean parameters of the gamma distribution respectively. The density of the gamma distribution is

$\alpha = \nu$, $\beta = \frac{\nu}{m}$
$\Rightarrow$ mean $= \frac{\alpha}{\beta} = m$
variance $= \frac{\alpha}{\beta^2} = \frac{m^2}{\nu}$

$$f(\lambda|\nu, m) = \frac{1}{\Gamma(\nu)} \left(\frac{\nu}{m}\right)^{\nu} \lambda^{\nu-1} \exp\left(-\frac{\nu}{m}\lambda\right)$$

A graphical representation of the modelling of emission distributions is shown in

Figure 2.5

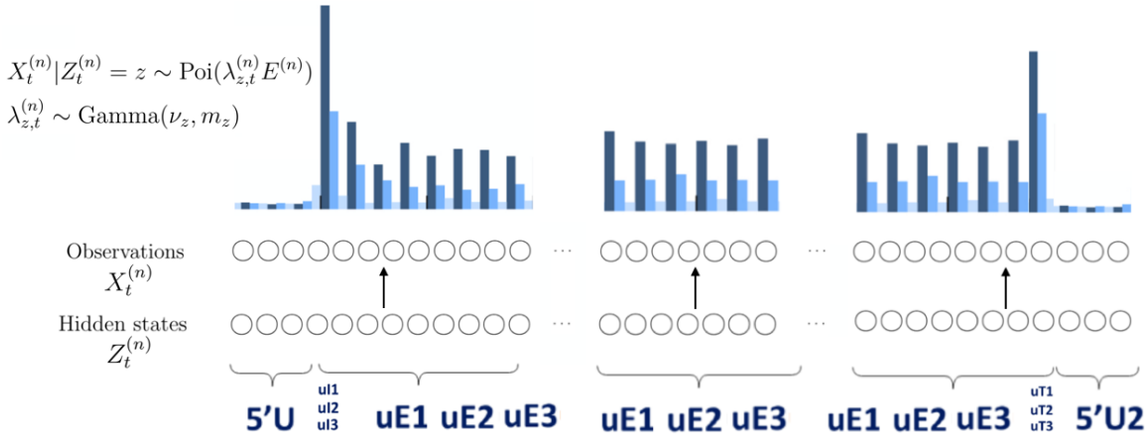


Figure 2.5: Distributions of observed ribosmome footprints depend on the hidden states (only display the part of translated uORF).

2.3 Some features of emission distribution

Since transcripts are assumed to be independent, we consider one transcript to study the emission distribution. In this section, we drop the superscripts of $\boldsymbol{X}^{(n)}$, $\boldsymbol{S}^{(n)}$ and $E^{(n)}$.

The Poisson rate is modeled by a gamma distribution with shape and mean parameterization instead of shape and rate parameterization. We use this parameterization so that the mean of λ (i.e., m_z) can be directly known.

The mean of $X_t|Z_t = z$ is dependent on the mean of $\lambda_{z,t}$ (i.e., m_z) in a simple form. Let $X_{z,t} := X_t|Z_t = z$ for simplicity. Then,

$$\mathbb{E}(X_{z,t}) = \mathbb{E}[\mathbb{E}(X_{z,t}|\lambda_{z,t})] = \mathbb{E}(\lambda_{z,t}E) = m_z E.$$

The variance of $X_{z,t}$ is given by,

$$\begin{aligned}\mathbb{V}\text{ar}(X_{z,t}) &= \mathbb{E}[\mathbb{V}\text{ar}(X_{z,t}|\lambda_{z,t})] + \mathbb{V}\text{ar}[\mathbb{E}(X_{z,t}|\lambda_{z,t})] \\ &= \mathbb{E}(\lambda_{z,t}E) + \mathbb{V}\text{ar}(\lambda_{z,t}E) \\ &= m_z E + E^2 \frac{m_z^2}{v_z}.\end{aligned}$$

mean $= \alpha$. variance $= \frac{\alpha}{\beta}$

$\text{Var}(Z) = \text{Var}(E(Z|\mu)) + E(\text{Var}(Z|\mu))$

$\text{Var}(Y|Z=z) = \text{Var}(E((Y|Z=z)|\mu))$
$+ E(\text{Var}((Y|Z=z)|\mu))$
$= \text{Var}(\mu E^n) + E(\mu E^n)$
$= (E^n)^2 \text{Var}(\mu) + E^n E(\mu)$
$= (E^n)^2 \frac{\alpha}{\beta} + E^n \alpha$

$\alpha = v$, $\beta = \frac{v}{m}$
mean $= \frac{\alpha}{\beta} = m$
variance $= \frac{\alpha}{\beta^2} = \frac{m^2}{v}$

We see that the variance of $X_{z,t}$ has negative relationship with ν_z. As $\nu_z \to \infty$, the variance of $X_{z,t}$ becomes $m_z E$, which is the same as the expectation of $X_{z,t}$. In this case, the random rate degenerates to a constant value $\lambda_{z,t}$, and the emission distribution degenerates to a Poisson distribution with constant rate $\lambda_{z,t}E$.

The above formulation of the emission distribution of $X_{z,t}$ is in a hierarchical form. By integrating over the random Poisson rate, we could get an equivalent single emission distribution of $X_{z,t}$. Given the hidden state, $Z_t = z$, the distribution of the observed ribosome footprint count, $X_{z,t}$, follows negative binomial distribution:

$X_{z,t} | Z = x_{z,t}$

$$X_{z,t} \sim \mathrm{NB}\left(\nu_z, \frac{\nu_z}{m_z E + \nu_z}\right),$$

where the negative binomial NB(r, p) has probability mass function

$$g(x|r,p) = \binom{x+r-1}{x} p^r (1-p)^x.$$

Poisson - Gamma NB

$x_i \sim \mathrm{Poiss}(\lambda_i) \Rightarrow p(x_i|\lambda_i)$.

$\lambda_i \sim \mathrm{Gam}(\alpha, \beta) \Rightarrow p(\lambda_i|\alpha,\beta)$

$p(x_i|\alpha,\beta) = \int p(x_i, \lambda_i|\alpha,\beta)\, d\lambda_i$

$= \int \underset{P}{p(x_i|\lambda_i,\alpha,\beta)}\, \underset{G}{p(\lambda_i|\alpha,\beta)}$

$= NB(\alpha, \beta)$.

2.4 Likelihood

To estimate model parameters, we need to compute the log-likelihood. As mRNA transcripts are assumed to be independent, we first derive the log-likelihood for a single mRNA transcript, before extending it to multiple mRNA transcripts.

In the previous section, in order to better visualise the mean of the random Poisson rate $\lambda_{z,t}$ and the mean of $X_{z,t}$, we use the shape ν_z and mean m_z parameterization for the gamma distribution. However, in this section, for computational convenience, we begin by using the shape α_z and rate β_z parameterization for the gamma distribution in the likelihood. After estimation, we convert α_z and β_z to ν_z and m_z. The connection between these two parameterization is given by:

$$\nu_z = \alpha_z, \qquad m_z = \frac{\alpha_z}{\beta_z}.$$

Hence, with the shape α_z and rate β_z parameterization for the gamma distribution, the emission distribution is as follows,

$$X_{z,t} \sim \text{Poi}(\lambda_{z,t}E),$$
$$\lambda_{z,t} \sim \text{Gamma}(\alpha_z, \beta_z),$$

Equivalently, in single distribution form,

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

In the likelihood expression, we use lower case $\boldsymbol{x}$ for ribosome footprint data as the data is assumed observed and non-random. In contrast, we still use capital $\boldsymbol{Z}$ for the unobserved hidden state sequence.

Complete log-likelihood for a single transcript:

Denote $\boldsymbol{x}$ and $\boldsymbol{Z}$ by $\boldsymbol{x}_{1:T}$ and $\boldsymbol{Z}_{1:T}$ respectively to incorporate the length of the transcript.

$$\log f(\boldsymbol{x}_{1:T}, \boldsymbol{Z}_{1:T}|\theta) = \log\left[f(x_{1:T} \mid Z_{1:T}, \theta) \cdot f(Z_{1:T} \mid \theta)\right] \quad \text{(HMM notes P2)}$$
$$= \log\left\{\pi_{Z_1}\left(\prod_{t=2}^{T} a_{Z_{t-1}Z_t}\right)\prod_{t=1}^{T} g\left(x_t \middle| \alpha_{z_t}, \frac{\beta_{Z_t}}{E+\beta_{Z_t}}\right)\right\}$$
$$= \log \pi_{Z_1} + \sum_{t=2}^{T} \log a_{Z_{t-1}Z_t} + \sum_{t=2}^{T} \log h(x_t \middle| \alpha_{Z_t}, \beta_{Z_t}, E),$$

where π_{Z_1} is the initial probability for state 1, g is the probability mass function of a negative binomial distribution, and $h(x \middle| \alpha, \beta, E) = g\left(x \middle| \alpha, \frac{\beta}{E+\beta}\right)$.

