Chapter 4. Modelling functional elements associated with ncRNAs

So far in thesis, the main focus has been on discussing issues related to applying comparative-genomics based approaches for genome-wide ncRNA finding. This is due to the fact that till now these approaches have been believed to be one of the most promising ncRNA finding strategies. With the evidence presented in the previous chapters, this belief has therefore been challenged, due to the finding of insufficient covariations, the existence of numerous synteny-non-conserved and potentially functional ncRNAs, *etc*. There is another related limitation of alignment approaches to this general problem: if a set of functionally related ncRNAs are mainly constrained at the structural level, their sequences may become very divergent at the primary-sequence level, making alignment very difficult (Torarinsson et al. 2006).

Accordingly, it is appropriate to consider what approaches might be viable for genome-wide ncRNA finding which do not rely on comparative genomics. One possible strategy is to apply machine learning techniques which can, given a set of unaligned functional ncRNAs, generate models of functional elements implicated in either the transcription or functioning of ncRNAs. Such models can then be used to scan the genomes in order to find novel ncRNAs.

From this chapter, I consider the computational modelling of two types of functional elements that may be associated with ncRNAs:

- the transcription start sites (TSSs) of ncRNAs
- the functional elements/sites that are associated with RNA motifs in RNA transcripts

In the first part of this chapter, I introduce the computational approaches that may be used to find the transcription regulatory regions, including enhancers/silencers and transcription start sites (TSSs). I start with a brief introduction of transcription regulatory regions, as well as the basics of available motif models and relevant machine learning techniques that have been used to discover motifs. Then I introduce an existing system, Eponine, which was designed to generate predictive models of functional sites, such as TSSs, in genomes.

In the second part of this chapter, I consider the direct detection of RNA motifs in genomes. I explore the possibility of applying available computational approaches for identifying RNA structural motifs in genomes. I also introduce a new model I have created for the purpose of discovering the functional sites which are associated with RNA structural motifs.

4.1. Computational detection of transcription regulatory regions

Access to and recognition of transcription units by transcription machinery are two critical steps in the generation of functional transcripts of all genes, including both protein-coding and ncRNA genes. The essential components involved in transcription initiation include RNA polymerases, transcription factors (TFs), DNA templates, and transcription regulatory elements on genomic DNA sequences. The regulatory elements that are on the same chromosome as the respective transcription units are also called *cis*-regulatory elements. Based on the distance from the genes they regulate, *cis*-regulatory elements can be further categorized into promoters, which are in close proximity to transcription start sites (TSSs), and enhancers/silencers, which can be at great distance from TSSs. A regulatory element may consist of multiple transcription factor binding sites (TFBSs) that can specifically interact with different TFs. A set of TFBSs for a particular TF may share unique sequence patterns, which are generally short and degenerate.

For each gene, the interaction of its promoter with a specific type of RNA polymerase and with a set of TFs determines the exact transcription start point. Different RNA polymerases together with specific sets of TFs favour different promoter sequences. In eukaryotes, there are three different types of RNA polymerases for transcribing genes into RNA molecules. RNA polymerase I only transcribes tandemly repeated ribosomal RNA genes (except 5S rRNA genes). RNA polymerase III transcribes tRNA genes, 5S rRNA genes, and some small nuclear RNA genes. RNA polymerase II transcribes all protein-coding genes. There is evidence indicating that RNA polymerase II is also responsible for transcribing many structural ncRNA and mRNA-like ncRNA genes (Lee et al. 2004). Genes that are transcribed by RNA polymerase I are referred to as pol I genes, and so forth. Modelling promoters of pol II or pol III genes is therefore potentially useful for ncRNA finding. In fact, the internal promoters of tRNA genes have been used as an important signal for tRNA finding in eukaryotic genomes (Fichant and Burks 1991; Pavesi et al. 1994; Lowe and Eddy 1997).

Enhancers/silencers are another type of transcription regulatory element. Their function may be independent of their orientations and distances relative to respective transcription start sites (For review see Khoury and Gruss 1983). Interaction of enhancers/silencers with transcription factors can alter the transcription efficiency of associated transcription units. One important regulatory mechanism of enhancers is inducing chromatin remodelling in eukaryotic cells (For reviews see Vignali et al. 2000; Berger 2002). The genomic DNA of eukaryotes is packaged with histone and non-histone proteins into compact chromatin. To allow transcription to be initiated, the structure of compact chromatin must be remodelled in order to allow efficient access by RNA polymerases. In particular, a class of complex enhancers, locus control regions (LCRs), may consist of multiple regions for initiating chromatin remodelling (For review see Dean 2006). While an enhancer can regulate transcription of only one gene, LCRs can be effective on a cluster of genes. For example, an LCR in mammalian genomes is suggested to regulate the temporal expression of the beta-globin locus, which consists of at least four genes (For review see Li et al. 2002).

Many computational methods have been developed in order to address the problems relevant to finding transcription regulatory regions in genomes. For instance, many motif finders have been developed to detect over-represented motifs. However, the over-represented motifs so discovered may not directly be useful for discriminating functional sites in genomes. One reason is that the individual interaction between a TF and its TFBS is rarely sufficient to trigger a particular regulatory mechanism. For instance, in eukaryotes, the transcription initiation may be associated with multiple TFBSs (for review see Sandelin et al. 2007). Consequently, for the purpose of finding particular functional sites in genomes, I consider the systems which can model the association of multiple TFBSs with particular functional sites.

In the following two subsections, I introduce the approaches for finding motifs and functional sites. In the first subsection (4.1.1.), existing computational approaches for discovering over-represented motifs are briefly introduced. Although these approaches were not directly used in the work presented in this thesis, this introduction provides essential knowledge for using methods that can perform selective classification of functional sites in the genomes. In the second subsection (4.1.2.), I introduce the computational approaches that can be used to model particular functional sites, such as TSSs and TTSs in genomes. The approaches described and developed here are applied in chapters 5 and 6.

4.1.1. Computational detection of over-represented motifs

Computational detection of over-represented motifs in a set of related sequences can be helpful when studying the regulatory mechanisms of gene expression. Although determination of the functional TFBSs for a TF in genomes can currently only be achieved by experiment, many computational systems have been designed for the purpose of finding over-represented patterns in a set of sequences containing genes known to be regulated by a particular TF. If over-represented motifs can distinguish sequences with genes with similar functions from background genomic sequences, these features can be suspected to be candidate regulatory elements, possibly TFBSs of the same TF(s).

Over the past decades, many computational approaches have been developed in order to find the over-represented motifs among a set of related sequences. There are two main issues in discovering motifs: 1) the type of model used to represent motifs; 2) the approach used to learn the parameters of the motif model. In the following of this section, these two issues are discussed.

4.1.1.1. Motif models

The first step towards modelling transcription regulatory regions is using a formulation to describe a set of TFBSs for a particular TF. There are at least two types of motif models that have been used for this purpose: *consensus* based models, and *profile* based models.

4.1.1.1.1. Consensus based models

A consensus is a string of simple symbols for describing the most probable nucleotide at each position of TFBSs. A consensus model is suitable for describing a set of TFBSs that are completely identical. Consensus based models have also been extended to incorporate ambiguous symbols. One strategy is to use the IUPAC-IUB alphabet (Nomenclature Committee of the International Union of Biochemistry 1986) to code the ambiguous symbols (Tompa 1999). For example, if both A and G are observed at a particular position of a set of TFBSs, "R" (purine) is thus used to represent this position; if all four types of nucleotides are observed, then "N" is used.

The significance of a consensus can be evaluated by several different scoring schemes. One widely used scoring scheme is the *z*-score, which measures how unlikely a consensus with certain occurrences in a given set of sequences is found given a background distribution (Tompa 1999). In brief, the *z*-score is the number of standard deviations of the observed frequency of a consensus from its expected frequency. The expected frequency of a consensus can be calculated by counting the number of occurrence in a set of random sequences, which can be generated using a high-order Markov chain modelling the background distribution (Sinha and Tompa 2002).

