Chapter 5

Predicting protein transmembrane topology and signal peptides: An HMM approach with a new parameter optimisation strategy

5.1 Introduction

5.1.1 The aim of this study

In Chapter 3, we have seen that using predicted protein transmembrane (TM) topology statistics, such as the fraction of N-terminal amino acids lying in a TM helix and the number of TM helices, improves subcellular localisation prediction. In this chapter, I investigated the possibility of developing an alternative to TMHMM (Krogh *et al.*, 2001) that was used in Lokum.

In this chapter, I also introduce a new strategy to optimise hidden Markov model (HMM) transition probabilities, based on nested sampling. This is an alternative to the classical approach of Baum-Welch optimisation procedure (see Section 1.3.1).

I tested this new methodology in optimising transition probabilities of an HMM that tries to automatically annotate a set of given sequences with their most probable TM topologies, and presence of SPs. This prediction is a mapping procedure of the most probable state path ("annotations") to best describe a sequence, according to a pre-determined model (see the "second task of HMMs", on page 15). Thus we require a good HMM model to make predictions from. Using our *a priori* knowledge about what sequence regions are preceded by what other sequential features etc., it is possible to construct a finite, deterministic state machine to describe this problem. We are also given a set of sequences from which it is possible to directly determine the emission probabilities of the symbols that a state can emit.

However, usually there is only a couple of available options to determine the relations, or transition probabilities, between these states: the use of the Baum-Welch algorithm to find a set of "optimal" probabilities by trying to find the optimal ordering of states which will maximise the series multiplication of emitted symbols' probabilities, or to manually set them. If there are multiple states in the HMM having the same emission probability distributions (like a loop state and a globular-region state that are both trained with some cytoplasmic sequence), the transition optimisation will be harder by the Baum-Welch algorithm which already does not guarantee finding the best probability set.

With the method I will introduce I try to overcome these difficulties by:

- using nested sampling to search the whole parameter space and also partially to steer Baum-Welch, which reduces the chance of getting stuck in a

local maxima, and,

- following a fully supervised training that utilises known state labels of a given set of training data.

5.1.2 Transmembrane topology and signal peptide prediction

Membrane proteins span bilayer lipid phases. Membrane spanning regions of these proteins are usually made up of transmembrane α-helices or antiparallel β-sheets. Most of the membrane proteins have α-helices, although there are a number of proteins containing β-barrel structures in the outer membrane regions of bacteria, and in the organelles mitochondria and chloroplasts. Tight bundling of these α-helical segments forms globular structures in membrane proteins. A typical transmembrane α-helix contains around 20-25 predominantly hydrophobic amino acid residues. This property forms the basis of computational methods in identifying membrane proteins.

Like transmembrane α -helices, SPs are also rich in hydrophobic residues. SPs typically range in length between 20 and 30 amino acids in eukaryotes (Emanuelsson *et al.*, 1999; von Heijne, 1990), however it is possible to have up to 70aa long SPs (for example, the SP of a protein, P1383, "Ring-infected erythrocyte surface antigen precurser" is 65aa long). They can be divided into three sections in terms of their amino acid content (see Section 3.3.1), with the core hydrophobic region and the cleavage site being more conserved (see Section 5.3.1 and Figure 5.5 in Results, as an interesting note on how mRNAs of SPs look like). This tri-partite structure is quite useful to predict SPs. Also, their cleavage sites feature a "-3,

-1" rule (von Heijne, 1986), corresponding to the positions occupied by small, conserved amino acids like G or A relative to the actual cleavage position (Figure 3.4a).