Complete log-likelihood for multiple transcripts

Now we extend to multiple transcripts. Again, we incorporate transcript length by denoting $\boldsymbol{x}^{(n)}$ and $\boldsymbol{Z}^{(n)}$ by $\boldsymbol{x}^{(n)}_{1:T}$ and $\boldsymbol{Z}^{(n)}_{1:T}$. Note that each transcript might have a different length, so T is dependent on n. To be less cumbersome, we omit the superscript of $T^{(n)}$ in the formulation below.

$$
\begin{aligned}
&\log \prod_{n=1}^{N} f(\boldsymbol{x}_{1:T}^{(n)}, \boldsymbol{Z}_{1:T}^{(n)}|\theta) \\
&= \sum_{n=1}^{N} \log f(\boldsymbol{x}_{1:T}^{(n)}, \boldsymbol{Z}_{1:T}^{(n)}|\theta) \\
&= \sum_{n=1}^{N} \left(\log \pi_{Z_1^{(n)}} + \sum_{t=2}^{T} \log a_{Z_{t-1}^{(n)} Z_t^{(n)}} + \sum_{t=2}^{T} \log h(x_t^{(n)} \,\big|\, \alpha_{Z_t^{(n)}}, \beta_{Z_t^{(n)}}, E^{(n)}) \right)
\end{aligned}
$$

add logistic

2.5 Expectation-Maximization algorithm

The Expectation-Maximization (EM) algorithm consists of 2 steps, expectation and maximization [16]. In the expectation step, the unknown hidden variables $\boldsymbol{Z}$ would be averaged out of the complete log-likelihood, resulting in the so called Q-function. This function is then optimized in the maximization step.

Iteration of these 2 steps increases the observed data likelihood function. By performing such iterative procedures with multiple initial values for θ, estimated model parameters $\hat{\theta}$ are obtained.

2.5.1 Expectation step

Similar to the previous section, we calculate the expected complete log-likelihood (Q-function) for a single transcript, and then extend to multiple transcripts.

Let θ and θ' be the current and previous estimated model parameters, respectively.

Q-function of a single transcript: $\log f(x_{1:T}, Z_{1:T}|\theta) = \log \pi_{Z_1} + \sum_{t=2}^{T} \log a_{Z_{t-1}, Z_t} + \sum_{t=2}^{T} \log h(x_t|\alpha_{Z_t}, \beta_{Z_t}, E)$

$$
\begin{aligned}
Q(\theta|\theta') = \; & \mathbb{E}_{\boldsymbol{Z}} \left(\log f(\boldsymbol{x}_{1:T}, \boldsymbol{Z}_{1:T}|\theta) \,|\, x_{1:T}, \theta' \right) \\
= \; & \sum_{k} (\log \pi_k) \, \mathbb{P}(Z_1 = k \,|\, x_{1:T}, \theta') \\
& + \sum_{t=2}^{T} \sum_{k,j} (\log a_{kj}) \mathbb{P}(Z_{t-1} = k, Z_t = j \,|\, \boldsymbol{x}_{1:T}, \theta')
\end{aligned}
$$

$$+\sum_{t=1}^{T}\sum_{k}(\log h(x_t|\alpha_k,\beta_k,E))\mathbb{P}(Z_t=k|\boldsymbol{x}_{1:T},\theta')$$

$\gamma_t(i) = P(i_t = q_i \mid O, \lambda)$ $\qquad$ $\xi_t(ij) = P(i_t = q_i,\ i_{t+1} = q_j \mid O, \lambda)$ $\qquad$ p9 HMM

Let $L_k(t) = \mathbb{P}(Z_t = k|\,\boldsymbol{x}_{1:T},\theta')$ and $H_{kj}(t) = \mathbb{P}(Z_{t-1} = k, Z_t = j|\,\boldsymbol{x}_{1:T},\theta')$, then

expectation of hidden states

$$\begin{aligned}
Q(\theta|\theta') &= \mathbb{E}_{\boldsymbol{Z}}(\log f(\boldsymbol{x}_{1:T},\boldsymbol{Z}_{1:T}|\theta)|\,x_{1:T},\theta') \\
&= \boxed{\sum_{k}}(\log\pi_k)\,L_k(1)+\sum_{t=2}^{T}\boxed{\sum_{k,j}}(\log a_{kj})H_{kj}(t) \\
&\quad+\sum_{t=1}^{T}\boxed{\sum_{k}}(\log h(x_t|\alpha_k,\beta_k,E))L_k(t)
\end{aligned}$$

Q-function of multiple transcripts:

Let

$$\begin{aligned}
L_k^{(n)}(t) &= \mathbb{P}(Z_t^{(n)} = k|\,\boldsymbol{x}_{1:T}^{(n)},\theta') \\
H_{kj}^{(n)}(t) &= \mathbb{P}(Z_{t-1}^{(n)} = k, Z_t^{(n)} = j|\,\boldsymbol{x}_{1:T}^{(n)},\theta').
\end{aligned}$$

Then

$$\begin{aligned}
Q(\theta|\theta') &= \mathbb{E}_{\boldsymbol{Z}}\left(\log\prod_{n=1}^{N} f\left(\boldsymbol{x}_{1:T}^{(n)},\boldsymbol{Z}_{1:T}^{(n)}\middle|\theta\right)\middle|\,x_{1:T}^{(1)},\cdots,x_{1:T}^{(n)},\theta'\right) \\
&= \sum_{n=1}^{N}\sum_{k}(\log\pi_k)\,L_k^{(n)}(1)+\sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{k,j}(\log a_{kj})H_{kj}^{(n)}(t) \\
&\quad+\sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{k}\left(\log h(x_t^{(n)}|\alpha_k,\beta_k,E^{(n)})\right)L_k^{(n)}(t)
\end{aligned}$$

There is no easy formula for calculating the probabilities $L_k^{(n)}(t) = \mathbb{P}(Z_t^{(n)} = k|\,\boldsymbol{x}_{1:T}^{(n)},\theta')$ and $H_{kj}^{(n)}(t) = \mathbb{P}(Z_{t-1}^{(n)} = k, Z_t^{(n)} = j|\,\boldsymbol{x}_{1:T}^{(n)},\theta')$ in the Q-function. We will describe below the forward and backward algorithms that are useful when we calculate these two probabilities.

By independence of transcripts, we drop the superscript of $\boldsymbol{x}_{1:T}^{(n)}$, and calculate the $L_k(t)$ and $H_{kj}(t)$ for each transcript. Let the length of a transcript be T. There are $M = 21$ states in the state space.