4.1.1.1.2. Profile based models

One problem with the consensus based motif model is its insufficiency for describing the differential preference toward different symbols at a particular position of a motif. A more flexible, and possibly more powerful, motif model is a *profile* based model, which can describe the alignment of a set of functionally related TFBSs. A widely used profile based model for representing motifs is a position frequency matrix (PFM) (also as position specific frequency matrix, PSFM) (for review see Wasserman and Sandelin 2004), which is a type of product-multinomial model. A PFM consists of a series of columns. Each column of a PFM is a multinomial distribution over all possible symbols of the alphabet used in each position of a motif. By using a PFM, each position of a sequence motif is treated independently, although this assumption may be biologically imprecise as shown in some analyses of protein-DNA interactions (Barash et al. 2003).

The probability of emitting a particular sequence pattern that starts at the i^{th} position of a sequence x from a PFM can be evaluated by:

$$M(x,i) = \prod_{l=1}^{|M|} P_l(x(i+l-1)) \qquad [4\text{-}1]$$

$|M|$ is the number of columns of the PFM. P_l returns the probability of a particular symbol emitted by the l^{th} column of the model. $x(i + l - 1)$ is the symbol at the $(i + l - 1)^{\text{th}}$ position of x. For modelling TFBSs, the possible symbols for each column consist of adenine (A), guanine

(G), cytosine (C), and thymine (T). A PFM can be displayed in the form of sequence logos (Schneider and Stephens 1990). A sequence logo for a PFM contains of a series of columns of stacked symbols, where the height of each symbol is proportional to its information content at each position. In the rest of this thesis, sequence logos are used to represent the primary-sequence motifs.

One advantage of using PFMs to describe motifs is that it is very easy to connect a motif model to statistical information theory. The statistical significance of a motif can be assessed by calculating the information content of a PFM. The information content at the l^{th} position of a site is:

$$I(l) = \sum_b P_{l,b} \log_2 \frac{P_{l,b}}{P_b} \qquad [4\text{-}2]$$

, where b refers to each of the possible bases; $P_{l,b}$ is the probability of base b at the l^{th} position; P_b is the frequency of base b in the background sequences (*e.g.* non-site sequences in the genomes). This formulation is equivalent to the relative entropy and the Kullback-Leibler distance, between the foreground motif model and the background sequence model (for review see Stormo 2000). Usually the base composition in the background sequence model is assumed to be independent and identically distributed (i.i.d.). One simple approach is to assume that each base in the background is equally probable and thus P_b is 0.25 for each base.

In order to search for a particular pattern in a given sequence, a PFM value is usually converted into a sum of a series of log-likelihood ratios with respect to a background sequence model *B*:

$$W(x,i) = \sum_{l=1}^{|M|} \log_2 \frac{P_l(x(i+l-1))}{B(x(i+l-1))} \qquad [4\text{-}3]$$

This conversion gives a position specific scoring matrix (PSSM), which is also called a position weight matrix (PWM) (for review see Wasserman and Sandelin 2004). Given a sequence region, a PWM can be used to evaluate the log-likelihood ratio between the foreground motif model and the background sequence model. A higher log-likelihood ratio can be interpreted as that the foreground model is more likely to generate a given sequence pattern than is the background model. The PWM scores have been shown to be proportional to the binding energy contribution of the bases (Berg and von Hippel 1987; Stormo 2000). A PWM can be used to scan for candidate TFBSs in a long sequence. For finding TFBSs in a sequence of length N, all N - $|M|$ + 1 sub-sequences of length $|M|$ must be enumerated and scored.

4.1.1.2. Algorithms for discovering motifs

In an *in silico* motif finding problem, the positions, patterns, and lengths of over-represented motifs in a set of related sequences may be initially unknown. Motif finding algorithms must be capable of optimizing these parameters given a set of sequences. In order to simplify the motif finding problem, existing motif finding algorithms usually require a user-defined motif length. Consequently, the parameters that need to be learned are the motif patterns, and their respective positions in individual sequences. Based on the models used, motif finding methods can be classified into *consensus* based and *profile* based methods, which are briefly introduced in the following, respectively.

4.1.1.2.1. Consensus based methods

Consensus based motif finding methods discover over-represented motifs by exhaustive enumeration of a set of motifs (Tompa 1999; Marsan and Sagot 2000; Pavesi et al. 2001). These methods usually use the following two steps to discover over-represented motifs:

- Enumerate all possible m-mer substrings in the given set of sequences.
- Score and rank the m-mer substrings by using some statistical measures, such as the z-score.

Consensus based methods can be very fast, if a suitable indexing structure, such as the suffix tree (Marsan and Sagot 2000), is used for organizing the sequences. While some evidence suggested that consensus based motif finding methods may suffer from high false positive rates (Osada et al. 2004), a recent survey reveals that these methods can have a performance comparable to that of profile-based methods (Tompa et al. 2005). However, there are considerations in using consensus based methods. Firstly, generating one consensus optimal for predicting new sites is not straightforward. Similar substrings must be clustered into fewer groups in a post-processing stage (Marsan and Sagot 2000). Secondly, for computational efficiency, some consensus based methods such as YMF (Sinha and Tompa 2000) and Weeder (Pavesi et al. 2001) restrict the number of mismatches allowed in a pattern. When several positions in a set of TFBSs with respect to a TF are weakly constrained, as in the cases of eukaryotes, consensus based methods may not work well (Pavesi et al. 2001).

4.1.1.2.2. Profile based methods

Profile based motif finding methods discover over-represented motifs by selecting oligonucleotides from the set of input sequences and then aligning them to generate profiles. These methods generally consist of two components:

- A likelihood function which can evaluate how likely a particular motif is to be over-represented given a set of sequences.
- An optimization procedure which can maximize the likelihood function.

A basic form of the likelihood functions used in many profile-based motif finding systems (for review see Stormo 2000) is the information content of a motif, as the formulation presented in [4-2]. The positions of a motif in individual sequences are referred to as the missing data. An important task of the optimization procedure is to search for the solution of missing data which may maximize the likelihood function. Two of the most widely used optimization algorithms are the Expectation Maximization (EM) (Lawrence and Reilly 1990; Bailey and Elkan 1994) and Gibbs Sampling (Lawrence et al. 1993).

EM algorithm

The EM algorithm is a general approach for maximizing a likelihood function with missing data. The EM algorithm iterates between two steps: in the first step, the expected values of the missing data are estimated, conditioned on the proposed model parameters; in the second step, given the expected values of the missing data, the new model parameters that can maximize the log likelihood function are chosen. The first step is the expectation step (E-step) and the second step is the maximization step (M-step). These two steps are iterated until a convergence criterion is satisfied.

There have been many extensions to the original EM based motif finding algorithm (Lawrence and Reilly 1990). For instance, the MEME (multiple expectation maximization for motif elicitation) algorithm is designed to model motifs with zero-or-one occurrences per sequence (ZOOPS) (Bailey and Elkan 1994), although the original EM motif finding algorithms were designed to find one occurrence per sequence. Another significant improvement to EM made in the MEME algorithm is its capability to detect multiple motifs within a single run.

Gibbs Sampling

In mathematics and physics, Gibbs Sampling is a sampling algorithm that is used to explore the joint probability of two or more random variables. It is a special case of the Metropolis-Hastings algorithm, which is a type of Markov chain Monte Carlo algorithm. A Gibbs Sampling approach for motif finding also consists of an iteration of two steps: predictive update step and sampling step (Lawrence et al. 1993), which correspond to the E-step and the M step of an EM algorithm respectively. However, unlike the deterministic process used in EM to find the missing data (*i.e.* the start sites of a motif in individual sequences), a stochastic process is adopted in the Gibbs Sampling motif finding algorithm (Lawrence et al. 1993). At the predictive update step of Gibbs Sampling, a sequence *z* is chosen and the other sequences are used to derive the model parameters, given the current site positions. At the sampling step, the probability of generating the site in each position of sequence *z* can thus be estimated conditioned on the current motif model. The new site position in sequence *z* is sampled with the probability distribution of the site positions.