Membrane topology describes which regions of the polypeptide chain span the membrane, and which portions lie on either of the watery sides of the lipid bilayer. Membrane topology prediction is important in many ways, as it can help biochemists design drugs or antibodies etc. which are bound to a membrane protein. Many researchers have studied automatic transmembrane topology prediction, and many predictors including TopPred (Claros & von Heijne, 1994), SOSUI (Hirokawa *et al.*, 1998), TMHMM (Krogh *et al.*, 2001) and HMMTOP (Tusnády & Simon, 2001) have been developed in recent years. In 2001 Müller *et al.* showed that all transmembrane prediction methods available at that time had a tendency to interpret hydrophobic parts of signal sequences and transit peptides as membrane-spanning regions. A year after this study, Lao *et al.* evaluated 12 transmembrane topology prediction methods, including the popular ones mentioned above, for their abilities to discriminate between signal peptides and transmembrane regions. These review studies showed that there is still room for improvement in the prediction performance of these programs. While it was shown that TMHMM performed better than the rest of the predictors, in general all the tested programs were badly affected by the presence of a signal peptide in tested sequences. Examples for other TM predictors developed after 2001 are ENSEMBLE (Martelli *et al.*, 2003), Phobius (Käll *et al.*, 2004), PONGO (Amico *et al.*, 2006), PRODIV-TMHMM (Viklund & Elofsson, 2004), and MEMSAT 3 (Jones, 2007).

Käll *et al.* (2004) developed a hidden Markov model (HMM) based system called Phobius, which combined the transmembrane protein topology predictor TMHMM (Krogh *et al.*, 2001) and the signal peptide (SP) predictor, SignalP (Nielsen *et al.*, 1997b) (SignalP is discussed in 1.1.1). This combinatorial design of Phobius has been shown (Käll *et al.*, 2007) to improve the performance of TMHMM: By forcing the predictor to choose either of the two sub-models, they increased the discrimination rate between transmembrane regions and N-terminal signal peptides, which resulted in fewer false positives for transmembrane regions.

Unfortunately, the stand-alone version of the Phobius program, although it is downloadable from the program's prediction service web page, does not come with the "model file" which contains the crucial program parameters. Academic users who want to use this application on their local servers or computers are required to sign a user license agreement. The "terms and conditions" of this license restricts full ownership of even other independent programs that somehow use Phobius or its modifications.[1] In the Lokum localisation prediction system I used TMHMM, because of the mentioned limitations in Phobius, and also because TMHMM is available as a stand-alone application. However, because Phobius has been shown to outperform TMHMM, I chose Phobius to be my sample model as a transmembrane predictor. Thus, the developed prototype predictor is an HMM system whose architecture is similar to that of Phobius.

[1]The LICENSOR retains ownership of the SOFTWARE delivered to the LICENSEE. Any modifications or derivative works based on the SOFTWARE are considered part of the SOFTWARE and ownership thereof is retained by the LICENSOR, and are to be made available to him upon request.

5.2 Materials and methods

5.2.1 Architecture of the HMM

It is no surprise that most of the major transmembrane protein prediction programs use Hhidden Markov models (HMMs) to predict protein transmembrane topology. The prototype predictor introduced in this chapter is also based on HMMs (see Section 1.3.1 for a brief description of HMMs).

The architecture of the program introduced here (Figure 5.1) is similar to that of Phobius (Käll *et al.*, 2004). In this model, I used an SP cleavage site motif by directly attaching it into the HMM as a "profile HMM" (Section 1.3.1) where inner states have no self-transitions. This motif was discovered by NestedMICA from a set of secretory protein sequences for the developed localisation prediction program Lokum, and is shown in Figure 3.4.

There are two major possible routes a sequence can be "threaded" into the shown HMM architecture: it could start by traversing through the SP states if this is a more probable option as determined by the dynamic programming part of the algorithm, or it can choose to go directly to the hub state.

In the first route, the SP is modeled as consisting of a three parts: An n-part, a hydrophobic core part (h-part), and a c-part which includes the cleavage site and connects it to the rest of the mature protein region. From here on, it can either go into a short or a long non-cytoplasmic state. Note that, if a sequence has an actual SP, it is not possible for the adjacent part to be in a cytoplasmic region, as the SP will be pointing towards the ER and will drag the rest of the mature part which remains behind it. However, if the N-terminal

were a non-SP transmembrane region, it could penetrate into the membrane from either direction. So the described HMM was designed to reflect this biological phenomenon, by not allowing an SP signal to be followed by a cytoplasmic loop.