Forward algorithm

Define the forward probability $F_k(t)$ as the joint probability of observing the first t variables $\boldsymbol{x}_{1:t}$, and the hidden variable at position t being in state k, i.e.,

$$\overset{=\alpha_t(i)}{F_k(t)} = \mathbb{P}(Z_t = k, \boldsymbol{x}_{1:t}), \qquad \text{p2 HMM}$$
$$1 < t \leq T,\ 1 \leq k \leq M.$$

This probability can be evaluated by the following recursive formula,

$$F_k(1) = \pi_k h(x_1|\alpha_k, \beta_k),$$
$$F_k(t) = \sum_{j=1}^{M} F_j(t-1) h(x_k|\alpha_k, \beta_k) a_{jk}(t-1), \qquad \text{p3 HMM}$$
$$1 < t \leq T,\ 1 \leq k \leq M.$$

Backward algorithm

Define the backward probability $B_k(t)$ as the conditional probability of observed data after position t, $\boldsymbol{x}_{t+1:T}$, given the hidden variable at position t being in state k, i.e.:

$$\overset{=\beta_t(i)}{B_k(t)} = \mathbb{P}(\boldsymbol{x}_{t+1:T}|Z_t = k) \qquad \text{p4 HMM}$$
$$1 \leq t \leq T-1,\ 1 \leq k \leq M.$$

Set $B_k(T) = 1$ for all $k = 1, 2, \cdots, 21$. Similar to the forward probability, the backward probability can be evaluated using the following recursion.

$$B_k(T) = 1, \qquad \text{p5 HMM}$$
$$B_k(t) = \sum_{j=1}^{M} h(x_{t+1}|\alpha_j, \beta_j) a_{kj}(t) B_j(t+1),$$
$$1 \leq t \leq T-1,\ 1 \leq k \leq M.$$

Use forward and backward algorithm results to calculate $L_k(t)$ and $H_{j,k}(t)$

Once we have the forward and backward probabilities, the probabilities $L_k(t)$ can be obtained by,

$$\begin{aligned} L_k(t) &= \mathbb{P}(Z_t = k|\boldsymbol{x}_{1:T}) \\ &= \frac{1}{\mathbb{P}(\boldsymbol{x}_{1:T})}\mathbb{P}(Z_t = k, \boldsymbol{x}_{1:T}) \\ &= \frac{1}{\mathbb{P}(\boldsymbol{x}_{1:T})}F_k(t)B_k(t), \end{aligned}$$

and $H_{j,k}(t)$,

$$\begin{aligned} H_{k,j}(t) &= \mathbb{P}(Z_t = k, Z_{t+1} = j|\boldsymbol{x}_{1:T}) \\ &= \frac{1}{\mathbb{P}(\boldsymbol{x}_{1:T})}\mathbb{P}(Z_t = k, Z_{t+1} = j, \boldsymbol{x}_{1:T}) \\ &= \frac{1}{\mathbb{P}(\boldsymbol{x}_{1:T})}F_k(t)h(x_{t+1}|\alpha_j, \beta_j)a_{k,j}(t)B_j(t+1), \end{aligned}$$

where

Check ?

$$\mathbb{P}(\boldsymbol{x}_{1:T}) = \sum_{k=1}^{M}\mathbb{P}(Z_T = k, \boldsymbol{x}_{1:T}) = \sum_{k=1}^{M}F_k(T). = \sum_{k=1}^{M}F_k(t)B_k(t)$$

$$= \sum_{k=1}^{M}\sum_{j=1}^{M}F_k(t)h(x_{t+1}|\alpha_j,\beta_j)a_{k,j}(t)B_j(t+1)$$

2.5.2 Maximization step

Note that $Q(\theta|\theta')$ is a sum of 3 parts. The first, second and third parts are functions of initial probabilities π_k, transition probabilities a_{kj} and emission parameters α_k, β_k. Maximization of each part is independent.

starts from 5'U

assume? derivation pdf ?? if find $\pi_1 \neq 1$, how to explain or how to modify?

By assumption, initial probability $\pi_1 = 1$. Then we just need to maximise the second part with respect to transition probabilities $a_{k,j}$ and the third part with respect to emission parameters α_k, β_k separately.

Transition probabilities

To maximise the part with transition probabilities, we use the Lagrange multiplier with the constraints $\sum_j a_{k,j} = 1$, and find the derivatives of $Q(\theta|\theta')$ with respect to

$$
\begin{aligned}
Q(\theta|\theta') &= \mathbb{E}_{\boldsymbol{Z}}\left(\log \prod_{n=1}^{N} f\left(\boldsymbol{x}_{1:T}^{(n)}, \boldsymbol{Z}_{1:T}^{(n)} \middle| \theta\right) \middle| x_{1:T}^{(1)}, \cdots, x_{1:T}^{(n)}, \theta'\right) \\
&= \sum_{n=1}^{N}\sum_{k}(\log \pi_k) L_k^{(n)}(1) + \sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{k,j}(\log a_{kj}) H_{kj}^{(n)}(t) \\
&\quad + \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{k}\left(\log h(x_t^{(n)}|\alpha_k, \beta_k, E^{(n)})\right) L_k^{(n)}(t)
\end{aligned}
$$

the transition probabilities. Then set the the derivatives to 0.

$$
\begin{aligned}
&\frac{\partial}{\partial a_{k,j}}\left\{Q(\theta|\theta') - \lambda\left(\sum_j a_{k,j} - 1\right)\right\} \\
&= \sum_{n=1}^{N}\sum_{t=2}^{T}\frac{\partial}{\partial a_{k,j}}(\log a_{kj}) H_{kj}^{(n)}(t) - \lambda \\
&= \sum_{n=1}^{N}\sum_{t=2}^{T}\frac{H_{kj}^{(n)}(t)}{a_{k,j}} - \lambda
\end{aligned}
$$

derivation p4

Set the partial derivative to 0,

$$
\lambda = \frac{1}{a_{k,j}}\sum_{n=1}^{N}\sum_{t=2}^{T} H_{kj}^{(n)}(t).
$$

Multiply $a_{k,j}$ on both side and sum over j, we get the value of λ,

$$
\lambda = \lambda\underbrace{\boxed{\sum_j a_{k,j}}}_{=1} = \sum_j\sum_{n=1}^{N}\sum_{t=2}^{T} H_{kj}^{(n)}(t) = \sum_{n=1}^{N}\sum_{t=2}^{T} L_k^{(n)}(t).
$$

Substitute λ, we get

$$
a_{k,j} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{kj}^{(n)}(t)}{\sum_{n=1}^{N}\sum_{t=2}^{T} L_k^{(n)}(t)},
$$

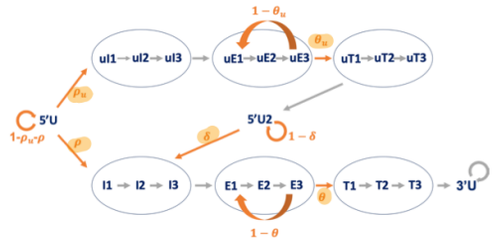
In our model setting, we only have 3 transition probabilities that need to be estimated, $\rho_u = a_{1,2}$, $\rho = a_{1,12}$ and $\delta = a_{11,12}$. The other 2 transition probabilities, θ_u and θ, are functions of the mRNA sequence. Hence,

depend on stop codon e.g. UAG, UAA, UGA

$$
\rho_u = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{1,2}^{(n)}(t)}{\sum_{n=1}^{N}\sum_{t=2}^{T} L_1^{(n)}(t)},
$$

$$
\rho = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{1,12}^{(n)}(t)}{\sum_{n=1}^{N}\sum_{t=2}^{T} L_1^{(n)}(t)},
$$

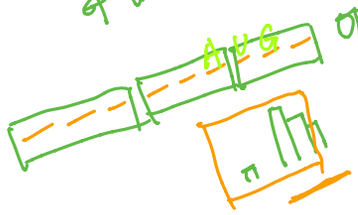
$$
\delta = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} H_{11,12}^{(n)}(t)}{\sum_{n=1}^{N}\sum_{t=2}^{T} L_{11}^{(n)}(t)}.
$$



ρ_u, ρ, δ all modify by logistic ?

Q1 if AUG sequence of multiple of three

$= \begin{cases} 1 & \text{if } Z_t = U,\ Z_{t+1} = A,\ Z_{t+2} = G \quad \text{AND} \\ 0 & \text{OR } Z_t = U,\ Z_{t+1} = A,\ Z_{t+2} = A \quad \text{AND} \\ & \text{OR } Z_t = U,\ Z_{t+1} = G,\ Z_{t+2} = A \quad \text{AND} \end{cases}$



Q2 but if not: e.g. 8

Need to consider way to group ?