Several improvements have been made to enhance the capability of the original Gibbs Sampling based motif finders (for review see Pavesi et al. 2004). The capabilities of the enhanced Gibbs Sampling motif finders include finding multiple motifs simultaneously (Thompson et al. 2003), modelling two-block motifs (GuhaThakurta and Stormo 2001; Liu et al. 2001), *etc*.

4.1.1.3. Considerations when using motif finding methods

Although many motif finding algorithms have been developed, computational detection of functional motifs in real genomes remains a challenging problem. Several independent surveys indicated that, in the context of genome-wide TFBS finding, the performance of available motif finding algorithms is far from being satisfactory (Hu et al. 2005; Tompa et al. 2005). An important finding is that most of the existing motif finding systems are not very effective in discriminating functional sites, particularly when complex genomes, such as the human and mouse genomes, are investigated.

Several possible reasons to the poor performance of existing motif-finding approaches have been proposed:

- The optimization procedure may get stuck in local optima.
- The background model used in many methods may be too simple to reflect the true

background in complex genomes.

- The architecture of functional sites may not be properly modelled as a single motif. For instance, TSSs may associate with two or more TFBSs.

A number of improvements have been made in order to address these issues (for review see Pavesi et al. 2004; MacIsaac and Fraenkel 2006). In the following subsection, I introduce methods that may be more suitable for prediction of functional sites in complex genomes.

4.1.2. Computational detection of functional sites

In transcription, TSSs are determined by the binding of multiple TFs to a set of TFBSs in close proximity to TSSs (for review see Fickett and Hatzigeorgiou 1997). For example, the transcription initiation of mammalian tRNA genes by RNA polymerase III is regulated by the binding of TFs to the A and B boxes (Hsieh et al. 1999) (Figure 4-1), which are within certain distances downstream of TSSs (Pavesi et al. 1994).

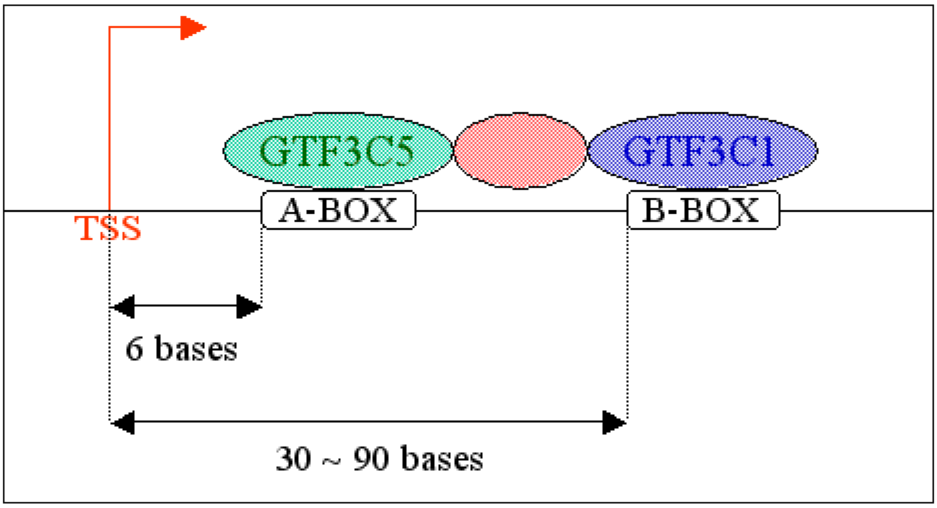

Figure 4-1. The transcription initiation of mammalian tRNA genes is regulated by A and B boxes

One computational approach for TSS finding is to model the promoters of genes, since promoters are in close proximity to TSSs. Although a number of TSS finding systems based on promoter modelling have been developed, most of them are specifically designed for finding the TSSs of protein-coding genes (for review see Fickett and Hatzigeorgiou 1997). For the purpose of finding the TSSs of ncRNAs, a system that can be used to learn new models given a new set of training sequences is of interest.

A possible approach to model TSSs is using Hidden markov models (HMMs). Complex HMMs, which recruit various states for modelling multiple signals associated with splicing and translation, have been used for finding eukaryotic protein-coding genes (Burge and Karlin 1997). Presumably *ad hoc* designed HMMs should be able to model complex regulatory elements by adequately connecting the states of relevant TFBSs. However, there are some concerns for applying HMMs to TSS modelling. First, it is generally difficult to guess a suitable HMM topology for any types of regulatory elements. Second, the parameter tuning of complex HMMs may easily be trapped in a local optimum (Durbin et al. 1998).

Over the past few years, several new systems have been developed to model regulatory modules which may consist of multiple TFBSs (Wasserman and Fickett 1998; GuhaThakurta and Stormo 2001; Bailey and Noble 2003; Zhou and Wong 2004; Aerts et al. 2005). Motif finding systems that use regulatory module models may potentially be applicable to finding promoters. However, for the purpose of predicting TSSs, there are concerns with these systems. First, the distance constraints between motifs in a module are generally un-modelled, or merely modelled by using a linear gap penalty (Bailey and Noble 2003), which appears to be unsuitable for describing the distance range between TFBSs, as observed in the tRNA gene promoters (Figure 4-1). Second, these module finding systems may report just an approximate area for regulatory modules, but not an actually functional site, which is not what we would expect from a TSS prediction algorithm.

Here, for the purpose of modelling the TSSs of ncRNA genes in the mammalian genomes, I chose to use an available system, Eponine (Down and Hubbard 2002), which was originally designed to model the TSSs of mammalian protein-coding genes. One feature of Eponine is that it has been designed to perform predictions of functional sites in genomes. Eponine has been demonstrated to be effective in discriminating TSSs (Down and Hubbard 2002) and transcription termination sites (TTSs) (Ramadass 2004) in mammalian genomes. In the following subsection (4.1.2.1.), I introduce the basics of the original Eponine implementation.

4.1.2.1. Modelling functional sites using Eponine

4.1.2.1.1. The Eponine Anchored Sequence Model

The Eponine Anchored Sequence Model (EAS) is a classification model that is aimed to be applied to individual points within a large genome, *i.e.* exact reference positions on the genome sequence, such as the base pair at which transcription starts (TSS). An essential component of the EAS model is a positioned constraint (PC), which consists of:

- A position weight matrix (PWM) which models a signal that may contribute to the classification of a particular functional site.
- A discrete probability distribution to describe the position of a PWM relative to the reference site.

In the EAS model, the score of a PC can be calculated as:

$$\phi(x,a) = \frac{\log(\sum_{i=-\infty}^{+\infty} P(i) \cdot W(x, i+a)\ )}{|W|} \qquad [4\text{-}4]$$

where *x* is a DNA sequence; *a* is a pre-defined reference site for each sequence *x*; *P*(*i*) is a discrete probability distribution for modelling the distance of a motif from the reference site (*i.e.* TSS, TTS, *etc*.); $W(x, i+a)$ is the PWM score for offset *i* relative to the reference site *a*. *P*(*i*) is usually in the form of a discrete Gaussian distribution. It should be noted is that, the PWM used in the Eponine models is actually a probability frequency matrix (PFM, see [4-1]) normalized with background base compositions. The difference between the PWM used in

Eponine and the general form of PWM (see [4-3]) is that, the latter is equivalent to the logarithm of the former. For simplicity, the term PWM is still used in describing the Eponine models, in order to be consistent with the terminology used in the papers relevant to Eponine (Down and Hubbard 2002; Down et al. 2006).

A particular point about the this scoring function is that, this function may allow, not only a strong motif with a very sharp position distribution relative to a particular reference site, but also short motifs with very broad distributions. This is caused by the summation of the position-constrained PWM scores across a region on a sequence. This design may be advantageous to the situation where there are general compositional biases toward some particular oligonucleotides, as what we have observed in the case of CpG overrepresentation in eukaryotic promoters. However, it should be noted that, by using such a scoring function, the EAS model is not designed to find optimal motifs that are over-represented in a set of sequences. Therefore, the EAS model is specifically designed to discriminate functional sites in the genomic context, *i.e.* the individual points within a large genome.