The second route that can be followed is to directly go to the hub state of the HMM, from where it is possible to go to either a cytoplasmic or a non-cytoplasmic region (the hub state serves as a symbolic state for better visualising state connections and does not emit any symbols). Non-cytoplasmic loops have been modeled as two states: one for modeling shorter ones, and one for the relatively longer loops. Both loops can be followed by a globular region, although this is not necessary. If a non-cytoplasmic globular state is visited from a non-cytoplasmic state, the system has to go back through the same type of loop, namely the same non-cytoplasmic state (the short or the long one). On the contrary, cytoplasmic loops were not modeled as two separate states, as it has been suggested that their length distributions show less variation (Krogh *et al.*, 2001). Similar to the loops on the other side, cytoplasmic loops can also be proceeded by cytoplasmic globular regions.

Emission probabilities for the cytoplasmic and non-cytoplasmic globular states in the HMM have been trained by using cytoplasmic and non-cytoplasmic amino acid sequence chunks of a set of annotated proteins (see "Datasets" below, Section 5.2.2). Similarly, amino acid distributions of SP, transmembrane and loop states were trained from the corresponding sequence chunks. As stated above, the emission probability distributions for the cleavage site positions of SP states were determined directly using the weights of the discovered cleavage site motif, instead of using a single distribution to represent the entire signal. Unlike cytoplasmic