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

NB(r,p) pmf: $P(x) = \binom{x+r-1}{x} p^r (1-p)^x$

$\log P(x) = \log\binom{x+r-1}{x} + r\log p + x\log(1-p)$

Emission parameters

For the part with emission parameters, we calculate the derivatives with respect to α_k and β_k. First, we need the expression for probability mass function $h(x_t^{(n)}|\alpha_k, \beta_k, E^{(n)})$,

$x = x_t^{(n)}$, $r = \alpha_z$, $p = \frac{\beta_z}{E+\beta_z}$

$$\log h(x_t^{(n)}|\alpha_k, \beta_k, E^{(n)})$$

$$= \log\binom{\alpha_k + x_t^{(n)} - 1}{x_t^{(n)}} + \alpha_k \log\left(\frac{\beta_k}{E^{(n)}+\beta_k}\right) + x_t^{(n)} \log\left(\frac{E^{(n)}}{E^{(n)}+\beta_k}\right)$$

The derivative of $Q(\theta|\theta')$ with respect to α_k is given by

$$Q(\theta|\theta') = \mathbb{E}_{\boldsymbol{Z}}\left(\log\prod_{n=1}^{N} f\left(\boldsymbol{x}_{1:T}^{(n)}, \boldsymbol{Z}_{1:T}^{(n)}|\theta\right) \middle| x_{1:T}^{(1)}, \cdots, x_{1:T}^{(n)}, \theta'\right)$$
$$= \sum_{n=1}^{N}\sum_{k}(\log\pi_k) L_k^{(n)}(1) + \sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{k,j}(\log a_{kj}) H_{kj}^{(n)}(t)$$
$$+ \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{k}\left(\log h(x_t^{(n)}|\alpha_k, \beta_k, E^{(n)})\right) L_k^{(n)}(t)$$

$$\frac{\partial}{\partial\alpha_k} Q(\theta|\theta')$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \frac{\partial}{\partial\alpha_k} \log h_k(x_t^{(n)}|\alpha_k, \beta_k, E^{(n)})$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \left(\frac{\partial}{\partial\alpha_k} \log\binom{\alpha_k + x_t^{(n)} - 1}{x_t^{(n)}} + \log\left(\frac{\beta_k}{E^{(n)}+\beta_k}\right)\right)$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \left(\frac{\partial}{\partial\alpha_k} \log\left\{\prod_{s}^{x_t^{(n)}} (\alpha_k + x_t^{(n)} - s)\right\} + \log\left(\frac{\beta_k}{E^{(n)}+\beta_k}\right)\right)$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \left(\sum_{s}^{x_t^{(n)}} \frac{\partial}{\partial\alpha_k} \log(\alpha_k + x_t^{(n)} - s) + \log\left(\frac{\beta_k}{E^{(n)}+\beta_k}\right)\right)$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \left(\sum_{s}^{x_t^{(n)}} \frac{1}{\alpha_k + x_t^{(n)} - s} + \log\left(\frac{\beta_k}{E^{(n)}+\beta_k}\right)\right)$$

The derivative of $Q(\theta|\theta')$ with respect to β_k is given by

$$\frac{\partial}{\partial\beta_k} Q(\theta|\theta')$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \frac{\partial}{\partial\beta_k} \log h_k(x_t^{(n)}|\alpha_k, \beta_k, E^{(n)})$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T} L_k^{(n)}(t) \frac{\partial}{\partial\beta_k} \left\{\alpha_k \log\beta_k - (\alpha_k + x_t^{(n)}) \log(E^{(n)} + \beta_k)\right\}$$

$$= \sum_{n=1}^{N} \sum_{t=1}^{T} L_k^{(n)}(t) \left(\frac{\alpha_k}{\beta_k} - \frac{\alpha_k + x_t^{(n)}}{E^{(n)} + \beta_k} \right)$$

If we set the derivatives to 0, there is no closed-form solution for both cases. In the maximisation step for the emission parameters, we will use a numerical method with the aid of derivatives calculated above.

2.6 Viterbi Algorithm

The final goal of the methods is to find the locations of translated uORFs and main coding regions within mRNAs. In term of hidden Markov model described above, we want to identify the sequence of states that maximize the posterior likelihood $\mathbb{P}(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{S}, E, \hat{\theta})$.

Once we apply the EM algorithm to estimate the parameters in the hidden Markov model, we can calculate the posterior likelihood for every possible sequence of states by the estimated parameters. The one with largest posterior likelihood would be our desired optimal path. This criterion is called the rule of Maximum A Posteriori (MAP). The optimal sequence found based on this criterion is called MAP path.

However, there are a significantly large number of paths to be considered, making a brute force approach unrealistic. We apply the Viterbi algorithm to find the MAP path. First, we demonstrate how the Viterbi algorithm works by a toy example. Then we apply the Viterbi algorithm in the same way on our model.

A toy example for how the Viterbi algorithm works

Suppose we have state space {A, B, C}. First, we start with trivial sub-paths of length 1. There are 3 possible sub-paths, $Z_1 = A$, $Z_1 = B$, $Z_1 = C$, (see Figure 2.6).

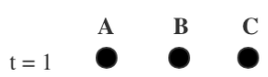


Figure 2.6: Start with trivial sub-paths of length 1.

Now extend to length 2. First suppose $Z_2 = A$, calculate the posterior likelihood of previous states, choosing the best preceding state, say $Z_1 = C$. Then we obtain an optimal sub-path of length 2, CA. Note that we assume the last state is A. Repeat for $Z_2 = B$ and $Z_2 = C$. We then get 3 sub-paths of length 2 assuming the last state is known (see Figure 2.7 (left)).

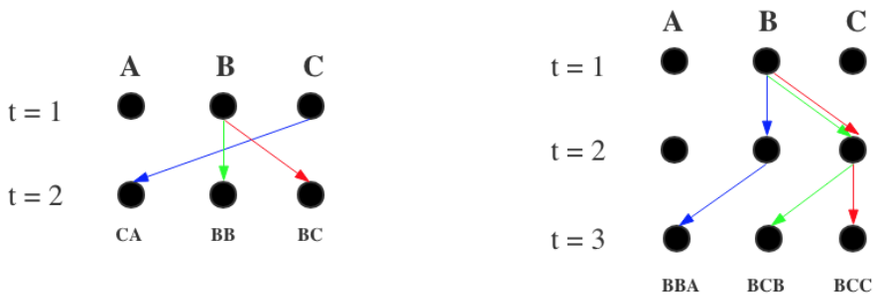


Figure 2.7: Sub-path of length 2 and 3.

We can do such a process for $t = 3, 4, \cdots, T$. One example of length 3 sub-paths is given in Figure 2.7 (right). At the end, we get 3 full-length best paths ending at A, B and C. Amongst these 3 paths, by choosing the one with largest posterior likelihood, we eventually get the optimal path.

In our case

For our model on ribosome footprint data, the Viterbi algorithm works in the same way as the example above, but just with more states. The main idea is to consider optimal sub-paths, and extend the sub-paths in length to get the optimal full-length path. For the correctness of Viterbi algorithm, see the paper *The Viterbi Algorithm* by G. David Forney Jr [17].

For the MAP path we found for each of the transcripts, we will retain transcripts whose MAP path contains a pair of translation start and stop sites (either for uORF or main coding region) and has a high posterior likelihood. We will find translated uORFs or main coding regions within mRNAs using the retained MAP paths.

Chapter 3

R package

This chapter introduces an R package *riboHMM2*. We demonstrate how the R package is implemented, followed by two data sets generated for simulation study. A vignette in the last section explains how to use the R package *riboHMM2*.

This R package is designed to identify the locations of translated upstream Open Reading Frames (uORFs) and the main coding regions in mRNAs.

Given the ribosome footprint count data of a set of mRNA transcripts, together with the corresponding mRNA sequences, one can use *riboHMM2* to estimate the parameters in the model constructed in previous chapter, and further to obtain the the locations of translated uORFs and the main coding regions in mRNAs.

The R package *riboHMM2* can be downloaded and installed in R by runing the following code.

```
library(devtools)
install_github("shimlab/riboHMM2")
```

3.1 Implementation

In this section, the main workflow of implementation is presented. Readers who want to reproduce the package could follow the workflow. Otherwise, you might want to skip this section, and jump to the next section for the data sets we included

in the R package.

However, many details of the implementation are omitted as the practical implementation would depend on the programming language used. We only present the workflow by listing the algorithms and functions used in pseudo codes. If readers want to know the details of how to implement the R package, they can download the package, and check the codes.

Baum - Welch

The *riboHMM2* model is implemented in two parts. First, we need to estimated the parameters from a given ribosome footprint count data. Then the estimated parameters are used to infer the locations of translated uORFs and main coding regions in mRNAs.

Viterbi

3.1.1 Estimation

Before we start, we set some global notations to avoid defining them in each algorithm,

$$n := \text{length of mRNA (varied across mRNAs)}$$

$$M := \text{number of hidden states} = 21 \ \ \text{(fixed)}$$

The EM algorithm is applied to estimate the parameters in the model. As the mRNA transcripts are usually very long, many terms in the log-likelihood would be driven to values that are smaller than the computer can represent in its memory. The numerical errors accumulate as the length of mRNA increases, which would lead to an incorrect result. This arithmetic underflow issue can be fixed by exploiting the *LogSumExp* trick.