It should be noted that the final score of each PC for each sequence must be normalized by $|W|$, the number of columns in each PWM. At first glance this normalization seems to be unnecessary; however, it is critical for learning the EAS models. The reason is that, in optimizing the parameters of the EAS models, the widths of PWMs are not a pre-defined and fixed value. The learning system of Eponine learns a set of optimal PWMs from a pool of candidate PWMs of varied widths. If a PWM score is not normalized, a PWM with more columns may be preferred. Similar normalization strategies has been used by some of the motif finding systems where the lengths of motifs are not pre-defined, such as the Gibbs Motif Sampler (Lawrence et al. 1993).

Learning the EAS models

The EAS model is so built by taking the weighted sum of a number of PC scores. This complex model is equivalent to the generalized linear model (GLM) (McCullagh and Nelder 1983), where each PC in this complex model is equivalent to a basis function in GLMs.

The general formulation of a GLM can be expressed as:

$$\eta(x) = \sum_{m} \beta_m \phi_m(x) + C \qquad [4\text{-}5]$$

The term, *x*, represents a sequence. ϕ is a set of basis functions. β is a set of weights associated with individual basis functions. "C" is the constant. For binary classifications (*e.g.* classifying sequences into positive and negative ones), one logistic function,

$$\sigma(\eta) = \frac{1}{1+e^{-\eta}} \qquad [4\text{-}6]$$

can be used to transform the raw output of GLMs to fit a sigmoid curve. Thus, the output of this transformation can be used to decide whether an input *x* belongs to a particular class.

For training an EAS model, the parameters that need to be learned include PCs, and the weights that associate with PCs. Each PC consists of a PWM and an associated probability position distribution, which also need to be learned. At the initial stage of training, the parameters of PWMs and associated position distributions should be largely unknown. A trainer should be able to recruit informative PWMs and discard non-informative ones. The Eponine trainer uses a combined strategy consisting of the relevance vector machine (RVM) algorithm (Tipping 1999) and a Monte Carlo sampling process:

- A number of random PWMs of certain widths, and random Gaussian position distributions, are initialized.
- Use the RVM algorithm to estimate the weights of PCs and thus prune

non-informative PCs.

- Recruit new PCs by using a Monte Carlo sampling process to adjust the widths and weights of PWMs, as well as the parameters (*i.e.* mean and width) that decide the shape of Gaussian position distributions.

The RVM algorithm is the core algorithm for learning informative PCs. Since the RVM is so important for training the EAS model for classification, it is discussed in the following.

The Relevance Vector Machine

The RVM is a Bayesian approach to learn parameters of GLMs (Tipping 1999). It can take a set of basis functions, corresponding to PCs in the EAS model, and then use a “pruning prior” to discard the basis functions that do not contribute significantly to a particular classification problem.

In general, the Bayesian way for estimating parameters for classification can be written as:

$$P(\beta \mid X, T) = \frac{P(T \mid X, \beta) \times P(\beta)}{P(T \mid X)} \qquad [4\text{-}7]$$

$P(\beta \mid X, T)$ is the posterior probability of a model with parameter set β, given paired input and target data, *X* and *T*, where $X = (x_1, x_2, \ldots, x_N)$, represents the *N* input points (*i.e.* sequences in this thesis), and $T = (t_1, t_2, \ldots, t_N)$, represents respective targets (or responses). $P(T \mid X, \beta)$ is the likelihood of the model given the data. $P(\beta)$ is the prior probability of β and $P(T \mid X)$ is the normalization constant. For binary classifications where $t_n = [0, 1]$, the likelihood can be calculated by:

$$P(T \mid X, \beta) = \prod_{n=1}^{N} \sigma(\eta_n)^{t_n} (1 - \sigma(\eta_n))^{1-t_n} \qquad [4\text{-}8]$$

, where η_n is the predicted output (of a GLM) for an input x_n.

When there is no prior knowledge of the model parameters (*e.g.* β_m's in [9]), a non-informative prior can be used. A non-informative prior can be a uniform distribution or a very broad exponential-family distribution. However, choosing an informative prior may enable the learning of a sparse model, which contains only a few basis functions. An advantage of training a sparse model is reducing the chance of overfitting to data. To achieve sparsity, the RVM framework uses an automatic relevance determination (ARD) Gaussian prior over each weight (Tipping 1999):

$$P(\beta_m \mid \alpha_m) = G(\beta_m \mid 0, {\alpha_m}^{-1}) \qquad [4\text{-}9]$$

, where the hyperparameter, α_m, is the inverse variance of each mean-zero Gaussian distribution. This choice of prior implies that there is a strong preference that many β_m's are close to zero. After optimizing parameter β and hyperparameter α, basis functions that are not informative for classification can be decided. If α_m is extremely large, the variance of the respective Gaussian distribution will be very small and the distribution, $P(\beta_m \mid \alpha_m)$, will peak at 0. A zero weight means that the associated basis function is non-informative and could be dropped.

For optimizing GLMs, the RVM algorithm has been shown to achieve a better sparsity than do other relevant algorithms (Tipping 1999). Thus, by using the RVM algorithm, the Eponine trainer is capable of exploring a large parameter space in order to select a set of PCs which can optimize the EAS model for classification. (Down and Hubbard 2002).

4.1.2.1.2. The Eponine Windowed Sequence model (EWS)

Using the EAS model for functional sites requires a set of positive training sequences, where reference points must be labelled in these sequences. TSSs and TTSs are extremely fortunate cases because lots of experimental evidence is available to indicate relatively definable regions for these sites. However, for other cases where the existence of common regulatory elements in a set of functionally related sequences is only suspected, it is difficult to adequately label training sequences with reference sites and thus the EAS strategy is not expected to work properly. An alternative is the Eponine windowed sequence (EWS) model, which is more suitable for modelling common motifs whose locations in individual sequences are varied or unknown.

The basic formulation of basis functions used in the EWS model is:

$$\phi(x) = Z \times \underset{s=u}{\overset{v}{optimal}}(\prod_{k=1}^{K}(\sum_{i=-\infty}^{\infty} P_k(i) \cdot W_k(x, s+i)^{\frac{1}{|W_k|}})^{\frac{1}{K}} \qquad [4\text{-}10]$$

and

$$Z = \frac{1}{|u| - |v| + 1} \qquad [4\text{-}11]$$

where the interval [u, v] is the u^{th} position to the v^{th} position that are accessible by the basis function ϕ, on sequence x; P_k is the discrete probability distribution of the distance between the k^{th} PWM (W_k) and the first PWM (W_1). This complex basis function is called the convolved sensors basis function (CSBF) in the EWS models.

A CSBF may contain more than one position constrained PWM. The reason for normalizing CSBFs with $1/k$ is similar to the use of $\frac{1}{|W|}$ for normalizing the PWMs in the EAS models (see subsection 4.1.2.1.1.), because currently the number of PWMs in a CSBF is not fixed. Otherwise, without a normalization factor, a CSBF with more PWMs may be preferred by the Eponine trainer. The normalization factors, $1/k$ and $\frac{1}{|W|}$, are modifications to the original Eponine implementation (Down 2002; Down and Hubbard 2004).

In order to explain how the score calculation in [4-10] is performed, I use a CSBF consisting three position-constrained PWMs as an example (Figure 4-2). Given a sequence x, the score on the first position is calculated by multiplying the three scores given by position-constrained PWMs 1 ~ 3. Although in the plot there is just a single fixed point for each position-constrained PWM (Figure 4-2, upper-left), it should be noted that the score for each position-constrained PWM is a summation over a position distribution *P*. The final score of a CSBF given sequence *x* is the optimal one in all the scores on the interval [*u*, *v*].