In the expectation step, we need the probabilities $L_k(t)$ and $H_{j,k}(t)$, which can be calculated efficiently with the help of forward and backward algorithm.

Hence, we start with forward algorithm to calculate the forward probabilities $F_k(t)$ (see Algorithm 1 in Appendix A). Similarly, we use backward algorithm to calculate the backward probabilities $B_k(t)$ in log scale (see Algorithm 2 in Appendix A). To use *LogSumExp* trick, we perform calculation in log scale.

$\left\{ \begin{array}{l} F_k(t) \\ B_k(t) \end{array} \right. \Rightarrow \left\{ \begin{array}{l} L_k(t) \\ H_{kj}(t) \end{array} \right. \Rightarrow \left\{ \begin{array}{l} \mu \\ \rho \\ \delta \end{array} \right. \Rightarrow Q: \left\{ \begin{array}{l} \text{closed-form} \\ \text{numerical} \end{array} \right. \Rightarrow$ Viterbi

Now we have forward and backward probabilities in log scale, which can be used to calculate the probabilities $L_k(t)$ and $H_{kj}(t)$ (see Function 3 4 in Appendix A). For $H_{kj}(t)$, we only need $H_{1,2}(t)$, $H_{1,12}(t)$ and $H_{11,12}(t)$. This would be used to estimate the transition probabilities ρ_u , ρ, δ. Recall that θ_u and θ are functions of mRNA sequences. We do not need to estimate them.

Up to this step, we have all terms needed in the Q-function. We can perform maximization step to update the parameters. As mentioned in chapter 2, transition probabilities and emission parameters can be updated separately. We create two functions for updating parameters (see Function 5, 6 in Appendix A). The function for updating transition probabilities uses the closed-form formulas derived in chapter 2, while a numerical method is applied in the function for updating emission parameters. The derivatives of Q-function with respect to emission parameters derived in chapter 2 are used in the numerical method to increase the efficiency.

Finally, we write a main function to perform EM algorithm by integrating all previous parts (see Function 7 in appendix A).

3.1.2 Inference

Once we get the parameter estimation done, we can use it to perform inference. We apply the Viterbi algorithm to find the locations of translated uORFs and main coding regions in mRNAs (see Algorithm 8 in appendix A).

The Viterbi algorithm would find the most likely sequence of hidden states. Hence, we would have the underlying state for each base on mRNA. If we only want the locations of start and end positions of the translated uORF and main coding region, we can extract the indices of the corresponding states from the whole predicted sequence.

3.2 Data sets

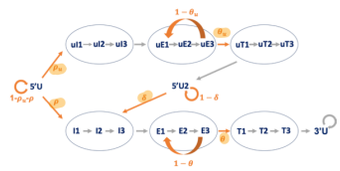
For the R package users to explore ribosome footprint counts, we generate ribosome footprint count data and include it in the R package.

Based on the model we constructed, we can generate ribosome footprint count data. However, the data generated from this model would be slightly different from the real ribosome footprint count data.

From our model, the hidden Markov chain would stay at 5'U for sometime, $\frac{1}{\rho_u+\rho}$ on average. And then jump to translated uORF or main coding region with probabilities $\frac{\rho_u}{\rho_u+\rho}$ and $\frac{\rho}{\rho_u+\rho}$ respectively. So on average, the length of 5'U in hidden Markov chain is $\frac{1}{\rho_u+\rho}$ regardless uORF is translated or not. But in the real ribosome footprint count data, if the uORF is untranslated, the length of 5'U should be longer. Also, we have the situation where both uORF and main coding region are untranslated. The ribosome footprint counts should be at a low level with the hidden states all being 5'U. But in our model, this situation occurs rarely. So what's situation?

Sum not 1?

For the reasons above, in the R package, we include two data sets. One is generated from the underlying true model, while the other one is generated based on what the ribosome footprint counts actually look like.



3.2.1 Counts generated from model

To generate count data from model we constructed, we need to specify the parameters.

We set the transition probabilities $\rho_u = 0.04$, $\rho = 0.01$, and $\delta = 0.05$. We use ν being 10 for all states. The mean parameters m for untranslated regions are all 5. The mean parameters for translated uORF and main coding region are given in Table 3.1.

uORF	uI1	uI2	uI3	uE1	uE2	uE3	uT1	uT2	uT3
Mean m	45	30	15	40	20	10	45	30	15
main coding region	I1	I2	I3	E1	E2	E3	T1	T2	T3
Mean m	90	75	30	60	50	20	90	75	30

Table 3.1: Mean parameters m for generating ribosome footprint counts

We use 5 different values for the gene expression level E: 1, 5, 10, 15, 20. For each gene expression level, we generate 100 sequences of ribosome footprint counts. The corresponding mRNAs sequences and gene expression levels are saved together as a list in R.

One issue of the data generation above is that the transition probabilities θ_u and θ are functions of mRNA sequence, which we need to specify. Rather than define a mRNA sequence, we let $\theta_u = \frac{1}{12}$ and $\theta = \frac{1}{32}$, so that the lengths of elongation parts of translated uORf and main coding region are $33 = (12-1)\times 3$ and $93 = (32-1)\times 3$ respectively. Then we insert stop codons at the corresponding mRNA sequence.

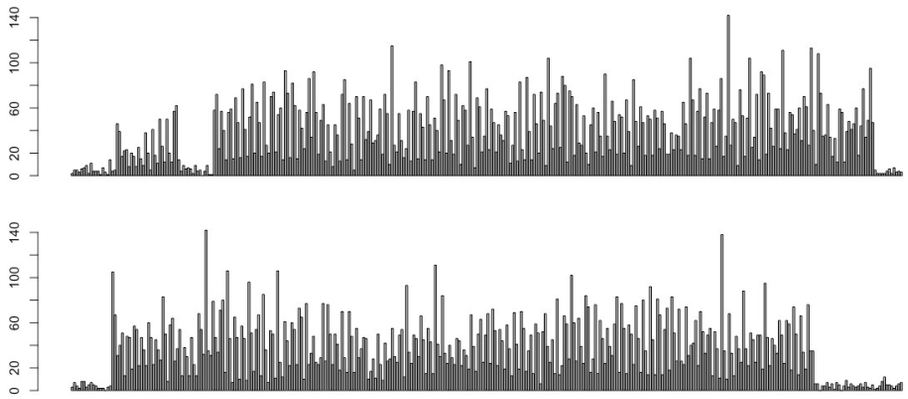


Figure 3.1: 2 types of ribosome counts generated from riboHMM2 model.

A plot of 2 types of ribosome counts is shown in Figure 3.1. The sub-figure on the top is the case when both uORF and main coding region are translated. The sub-figure on the bottom is the case when only the main coding region is translated. As we see from the figure, the length of front end untranslated regions of both sequences are roughly the same. While in the real situation, the front end untranslated regions would be longer if only the main coding region is translated.

3.2.2 Counts generated based on what real ribosome footprints look like

To mimic the real ribosome footprint counts, we modify the underlying Markov chain so that it will go through uORF and main coding region in order with probability 1. And the emission distribution is changed as well. When the hidden Markov chain is in uORF, there is a chance that the emission distribution is switched to the emission distribution of untranslated region. Hence, the count numbers in the untranslated uORF would be similar to the count numbers in region with state being 5'U. And the lengths of front end untranslated regions would be similar regardless the uORF is translated or not.

To be more specific, we use the same parameters for mean m and shape ν as above. While the transition probability ρ is set to be 0. And define new parameters p_1 and p_2 as the probabilities of uORF and main coding region being translated.

Again, we use 5 different values for the gene expression level E: 1, 5, 10, 15, 20. This time, for each gene expression level, we generate 5,000 sequences of ribosome footprint counts. The corresponding mRNAs sequences and gene expression levels are saved together as a list in R.

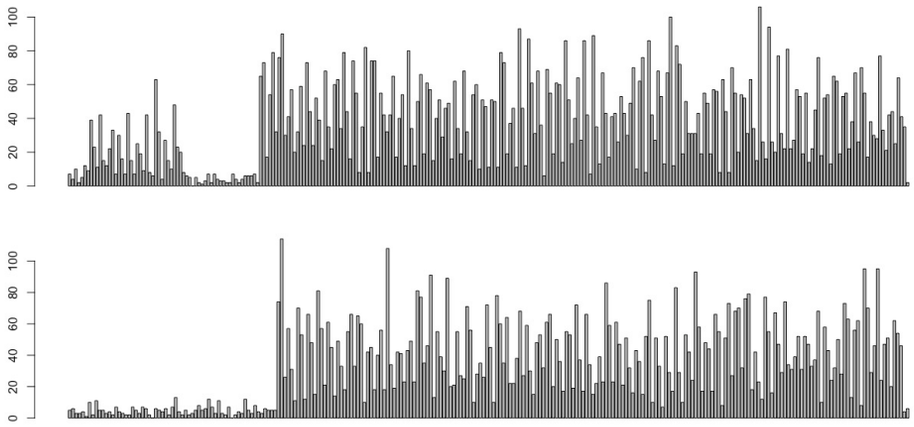


Figure 3.2: 2 types of ribosome counts generated based on what real ribosome footprints look like.

Similar to previous section, we plot 2 types of ribosome counts. In Figure 3.2, the sub-figure on the top is a ribosome footprint count sequence with both uORF and main coding region being translated. On the bottom, we have a count sequence that only main coding region is translated. From the figure, we see that the length of front end untranslated region is longer if only the main coding region is translated. And the starting position of main coding region is roughly the same.

3.3 Vignette

In this section, we explain how to use the R package. Two main functions are demonstrated, `estimate` and `optimalPath`.

For demonstration purpose, we only select 50 transcripts, 10 for each gene expression level, from the data set generated in 3.2.1. As the number of transcripts used increases, the running time would increase. Then, instead of using a personal com-

puter, users might want to submit the running job in a computer cluster and wait for the result.

Create vector for indices of sequences we want to use.

```
k <- 10
ii <- c()
for (i in 1:5){
  ii <- c(ii, ((i-1)*100+1):((i-1)*100+k))
}
print(ii)
```

```
 [1]   1   2   3   4   5   6   7   8   9  10
[11] 101 102 103 104 105 106 107 108 109 110
[21] 201 202 203 204 205 206 207 208 209 210
[31] 301 302 303 304 305 306 307 308 309 310
[41] 401 402 403 404 405 406 407 408 409 410
```

Then, we extract sequences of ribosome footprint counts, mRNAs and gene expression levels from first data set, which is stored in `uORF` as a list.

```
df <- uORF[ii]
X <- RNA <- list(); E <- c()
for (i in 1:length(df)){
  X[[i]] <-  df[[i]]$x
  RNA[[i]] <- df[[i]]$RNA
  E[i] <- df[[i]]$E
}
```

Before we estimate the parameters in our model, we need one more step. We need to decide initial values of parameters. Here we use a function `init_generate` to make an initial guess. Users can try different initial values.

```
init <- init_generate(df, cutoff=10, r=4)
```

Now we have input data and initial values ready. Run the main function `estimate` as follow.

```
res <- estimate(X, RNA, E, init, tol=0.001, detail=TRUE)
```

The option detail=TRUE is used to print out the estimation process. If users do not need to monitor the estimation process, the option should be set to FALSE. The output printed for estimation process is given below:

```
--------------------------------------------------
current time:  2019-05-08 04:35:08
Initial transition: 0.04516372 0.01129093 0.05645465
Initial v:
 3.337 3.337 3.337 3.337 3.337 3.337 3.337 3.337 3.337
 3.337 3.337 3.337 3.337 3.337 3.337 3.337 3.337 3.337
Initial m:
 55.329 41.497 13.832 55.329 41.497 13.832 55.329 41.497 13.832
 69.162 41.497 20.749 69.162 41.497 20.749 69.162 41.497 20.749

--------------------------------------------------
-------------------------- Entering iteration:  1

            . . . . . . . . . . . .
        (part of result is omitted here)
            . . . . . . . . . . . .

------------------------------------- Iteration 3 took: 0.1 mins
------ New Q = -48590.192, Old Q = -48590.192, increased by 0

*****************************************************************
Tolerance satisfied. Optimization finished
current time:  2019-05-08 04:36:16
------------------------------------- Total time used: 1.1 mins
*****************************************************************
```

The estimated mean parameters m are the key features of ribosome footprints. They can be accessed directly from the estimation result by calling `res`. (We assigned the result in `res` in previous code.)

```
print(res)

Mean parameters of translated uORF:
  uI1  45.1         uE1 39.57         uT1 48.47
  uI2  31.1         uE2 19.99         uT2 33.61
  uI3  16.5         uE3 10.31         uT3 16.07

Mean parameters of main coding region:
   I1 93.84          E1 60.48          T1 92.65
   I2 71.62          E2 50.28          T2 74.74
   I3 27.77          E3 20.27          T3 29.23

Mean parameters of untranslated region:
   5'U 4.88      5'U2 5.07       3'U 4.95

Number of iterations:
                      3
```

All estimated parameters and their updating histories can be called by

```
v_hat <- res$v_hat
m_hat <- res$m_hat
trans_hat <- res$trans_hat
res$v_history
res$m_histroy
res$trans_history
```

Next we use the function `optimalPath` to infer the underlying hidden states. For example, for the 5th transcript, the optimal path is obtained by

```
i <- 5
optPth <- optimalPath(df[[i]]$x, df[[i]]$RNA, E[i],
                      v_hat, m_hat, trans_hat)$optPth
```

The whole hidden Markov chain is stored in `optPth`. To find the start and end positions of the translated uORF, simply type:

```
start_uORF <- which(optPth == 2)
end_uORF <- which(optPth == 10)
c(start_uORF, end_uORF)
[1] 20  46
```

Similarly, for the start and end positions of the main coding region, simply type:

```
start_main <- which(optPth == 12)
end_main <- which(optPth == 20)
c(start_main, end_main)
[1] 61  339
```

Chapter 4

Simulation studies

In this chapter, we conduct three simulation studies. At each case, we check that our model has the ability to identify the locations of translated uORFs and main coding regions in mRNAs.

In the first case, we use the data set generated from our model. For the second case, we use the data set generated based on what real ribosome footprint counts look like. In the last case, we add noise to mRNA sequences in previous data set.

We present the descriptions of simulations and the results below. All simulation code is included in Appendix B.

4.1 Case 1: use counts generated from our model

In this case, we use the ribosome footprint count data generated from our model. The result shows that our model can correctly identify the translated uORFs and main coding regions of mRNAs.

The ribosome footprint count data generated from model is saved as a list, named `uORF` in the R package *riboHMM2*. The description of this data set is in the previous chapter (see section 3.2.1).

We use ribosome count data over 500 mRNA transcripts to estimate the parameters. The estimated parameters are close to the true parameters, which indicates we are

able to capture the structure of ribosome footprint counts. (We do not check how accurate the parameter estimation is, since our final goal is to identify the locations of translated uORFs and main coding regions.) Then we apply the Viterbi algorithm to find the full paths of hidden Markov states of mRNA transcript. The inferred locations of translated uORFs and main coding regions are extracted, and checked against the true locations. The result shows that all locations of translated uORFs and main coding regions of 500 mRNAs are correctly identified.

We repeat the above simulation with multiple of data sets generated in the same way, but with different seeds. Same results yield, which indicates that our model has the ability to identify the translated uORFs and main coding regions within mRNAs, in the case of using counts generated from our model.

4.2 Case 2: use counts generated based on what real ribosome footprints look like

As we mentioned in previous chapter, the count data generated from our model is a bit different from the real ribosome footprint count data. We should check that our model can identify the translated regions with real ribosome footprint count data. Unfortunately, we are still in the stage of pre-processing the raw data of ribosome footprints. Hence, we generate data based on what real ribosome footprint counts look like, and use it to pre-check that our model works with real ribosome footprint count data.

The ribosome footprint count data generated based on what real ribosome footprints look like is saved as a list, nameed `ribo` in the R packge *riboHMM2*. The description of this data set is also in the previous chapter (see section 3.2.2).

We repeat the same process as in case 1, yielding a similar result. The estimated parameters are close to the true parameters, except for the transition probabilities, as there is no true probability in this case. For the locations of translated uORFs and main coding regions, we also identify them correctly in all 500 mRNA transcripts.