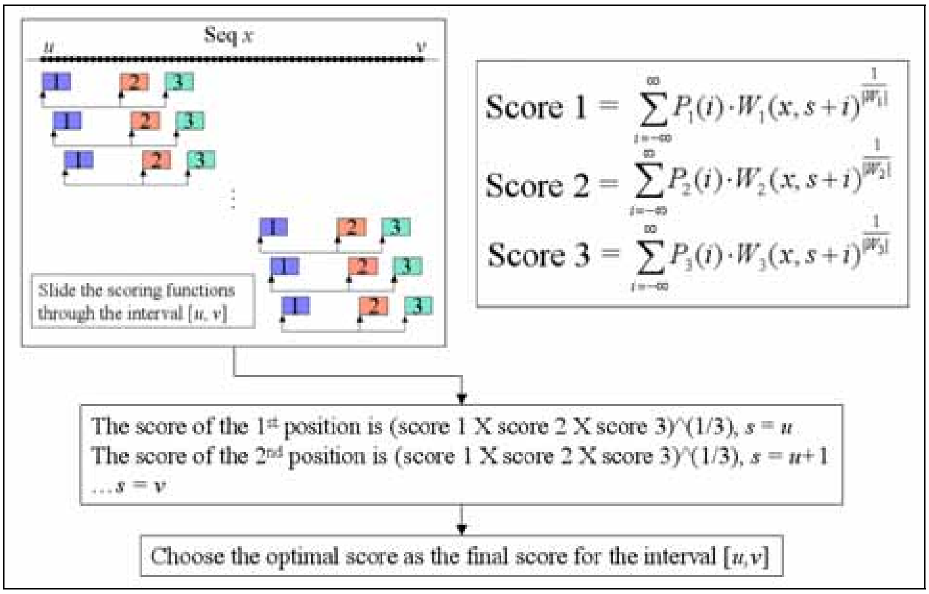
Seq x
u
v
1
2
3
Slide the scoring functions through the interval [u, v]
Score 1 = $\sum_{i=-\infty}^{\infty} P_1(i)\cdot W_1(x,s+i)^{\frac{1}{|W_1|}}$
Score 2 = $\sum_{i=-\infty}^{\infty} P_2(i)\cdot W_2(x,s+i)^{\frac{1}{|W_2|}}$
Score 3 = $\sum_{i=-\infty}^{\infty} P_3(i)\cdot W_3(x,s+i)^{\frac{1}{|W_3|}}$
The score of the 1st position is (score 1 X score 2 X score 3)^(1/3), $s = u$
The score of the 2nd position is (score 1 X score 2 X score 3)^(1/3), $s = u+1$
…$s = v$
Choose the optimal score as the final score for the interval [u,v]

Figure 4-2. How to calculate the score of a CSBF consisting of three PWMs and associated position distributions

Learning the EWS models

For training the EWS model, two types of parameters must be learned: 1) the probability distribution of positions and 2) PWMs. For distributions of positions, the training process is very similar to that for training the EAS models (see subsection 4.1.2.1.1.), except that the reference site is replaced with one of the position constrained PWMs in each CSBF. The Monte Carlo sampling process is used to optimize the choice of CSBFs. A new member PWM is randomly sampled from the pool of CSBFs, and then the so generated new CSBFs, will be re-weighted and pruned by using a RVM strategy. Through iterating the Monte Carlo sampling process and the pruning process using the RVM, an EWS model consisting of a set of CSBFs could be learned.

4.2. Modelling local RNA motifs

In the previous parts of this thesis, ncRNA classifiers and the modelling of transcription regulatory elements of ncRNAs have been discussed. Due to the particular types of signals that are used in these methods, there are certain limitations on the scopes of their applications. Firstly, existing comparative algorithms may overlook the RNA structural motifs spanning only a region in a transcript. Secondly, when modelling transcription regulatory elements, any RNA motifs implicated in the regulation of ncRNA expression are essentially ignored.

The transcripts of ncRNA genes are not the only RNAs that may contain RNA structural motifs. Evidence suggests that local RNA structures may be implicated in the regulation of protein translation (for review see Kozak 2005). Besides, single-stranded regions in transcripts can also be part of functional motifs (for review see Mattaj 1993). The local RNA motifs discussed here are considered as a composite of primary-sequence patterns and local RNA structures, where different parts of a composite motif may be separated by unstructured and/or functionally unimportant regions of variable length.

One type of computational approach for identifying local RNA motifs is to search for the consensus RNA motifs in a group of functionally related transcripts. Existing algorithms for finding consensus RNA motifs in transcripts can be generalized into three major categories: variants of the Sankoff's algorithm, variants of stochastic context-free grammars (SCFGs), and variants of genetic algorithms. In the following subsection (4.2.1.), I briefly introduce existing algorithms for finding local RNA motifs, and the considerations in using these algorithms.

As previously discussed (see subsection 4.1.2.), computational modelling of functional sites requires algorithms that can combine the contribution from multiple TFs. A similar approach is required to combine the contributions of local RNA motifs to generate a predictive model. In an attempt to address this, I developed a new RNA motif extension to the Eponine modelling system. The addition of this new extension allows the modelling of functional sites as a composite of primary-sequence and secondary-structure motifs from a set of unaligned functionally related sequences. This is described in subsection 4.2.2.

4.2.1. Available methods for finding consensus RNA motifs in sequences

4.2.1.1. The Sankoff's algorithm and variants

Given a set of sequences, Sankoff's algorithm can generate optimal primary-sequence alignment and secondary-structure minimum free energy (MFE, see subsection 1.3.1) simultaneously (Sankoff 1985). However, the time complexity is $O(N^{3K})$ and the space complexity is $O(N^{2K})$, where N is the sequence length and K is the number of sequences. It is therefore not practical to apply Sankoff's algorithm to finding consensus RNA motifs in a set of sequences. Variants of Sankoff's algorithm have thus been created in order to find consensus RNA motifs in an acceptable time. Two modifications have been adopted by different implementations in order to accelerate the search process. Firstly, only local hairpins are considered by inhibiting branching configuration. A branching configuration is the partition of one sequence into two structural regions in the base-pair dependent energy rule (Nussinov and Jacobson 1980). Inhibiting branching configuration is equivalent to taking out $W(i,k-1)$ from [1.2] of subsection 1.3.1.1. , reducing the time complexity from $O(N^6)$ to $O(N^4)$ for pairwise alignments.

The second modification for accelerating Sankoff's algorithm is to use progressive alignment methods. The strategy of progressive alignment methods is to find the best pairwise alignments first, and then other alignments or single sequences can be consecutively added to existing alignments. In the primary form of progressive alignment methods, once a group of sequences have been aligned, their relations cannot be altered at later steps. The procedure of combining alignments terminates when all sequences have been aligned. The time complexity can be $O(L^4N^4)$, where L is the average sequence length; N is the number of sequences (Gorodkin et al. 2001).

Progressive alignment methods can efficiently generate acceptable multiple sequence alignments; however, these methods are greedy and alignments can be trapped in a local optimum. The reason for this is that the best pairwise alignments do not necessarily contain optimal motifs shared by all sequences, and globally optimal motifs may be only sub-optimal when comparing two sequences. When finding primary-sequence motifs, additional approaches can be used to improve multiple sequence alignments. Related techniques include iterative refinement methods, simulated annealing, Gibbs sampling, *etc* (For reviews see Durbin et al. 1998). Nonetheless, no variants of Sankoff's algorithm use these approaches and the primary form of progressive alignment methods is still the most common strategy used by variants of Sankoff's algorithms.