Again, multiple of simulations have been done with data generated using different seeds.

4.3 Case 3: use counts with noise in mRNA sequences

In our model, transition probabilities of the underlying Markov chain depend on the mRNA sequence as the termination of translation depends on canonical stop codons, {UAG, UAA, UGA }. In previous simulations, we only put 2 stop codons in the mRNA sequence, one for translated uORF and the other for main coding region. However, in a real situation, we might have more stop codons in the mRNA sequence, which might affect the parameter estimation and inference on the locations of translated regions. Hence, we should check how the noise in the mRNA sequence would change the performance of our model.

We use count data based on what real ribosome footprint counts look like as in case 2 of the previous section. But this time, we add noise in the mRNA sequences by adding more stop codons. We repeat the analysis as above, yielding a similar result. All locations of translated uORFs and main coding regions in 500 mRNAs are correctly identified.

Again, multiple of simulations have been done with data generated using different seeds.

Hence, we conclude that the effect of ribosome counts dominates the effect of stop codons in mRNA sequence, and the model is robust to noise in the mRNA sequence.

Chapter 5

Future study

So far, we developed a method which uses ribosome footprint data for more comprehensive identification of translated uORFs and main coding regions within mRNAs. Futhermore, we performed three simulation studies to demonstrate that the model has the potential to identify the locations of the translated uORFs and main coding regions within mRNAs.

5.1 Real data analysis

The method works on simulated data. However, the real ribosome footprint count data is much noisier than the simulated data. Currently, a PhD student, Yupei You, is pre-processing the raw ribosome footprint count data. The raw data is in fastq format, which needs to be converted into bam file format, and further into sequences of counts.

Once we have sequences of ribosome footprint counts, we could apply our model on this data to identify the locations of desired regions within mRNA. Then we would validate the results against the results from Raj et al. in 2016 [12]. Furthermore, our collaborator Dr. Wang at UT Health in US will check that, the regions of translated uORFs that we proposed do synthesize proteins.

5.2 Model modification

Our current model allows translation to start at non-canonical start codons, but we assume all codons have the same transition probabilities ρ_u, ρ, and δ. We will allow each codon (or groups of codons) to have codon-specific transition probabilities by modelling the probabilities as functions of mRNA sequence. This extension potentially improves the performance our method, and provides new biological understanding on regulatory mechanisms on translation.

Appendix A

Algorithms and functions used in R package

Algorithm 1: Forward algorithm

Result: Forward probability $\log F_k(t)$, saved in matrix logF

Initializtion:

logF := matrix of size M by n with all entries being $-\infty$

$\text{logF}[1,1] \leftarrow \log h_k(x_1)$

for $t = 2, \cdots, n$ **do**

  for $k = 1, \cdots, M$ **do**

    $\text{logF}[k,t] \leftarrow \log \left\{ \sum_j \exp\left[\ \text{logF}[j,t-1] + \log h_k(x_t) + \log a_{jk}(t-1)\ \right] \right\}$

  end

end

Algorithm 2: Backward algorithm

Result: Backward probability $\log B_k(t)$, saved in matrix logB

Initializtion:

logB := matrix of size M by n with all entries being 0

for $t = n-1, \cdots, 1$ **do**

  for $k = 1, \cdots, M$ **do**

    $\text{logB}[k,t] \leftarrow \log \left\{ \sum_j \exp[\ \text{logB}[j,t+1] + \log h_j(x_{t+1}) + \log a_{kj}(t)\ ] \right\}$

  end

end

Function 3: Compute $L_k(t)$

Input: logF, logB

Result: $L_k(t)$, save in matrix L

L $\leftarrow \exp(\text{logF} + \text{logB} - \log \sum_j \exp(\text{logF}[j, n]))$

Function 4: Compute $H_{kj}(t)$

Input: logF, logB

Result: $H_{kj}(t)$, save in matrix H

Initializtion:

H := empty matrix of size $n-1$ by 3.

kk := $[1, 1, 11]$ and jj := $[2, 12, 12]$

for $t = 1, \cdots, n-1$ **do**

  for $i = 1, 2, 3$ **do**

    numer $\leftarrow \text{logF}[t, kk[i]] + \text{logB}[t+1, jj[i]] + \log a_{kk[i],jj[i]}(t) + h_{jj[i]}(x_{t+1})$

    denom $\leftarrow \log \sum_j \exp)(\text{logF}[j, n])$

    H[t,i] $\leftarrow \exp(\text{numer} - \text{denom})$

  end

end

Function 5: Update transition probability ρ_u, ρ, δ

Input: H

Result: ρ_u, ρ, δ

Using the closed-form formula derived in chapter 2
to update the transition probability

Function 6: Update emission parameters ν, m

Input: L

Result: ρ_u, ρ, δ

Using numerical method with derivatives provided
to update the emission parameters

Function 7: Parameter estimation

Input: ribosome footprint counts, mRNAs, E, initial $\theta^{(0)}$, threshold
Result: Parameters θ

Initializtion:
4 empty lists for logF, logB, L, H
m := number of mRNAs in the input

```
while not convergent do
    /* Calculated the terms needed in likelihood */
    for i = 1, ··· , m do
        Compute logF, logB, L and H with i-th mRNA transcript
        Save them into i-th elements of corresponding lists.
    end

    /* Update parameters */
    for k = 1, ··· , M do
        Use L calculated above in the formula derived from chapter 2
            to update emission parameters ν, m
        Use L and H calculated above in the formula derived from chapter 2
            to update transition probability ρ_u, ρ, δ
    end

    /* Check convergence */
    if the value of Q-function increases > threshold then
        Enter next iteration
    else
        Break the while loop
    end
end
```

Algorithm 8: Viterbi algorithm

Input: a sequence of ribosome footprint, the estimated parameters
Result: the most likely sequence of hidden states

Initializtion:

G := a length M vector for log-likelihood of 4 sub-paths
S := a $n-1$ by M matrix for storing sequence of sub-paths

```
/* paths start with state 1.  So first entry is 0, other entries are −∞      */
G := (0, −∞, ··· , −∞)

/* increase the length of sub-paths                                          */
for t = 2, ··· , n do
    /* consider sub-path ends with state k                                   */
    G_temp := [0, ··· , 0] ;                                        // length M
    for k = 1, ··· , M do
        /* likelihoods of all previous sub-paths chained with state k        */
        Gk_temp ← G + log a[, k] + h_k(x_t)
        /* choose the one with largest likelihood                            */
        G_temp[k] ← max(Gk_temp)
        /* record this optimal previous sub-path by saving the last state    */
        S[t-1,k] ← index which maximises Gk_temp.
    end
end

/* backtrack to find optimal previous state at a time                        */
s ← index which maximises G
for t = n, ··· , 2 do
    s ← (S[t-1, s[1]], s)
end
```

Appendix B

R code for simulation studies

Simulation studies: case 1

```
library(riboHMM2)

# extract sequence of counts, mRNAs and gene expressions
X <- RNA <- list(); E <- c()
for (i in 1:length(uORF)){
  X[[i]] <- uORF[[i]]$x
  RNA[[i]] <- uORF[[i]]$RNA
  E[i] <- uORF[[i]]$E
}

# estimate parameters
init <- init_generate(uORF, cutoff=10, r=4)
res <- estimate(X, RNA, E, init, tol=0.001, detail=FALSE)
v_hat <- res$v_hat
m_hat <- res$m_hat
trans_hat <- res$trans_hat

# identify the locations
location_uORF <- matrix(0, length(uORF), 2)
location_main <- matrix(0, length(uORF), 2)
```

```r
for (i in 1:length(uORF)){
  optPth <- optimalPath(uORF[[i]]$x, uORF[[i]]$RNA, E[i],
                        v_hat, m_hat, trans_hat)$optPth
  if (sum(optPth==2)){
    location_uORF[i,] <- c(which(optPth==2), which(optPth==10))
  }
  if (sum(optPth==12)){
    location_main[i,] <- c(which(optPth==12), which(optPth==20))
  }
}

# check against with true locations
num_err_uORF <- 0
num_err_main <- 0
for (i in 1:length(uORF)){
  if (location_uORF[i,1] == 0){
    num_err_uORF <- num_err_uORF + (2 %in% Z[[i]])
  }else{
    num_err_uORF <- num_err_uORF +
      sum(Z[[i]][location_uORF[i,]] != c(2,10))
  }
  if (location_main[i,1]==0){
    num_err_main <- num_err_main + (12 %in% Z[[i]])
  }else{
    num_err_main <- num_err_main +
      sum(Z[[i]][location_main[i,]] != c(12,20))
  }
}
print(c(num_err_uORF, num_err_main))

[1] 0  0
```