4.2.1.2. The stochastic context-free grammars (SCFGs)

Just as in the prediction of RNA secondary structures, statistical models, such as SCFGs (see subsection 1.3.3) and McCaskill's sampling algorithm (McCaskill 1990), can replace MFE for finding the consensus RNA motifs among sequences. PMcomp/PMmulti (Hofacker et al. 2004) uses McCaskill's sampling algorithm to do pairwise/multiple structural alignments. Its time complexity and space complexity is as high as $O(N^6)$ and $O(N^4)$ respectively for pairwise alignments. The computational complexity of PMcomp/PMmulti is not less than that of Sankoff's algorithm. For multiple structural alignments, it also uses progressive alignment methods in order to restrict computational complexity. For pure SCFGs-based algorithms that can do *ab initio* structural alignments, the computational complexity is at least as high as for the original Sankoff's algorithm. In order to reduce complexity, variants of SCFGs (Knudsen and Hein 1999; Knudsen and Hein 2003) take alignments that are generated by popular multiple-sequence-alignment programs, such as ClustalW, and then refine alignments using SCFGs. One problem with this approach is that the quality of initial multiple sequence alignments nearly determines the performance of variants of SCFGs. If the initial alignments were trapped in a local optimum in terms of RNA motifs, it seems unlikely that further refinement at the structural level could give optimal answers (Knudsen and Hein 1999). In addition, perfectly identical RNA secondary structures, which may not be always practical for modelling RNA motifs in genomes, are sometimes assumed (Knudsen and Hein 2003).

4.2.1.3. Genetic-algorithm based approaches

Unlike the current implementations of variants of Sankoff's algorithm or variants of SCFGs, GA-based approaches are less easily trapped in a local optimum. Although GA-based approaches are not guaranteed to find the optimal solution, they can be very good in predicting RNA structures (Chen et al. 2000; Taneda 2005). One problem with the current GA-based approaches is that primary-sequence motifs are not generally considered as part of RNA motifs; few GA-based approaches have been designed to find both types of motifs simultaneously.

4.2.1.4. Uncategorized RNA-motif finding approaches

There are other types of consensus RNA-motif finding algorithms that cannot easily be classified into the above categories. One type of algorithms is to take folded sequences and then align the predicted RNA structures. These programs do not predict RNA structures by themselves. Instead, the structure of each sequence may be taken from the prediction made by MFE-based RNA secondary-structure prediction algorithms, such as Mfold (Zuker 1989) and RNAfold of the Vienna package (Hofacker 2003). MARNA (Siebert and Backofen 2005), RNAForester (Hochsmann et al. 2004), and RNADistance (Hofacker 2003) are three examples.

For instance, from the predicted RNA structures for sequences, MARNA identifies seeds of both primary-sequence and RNA structural motifs and then feeds these motifs to T-Coffee (Notredame et al. 2000). One concern with such algorithms is that their performance can be influenced by the accuracy of the optimal global structures predicted. Besides, these algorithms may be vulnerable to the cases where the consensus RNA motifs between a set of sequences is quite different from the optimal structures for individual sequences.

Another type of algorithms, such as RNAalifold (Hofacker et al. 2002) and MSARI (Coventry et al. 2004), are designed to find consensus RNA motifs in primary-sequence alignments that are generated by using popular multiple sequence alignment programs, such as ClustalW. These algorithms take compensatory mutations as the evidence for supporting the existence of a global RNA motif (Coventry et al. 2004; Washietl et al. 2005). One concern with these algorithms is that, they depend on the primary-sequence alignments, which may, under certain circumstances, be incapable of revealing the consensus RNA structures between sequences. Their performance should be sensitive to the sequence identities between given sequences, although the required identities were not clearly defined in their original papers.

Consequently, currently available algorithms are not practical enough for modelling regulatory RNA motifs in genomes, since there are so many considerations and restrictions in using them. Given a set of functionally related regions in transcripts, there should be an algorithm that can model both common primary-sequence and structural motifs efficiently. The resulting model should be potentially applicable to genome-wide regulatory RNA motif finding. Therefore, I extended Eponine to include local RNA structural motifs in order to create an ncRNA modelling tool, which can be applied to finding RNA-motif associated functional sites in genomes.

4.2.2. Extending Eponine to include RNA structural motifs

Both the EAS and EWS models of the Eponine package (see subsection 4.1.2.1.) are useful for modelling primary-sequence motifs and the relations of motifs to other reference sites. Similarly the Eponine RNA-motif extension should model both RNA structural motifs and the relations of structural motifs to other sites. RNA motifs should be considered as yet another type of motifs that are in sequences, except that RNA motifs possess structural features, including stems and loops. In brief, the Eponine RNA-motif extension aims at modelling the regulatory RNA motifs that are constituted by specific arrangement of both primary-sequence motifs and structural motifs, with appropriate scoring scheme.

Primary-sequence motifs are modelled by PWMs in the EAS and EWS models. Similarly, a formal description of structural features must be chosen in order to extend both the Eponine models to include structural motifs. One possibility for modelling individual hairpins is to use Covariance Models (CMs), which are SCFG-based RNA profiles. However, for several reasons, I decided that CMs may not be adequate for extending Eponine models. Firstly, training an Eponine RNA-motif model that consists of CMs can be very time-consuming, because numerous CMs can be temporarily generated in the training process and each must be assessed and updated. The time complexity of evaluating each CM is at least $O(L^3)$, where L is the length of each candidate region for a particular hairpin (Durbin et al. 1998). Secondly, it is difficult to adapt the scores of CMs on sequences for EAS and EWS models. Distributions of the CM scores may vary greatly across different types of RNA motifs. There is no obvious solution for combining the CM scores and the PWM scores in order to model primary-sequence and structural motifs simultaneously.

Another question for modelling RNA motifs is how to properly address variations of structural features. Although variations in hairpins are commonly believed to be disastrous for some structural RNA genes, evidence indicates that a certain degree of variation exists in RNA structural motifs of similar functions. An example is the transcription termination signals of bacterial genes, where the sizes of stems can vary from 5 to 30 base pairs and the lengths of loops vary from 3 to 9 bases (de Hoon et al. 2005).

Using existing probabilistic models cannot properly address dimensional variations of RNA structural motifs. For instance, standard CMs using general topologies can tolerate small size variations of hairpins, but they cannot model these variations explicitly. To explicitly model such variations, CMs need additional techniques, such as duration modelling. Duration modelling is a technique used for addressing the length distribution explicitly (Durbin et al. 1998). However, if such techniques are used, the computational complexity will be much higher. In addition, other structural features, such as folding energies of hairpins, may still need to be modelled by other yet unmentioned techniques.

Therefore, in developing the RNA motif extension of Eponine, I decided to use a local RNA structural model which is not based the classic probabilistic model of RNA structures, such as CMs. The new model should be able to model a variety of features of local RNA hairpins. There are two steps in training the models: firstly, candidate hairpins for each sequence should be first located; and secondly, the Eponine trainer learns a model describing the structural features of the consensus RNA motifs of these sequences. In the following two subsections, I introduce the implementation of the Eponine RNA-motif extension, including the approaches to locate local hairpins (subsection 4.2.2.1.) and the way structural features are modelled (subsection 4.2.2.2.).

4.2.2.1. Locating local hairpins

The RNA motifs, which the Eponine RNA-motif extension is designed to model, are specific arrangements of a set of single-stranded and double-stranded regions in sequences. Consequently, predicting and evaluating RNA secondary structures of given sequences is necessary. It is reasonable to assume that any position in each sequence can be the start point of a hairpin structure. Proposed RNA motif models should evaluate all hairpins that may start at each position of each sequence.

Predicting hairpins that may be functionally important is not straightforward. Firstly, optimal structures can be predicted only for regions of restricted length, but not for the full-length region of long sequences. The time complexity for predicting optimal structures by using either MFE or SCFGs is proportional to the cubic sequence length. Given any fragment of genomic sequence, one practical strategy for finding candidate functional motifs is to chop the original sequence into consecutively windowed regions and then predict hairpins for individual regions. Although this approach may sacrifice some hairpins that span a region larger than the window size, stable hairpins within windowed regions can still be predicted. It is also reasonable to infer that long-range interactions in large hairpins should depend on stable hairpins within windowed regions. By evaluating hairpins in windowed regions, trained models can be applied to genome-wide RNA motifs finding; all regions in each sequence can be consecutively evaluated by sliding the windows through all positions. Similar strategies have been used by other algorithms for genome-wide ncRNA finding (Rivas and Eddy 2001; di Bernardo et al. 2003). The time complexity of folding windowed RNA secondary structures for multiple sequences is thus $O(LNM^3)$, where L is number of sequences; N is the average number of windows per sequence; M is the length of windowed regions.