Simulation studies: case 2

```
library(riboHMM2)

num_E <- length(ribo)/5  # number of chains for each E

# extract 100 transcripts from each gene level
k <- 100
ii <- c()
for (i in 1:5){
  ii <- c(ii, ((i-1)*num_E+1):((i-1)*num_E+k))
}
ribo <- ribo[ii]

# extract sequence of counts, mRNAs and gene expressions
X <- RNA <- Z <- list(); E <- c()
for (i in 1:length(ribo)){
  X[[i]] <- ribo[[i]]$x
  RNA[[i]] <- ribo[[i]]$RNA
  E[i] <- ribo[[i]]$E
  Z[[i]] <- ribo[[i]]$z
}

# estimate model parameters
init <- init_generate(ribo, cutoff=10, r=4)
res <- estimate(X, RNA, E, init, tol=0.001, detail=TRUE)
v_hat <- res$v_hat
m_hat <- res$m_hat
trans_hat <- res$trans_hat


# identify the locations
location_uORF <- matrix(0, length(ribo), 2)
location_main <- matrix(0, length(ribo), 2)
for (i in 1:length(ribo)){
```

```r
  optPth <- optimalPath(ribo[[i]]$x, ribo[[i]]$RNA, E[i],
                        v_hat, m_hat, trans_hat)$optPth
  if (2 %in% optPth){
    location_uORF[i,] <- c(which(optPth==2), which(optPth==10))
  }
  if (12 %in% optPth){
    location_main[i,] <- c(which(optPth==12), which(optPth==20))
  }
}

# check against with true locations
num_err_uORF <- 0
num_err_main <- 0
for (i in 1:length(ribo)){
  if (location_uORF[i,1] == 0){
    num_err_uORF <- num_err_uORF + (2 %in% Z[[i]])
  }else{
    num_err_uORF <- num_err_uORF +
      sum(Z[[i]][location_uORF[i,]] != c(2,10))
  }
  if (location_main[i,1]==0){
    num_err_main <- num_err_main + (12 %in% Z[[i]])
  }else{
    num_err_main <- num_err_main +
      sum(Z[[i]][location_main[i,]] != c(12,20))
  }
}
print(c(num_err_uORF, num_err_main))

[1] 0  0
```

Simulation studies: case 3

```
library(riboHMM2)
num_E <- length(ribo)/5  # number of chains for each E

# extract 100 transcripts from each gene level
k <- 100
ii <- c()
for (i in 1:5){
  ii <- c(ii, ((i-1)*num_E+1):((i-1)*num_E+k))
}
ribo <- ribo[ii]

# add noise in mRNA sequences
for (i in 1:length(ribo)){
  i_one <- which(ribo[[i]]$RNA==1)
  ribo[[i]]$RNA[i_one[1] + c(3, -4, -8)] <- 1
  ribo[[i]]$RNA[i_one[2] + c(3, -4, -8)] <- 1
}

# extract sequence of counts, mRNAs and gene expressions
X <- RNA <- Z <- list(); E <- c()
for (i in 1:length(ribo)){
  X[[i]] <- ribo[[i]]$x
  RNA[[i]] <- ribo[[i]]$RNA
  E[i] <- ribo[[i]]$E
  Z[[i]] <- ribo[[i]]$z
}

# estimate model parameters
init <- init_generate(ribo, cutoff=10, r=4)
res <- estimate(X, RNA, E, init, tol=0.001, detail=TRUE)
v_hat <- res$v_hat
m_hat <- res$m_hat
trans_hat <- res$trans_hat
```

```r
# identify the locations
location_uORF <- matrix(0, length(ribo), 2)
location_main <- matrix(0, length(ribo), 2)
for (i in 1:length(ribo)){
  optPth <- optimalPath(ribo[[i]]$x, ribo[[i]]$RNA, E[i],
                        v_hat, m_hat, trans_hat)$optPth
  if (2 %in% optPth){
    location_uORF[i,] <- c(which(optPth==2), which(optPth==10))
  }
  if (12 %in% optPth){
    location_main[i,] <- c(which(optPth==12), which(optPth==20))
  }
}

# check against with true locations
num_err_uORF <- 0
num_err_main <- 0
for (i in 1:length(ribo)){
  if (location_uORF[i,1] == 0){
    num_err_uORF <- num_err_uORF + (2 %in% Z[[i]])
  }else{
    num_err_uORF <- num_err_uORF +
      sum(Z[[i]][location_uORF[i,]] != c(2,10))
  }
  if (location_main[i,1]==0){
    num_err_main <- num_err_main + (12 %in% Z[[i]])
  }else{
    num_err_main <- num_err_main +
      sum(Z[[i]][location_main[i,]] != c(12,20))
  }
}
print(c(num_err_uORF, num_err_main))
```

```
[1] 0  0
```

Bibliography

[1] Francis Crick. Central dogma of molecular biology. *Nature*, 227(5258):561, 1970.

[2] Cristina Barbosa, Isabel Peixeiro, and Luisa Romao. Gene expression regulation by upstream open reading frames and human disease. *PLoS genetics*, 9(8):e1003529, 2013.

[3] David R Morris and Adam P Geballe. Upstream open reading frames as regulators of mrna translation. *Molecular and cellular biology*, 20(23):8635–8642, 2000.

[4] Sarah E Calvo, David J Pagliarini, and Vamsi K Mootha. Upstream open reading frames cause widespread reduction of protein expression and are polymorphic among humans. *Proceedings of the National Academy of Sciences*, 106(18):7507–7512, 2009.

[5] Tania Arcondéguy, Eric Lacazette, Stefania Millevoi, Herve Prats, and Christian Touriol. 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):7997–8010, 2013.

[6] Shelley R Starck, Jordan C Tsai, Keling Chen, Michael Shodiya, Lei Wang, Kinnosuke Yahiro, Manuela Martins-Green, Nilabh Shastri, and Peter Walter. Translation from the 5' untranslated region shapes the integrated stress response. *Science*, 351(6272):aad3867, 2016.

[7] Mitchell Guttman, Pamela Russell, Nicholas T Ingolia, Jonathan S Weissman, and Eric S Lander. Ribosome profiling provides evidence that large noncoding rnas do not encode proteins. *Cell*, 154(1):240–251, 2013.

[8] Jennifer Harrow, Adam Frankish, Jose M Gonzalez, Electra Tapanari, Mark Diekhans, Felix Kokocinski, Bronwen L Aken, Daniel Barrell, Amonida Zadissa, Stephen Searle, et al. Gencode: the reference human genome annotation for the encode project. *Genome research*, 22(9):1760–1774, 2012.

[9] Laurens G Wilming, James GR Gilbert, Kerstin Howe, S Trevanion, T Hubbard, and Jennifer L Harrow. The vertebrate genome annotation (vega) database. *Nucleic acids research*, 36(suppl_1):D753–D760, 2007.

[10] Nicholas T Ingolia, Sina Ghaemmaghami, John RS Newman, and Jonathan S Weissman. Genome-wide analysis in vivo of translation with nucleotide resolution using ribosome profiling. *science*, 324(5924):218–223, 2009.

[11] Nicholas T. Ingolia. Ribosome profiling: new views of translation, from single codons to genome scale. *Nature Reviews Genetics*, 15:205 EP –, 01 2014.

[12] Anil Raj, Sidney H Wang, Heejung Shim, Arbel Harpak, Yang I Li, Brett Engelmann, Matthew Stephens, Yoav Gilad, and Jonathan K Pritchard. Thousands of novel translated open reading frames in humans inferred by ribosome footprint profiling. *Elife*, 5:e13328, 2016.

[13] David E Weinberg, Premal Shah, Stephen W Eichhorn, Jeffrey A Hussmann, Joshua B Plotkin, and David P Bartel. Improved ribosome-footprint and mrna measurements provide insights into dynamics and regulation of yeast translation. *Cell reports*, 14(7):1787–1799, 2016.

[14] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[15] Lawrence R Rabiner and Biing-Hwang Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986.

[16] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[17] G David Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.