Secondly, predicting the sub-optimal hairpins for each sequence seems necessary. Evidence suggests that optimal structures do not necessarily represent the functional forms of various regulatory RNA motifs. In addition, RNA folding may alter in response to certain conditions, such as the binding of ligands, increases in di-ionic strength in solution, interaction with RNA binding proteins, post-transcription modifications, *etc*. For finding consensus RNA motifs among sequences, only optimal folding for each sequence may not be sufficient.

Exhaustively enumerating all possible hairpins that may fold in each sequence is computationally expensive and impractical. There are at least two simpler approaches for predicting sub-optimal hairpins for each sequence. The first approach is to collect the optimal hairpin for each position of each windowed region (Figure 4-3, algorithm A). For each position *i* within a windowed region, the optimal hairpin, which is conditioned on that position *i* must pair with another position *j*, is saved, where $i < j <$ window size. By scanning sliding windows for each sequence, optimal hairpins that start at individual positions in each sequence are collected. These site-specific optimal hairpins are not necessarily the components of globally optimal structures. This approach is similar to Zuker's suboptimal folding algorithm, and to the inside and outside directions of the CYK algorithm (Durbin et al. 1998). The consideration of this approach is time complexity. In addition to the time complexity $O(N^3)$ for calculating the energy matrix in using Zuker's MFE algorithm, additional time complexity, $O(\text{window size}^2)$, is required in order to trace respective optimal hairpins for all possible paired positions in each windowed region.

By contrast, the second approach for collecting sub-optimal hairpins for each sequence is much simpler. Only the optimal structure for each windowed region is predicted (Figure 4-3, algorithm B). From the optimal structure, individual hairpins are extracted, and then saved with their respective start positions. By scanning sliding windows for each sequence, a series of optimal hairpins that start at distinct positions in each sequence are collected. Just like the situation of the first approach, these site-specific optimal hairpins are not necessarily the components of optimal global folding. The second approach can be much faster than the first one, because much less folding space is explored (Figure 4-3).

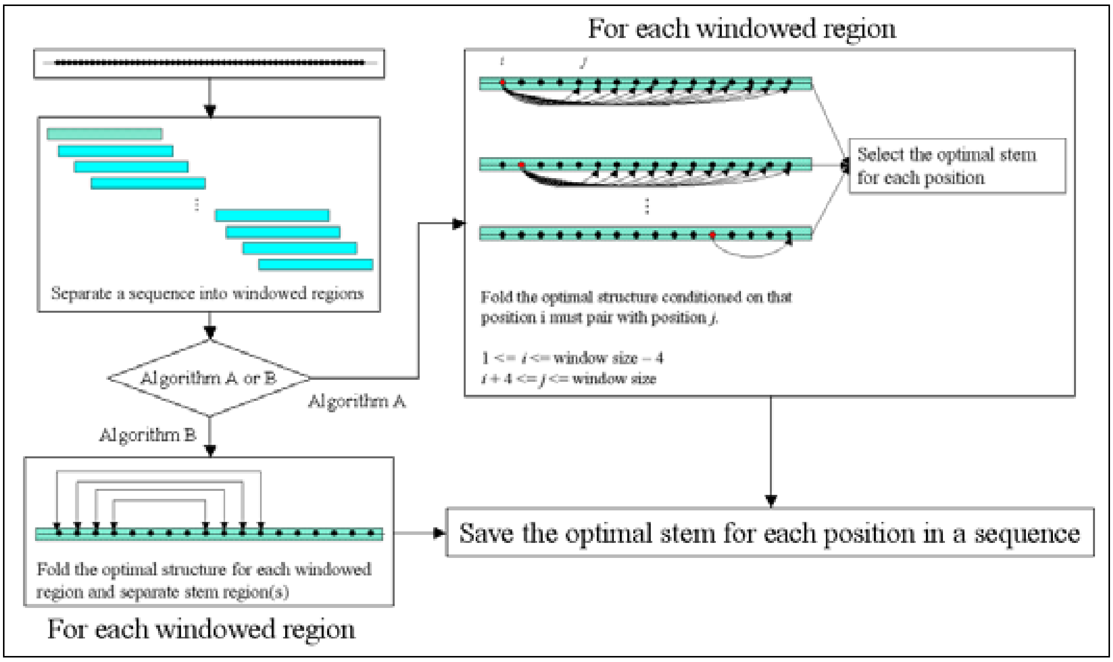
For each windowed region
Select the optimal stem for each position
Fold the optimal structure conditioned on that position i must pair with position j.
1 <= i <= window size – 4
i + 4 <= j <= window size
Separate a sequence into windowed regions
Algorithm A or B
Algorithm A
Algorithm B
Save the optimal stem for each position in a sequence
Fold the optimal structure for each windowed region and separate stem region(s)
For each windowed region

Figure 4-3. Two modes (algorithm A: the stringent mode and algorithm B: the fast mode) for finding local hairpins for windowed regions

In order to compare the performance of different approaches for predicting RNA structural motifs, human tRNAs of exactly the same length, 72 bases, were used as the test data set. Windows of different sizes were also tried to investigate possible effects. The targets for this evaluation included D arm, anticodon arm, and T arm (Figure 1-3), of 168 human tRNAs. The implementation for predicting RNA structures follows Zuker's MFOLD algorithm and uses the same parameters (Zuker 1989). The result reveals that the first approach (Algorithm A, Table 4-1) is better than the second one (Algorithm B, Table 4-1); however, it also suggests that the second approach is still useful, if the results of the second approach are compared to the predictions made by RNAfold (default, RNAfold, Table 4-1) (Hofacker et al. 1994-2006) with default parameters.

Algorithm A

	D arm	Anticodon arm	T arm
Window size: 50	112	150	132
Window size: 100	112	150	131

Algorithm B

	D arm	Anticodon arm	T arm
Window size: 50	80	146	131
Window size: 100	64	142	131

RNAfold

	D arm	Anticodon arm	T arm
default	35	28	58

Table 4-1. Performance of different algorithms for three hairpins of 168 human tRNAs

Algorithm A: The stringent mode. Individual hairpins are extracted from all optimal structures conditioned on that the i^{th} base should pair with the j^{th} base in each windowed regions, where $i < j <$ window size.

Algorithm B: The fast mode. Individual hairpins are extracted from the optimal structure for each windowed region.

Values in cells are the numbers of correct predictions (made by different algorithms) for respective arms. For D arm, the criteria of correct prediction is existence of a hairpin at 9th or 10th position, with stem size 3 ~ 4 base pairs and loop sizes 7 ~ 10 bases. For anticodon arm, the correct prediction should be at 26th or 27th position, with stem size 4 ~ 5 base pairs and loop size 7 ~ 9 bases. For T arm, the correct prediction should be at 48th or 49th position with stem size 4 ~ 5 base pairs and loop size 7 ~ 9 bases. The performance of RNAfold is assessed by using its default parameters.

In addition to the successful identification of three distinct hairpins of tRNAs, both Algorithms A and B predict extra hairpins. The biological significance of these extra hairpins is not clear. It is possible that these secondary structures could never fold in real tRNAs because they are relatively unstable compared to the optimal structures of individual tRNAs. By using the Eponine learning scheme, this redundancy should not be a serious problem, because only stable hairpins that can be consistently found in individual sequences are useful for distinguishing positive training sequences from negative training sequences. In the following text, algorithms A and B are referred to as the stringent model and the fast mode, respectively, of the Eponine RNA-motif extension.

4.2.2.2. Modelling structural features with probability distributions

Having evaluated the capability of the module responsible for locating local hairpins in sequences, consideration is now given to applying the Eponine training framework to model RNA motifs. One important issue is about designing a scoring scheme of the secondary structures in sequences.

Before moving further to discuss the scoring of complex RNA motifs composed of many hairpins, the scoring of a simple hairpin is first considered. In an oversimplified hairpin (Figure 1-2, A), there is only one single-stranded region (hairpin loop), and one non-interrupted double-stranded region (stem). Numerical parameters, which can potentially be applied to distinguishing one simple hairpin from the other, include dimensions of hairpins, free energy of the local region, free energy of the stem region, *etc*. Dimensions of each hairpin include loop size and stem size. If functions of RNA structural motifs depend on adequate combinations of individual features, then it seems reasonable to draw an analogy between primary-sequence motifs and features of RNA hairpins. Each feature of a hairpin seems analogous to each column of a PWM.

Each column of PWMs is a discrete distribution over all possible symbols in the used alphabet; similarly, each feature of hairpins can be modelled with a probability distribution. The mean of each distribution is the most frequently found value for one particular feature. For example, because the most frequently found stem size for *rho*-independent transcription termination signals is 9 (de Hoon et al. 2005), the mode of the corresponding discrete probability distribution should be 9. The deviation of each distribution can represent the degree of variations, such as different stem sizes that are observed in *rho*-independent transcription termination signals.

The probability of emitting a sequence *x* that harbours an RNA structural motif (RM) is:

$$RM(x,i) = (\prod_{r=1}^{R} P_r(F_r(x,i)))^{\frac{1}{R}} \qquad [4\text{-}12]$$

, where *R* is the number of features that are used to model each hairpin; P_r is the proposed probability distribution of the r^{th} feature of a particular RNA structural motif; the model of this structural motif is $P = (P_1, P_2, P_3..., P_R)$; F_r is the function that returns the numerical value of the r^{th} feature of a hairpin, which folds at the i^{th} position of sequence *x*. $\frac{1}{R}$ is used to normalize the score of each hairpin. It seems this normalization is unnecessary; however, it is very important for modelling primary-sequence and structural motifs simultaneously. For each primary-sequence motif, the PWM score is the normalized joint probability of individual positions. For generating a scoring scheme that can sensibly combine scores from both PWM scores and RM scores, a similar normalization that is applied to PWM scores should also be applied to hairpin scores.

Compatibility between RM scores and PWM scores is one of most critical issues in developing the Eponine RNA-motif extension. If the extension uses an inappropriate scoring scheme that may make the order of magnitude of RM scores significantly different from that of PWM scores, the trained models may be biased to contain only RMs or only PWMs. Before the use of normalized RM scores, empirical rules have been used in order to make non-normalized RM scores compatible with PWM scores. For example, by comparing distributions of the scores of PWMs and non-normalized RMs, some multiplication factors were derived for transforming RM scores. However, the optimal value of the multiplication factor may change greatly under different conditions, especially when more than two different structural features are used to model RNA structural motifs.

By using joint probability of structural features to score each hairpin, many structural features can be modelled explicitly. By contrast, some features, such as stability of a particular hairpin, cannot be modelled explicitly by using CMs. In addition, with normalized RM scores, distinct features can be treated as individual columns of a PWM. Theoretically, it is possible for the Eponine trainer to randomly choose distinct features to learn an optimal sensor for an RNA structural motif, just as the addition and subtraction of columns in learning the optimal PWM for modelling a primary-sequence motif (for details see subsection 4.1.2.1.1.).

Currently, the probability distribution for modelling each structural feature is a discrete Gaussian distribution; however, it should be noted that a discrete Gaussian distribution may not be the best one for describing all the distributions of stem size, loop size, local energy, *etc*. If there is a strong peak in the distribution of structural features, the width (deviation) of a Gaussian distribution should be assigned a small value, such that the there are light tails in this distribution. However, in cases where the distribution of features is flat within a certain range, the width of the Gaussian distribution must be a large value in order to simulate the flatness in local regions.

4.2.2.3. Applying RM scores to the EAS and EWS models

With the RM scoring scheme created in the previous section, the Eponine RNA-motif extension is able to model RNA motifs that are composed of primary-sequence patterns and secondary-structure motifs. In the following, the way the RM scoring scheme is adapted into the existing Eponine sequence models is introduced.

4.2.2.3.1. Using RM scores in the EAS model – the Eponine Anchored RNA-motif model

The formulation of basis functions for the EAS model is:

$$\phi(x) = \frac{\log(\sum_{i=-\infty}^{+\infty} P(i) \cdot W^{'}(x, i+a)\ )}{|W^{'}|} \qquad [4\text{-}13]$$

For modelling structural motifs:

$$W^{'}(x, i + a) = \exp(RM(x, i + a)) \qquad [4\text{-}14]$$

and

$$|W^{'}| = 1 \qquad [4\text{-}15]$$

The operation "exp" is used for avoiding the exceptional situations where the returned value from a *RM* is 0. This situation may occur when there are no significant RNA motifs starting at a particular position in a sequence. $|W^{'}|$ is assigned with 1, because the normalization has been performed in the calculating the value of each *RM* (see [4-12]). Apart from that, for modelling primary-sequence motifs, $W^{'}$ is simply replaced with *W*. Such an extension to the Eponine EAS model is referred to as the Eponine Anchored RNA-motif model (the EAR model)

The new Eponine trainer uses a Monte Carlo sampling process for learning an optimal set of positioned RMs: 1) the mean and width of distributions are assigned randomly; 2) new RMs are generated by sampling features from all hairpins predicted in all training sequences; 3) new RMs can also be generated by adjusting the mean or the width of randomly chosen distributions of structural features in existing RMs. After positioned RMs are updated, the Eponine trainer uses the RVM to re-estimate their respective weights, which correspond to weights of basis functions in GLMs.

4.2.2.3.2. Using RM scores in the EWS model – the Eponine Windowed RNA-motif model

The formulation of basis functions for the EWS model is:

$$\phi(x) = Z \times \underset{s=u}{\overset{v}{optimal}}\left(\prod_{k=1}^{K}\left(\sum_{i=-\infty}^{\infty} P_k(i) \cdot W^{'}{}_{k}(x, s+i)^{\frac{1}{|W^{'}{}_{k}|}}\right)^{\frac{1}{K}}\right) \qquad [4\text{-}16]$$

For modelling structural motifs, $W^{'}$ is substituted with *RM*. For modelling primary-sequence motifs, $W^{'}$ is substituted with *W*. Such an extension to the Eponine EWS model is referred to as the Eponine Windowed RNA-motif model (the EWR model).

The Eponine trainer uses a Monte Carlo sampling process, which is similar to the optimization of RMs for the EAS models, to optimize the parameters of RMs for the EWS models.

Consequently, by using the scoring scheme designed to simultaneously model RNA structural and primary-sequence motifs, Eponine is now capable of modelling the consensus RNA motifs in a set of anchored or unanchored sequences.

4.3. Summary

In this chapter, I introduced methods for motif finding and functional site finding in preparation for modelling regulatory regions that may be implicated in the transcription of ncRNAs. For the purpose of finding functional sites, such as TSSs and TTSs, in complex genomes, there are three main requirements:

- Modelling an association of multiple motifs to describe functional sites.
- Modelling the distribution of individual motifs with respect to a particular functional site location.
- High selectivity in classification of functional sites in a large genome.

At the time of preparation of this thesis, Eponine appears to be one system that takes all these issues into consideration. Therefore, in the next chapter, the Eponine sequence models are applied to the modelling of the TSSs of mammalian RNA polymerase III genes.

In addition, I developed a new RNA-motif extension to the Eponine sequence models.

This new extension is particularly designed for finding the consensus RNA motifs in a set of sequences. The unique features of this new tool include that:

- It is an alignment-independent method.
- The models so trained may consist of primary-sequence patterns and secondary-structure motifs, which may give insights to the functional regions in a set of sequences.
- It is a local RNA-motif modelling tool, which means that a global conservation of RNA secondary structures in the set of sequences under investigation is not required.
- It may still work if not all the sequences under investigation fold into the same RNA motifs.
- The models so trained may potentially be useful for discriminating in genomes the functional sites associated with RNA motifs.

Chapter 6 is dedicated to the evaluation of the capability of the new RNA-motif modelling tool. The potential applications of the Eponine RNA-motif extension in genome-wide ncRNA finding will also be explored.