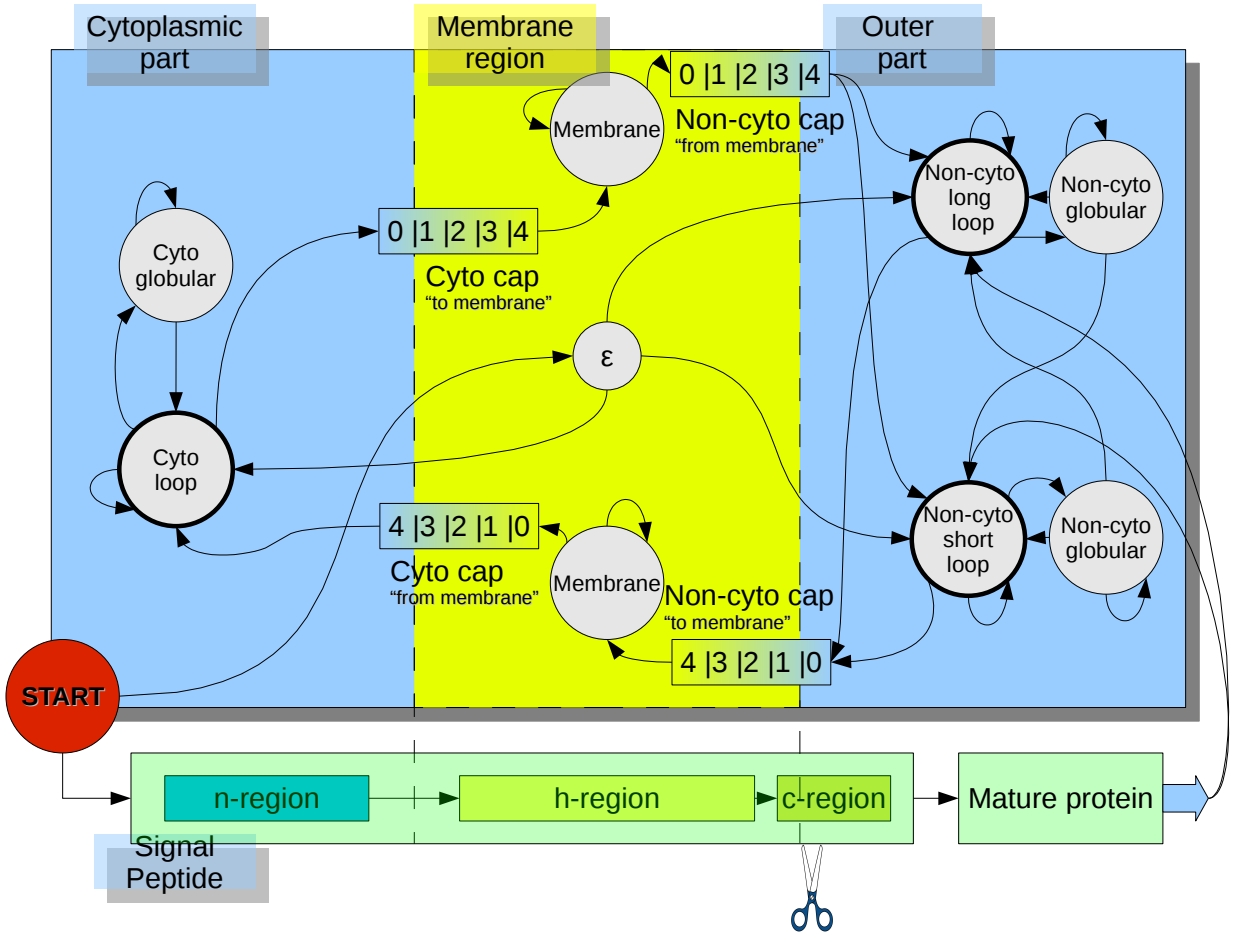


Figure 5.1: **The architecture of the developed transmembrane predictor** The "cyto" HMM states indicate those representing globular or loop protein regions on the cytoplasmic side, while the "non-cyto" ones correspond to regions lying outside of the cell. Globular and loop regions are represented in different states, to better represent these structures by allowing different length distributions and amino acid distribution for each. The model can either follow the path of a Signal Peptide (SP) or the alternative route where it may pass through transmembrane regions. ε represents the "hub state" which does not emit any symbol but connects certain states. Cap columns are shown in the direction the HMM moves (0 to 4, or 4 to 0), and "to membrane" caps correspond to the positions shown in Tables 5.1 and 5.2, where "0" lies outside of the membrane. The system can terminate while being in any of the states depicted in thicker circles.

regions, for instance, symbol emission distributions at different positions of the cleavage site vary significantly (see Section 3.3.1 for the "-3 -1" rule).

5.2.1.1 Representing helix caps in the HMM

Amino acid	POS 0	POS 1	POS 2	POS 3	POS 4
TRP	0.012	0.034	0.040	0.026	0.030
SER	0.059	0.053	0.045	0.049	0.068
ASN	0.056	0.020	0.018	0.018	0.019
ALA	0.049	0.101	0.121	0.101	0.094
HIS	0.029	0.009	0.014	0.015	0.009
TYR	0.034	0.056	0.037	0.039	0.050
PHE	0.038	0.074	0.078	0.082	0.094
ARG	0.148	0.021	0.020	0.020	0.016
LEU	0.053	0.187	0.181	0.163	0.147
PRO	0.035	0.032	0.029	0.037	0.025
MET	0.015	0.052	0.044	0.046	0.040
GLU	0.041	0.011	0.016	0.011	0.011
ILE	0.025	0.098	0.099	0.113	0.114
VAL	0.041	0.104	0.097	0.097	0.104
GLY	0.070	0.059	0.065	0.074	0.077
LYS	0.136	0.016	0.017	0.018	0.014
GLN	0.031	0.010	0.016	0.016	0.019
THR	0.051	0.045	0.044	0.047	0.049
ASP	0.059	0.010	0.010	0.011	0.002
CYS	0.018	0.008	0.010	0.017	0.018

Table 5.1: **Amino acid emission probabilities in the transmembrane helix cytoplasmic side cap.** POS 0 refers to the residue which falls in the cytoplasmic part, while positions 1 to 4 are within the helix. POS0 also corresponds to position-0 of the top and position-4 of the bottom "cyto" caps shown in Figure 5.1.

Transmembrane helices show different amino acid propensities in different positions, too, although these are not so obvious and consistent to be represented as constant motifs. It has been suggested (Jones *et al.*, 1994; Richardson & Richardson, 1988) that middle regions, parts closer to the cytoplasmic loops, and parts

Amino acid	POS 0	POS 1	POS 2	POS 3	POS 4
TRP	0.022	0.039	0.045	0.040	0.038
SER	0.079	0.040	0.059	0.062	0.054
ASN	0.053	0.026	0.024	0.023	0.015
ALA	0.053	0.103	0.099	0.087	0.110
HIS	0.031	0.016	0.015	0.018	0.015
TYR	0.039	0.051	0.048	0.048	0.050
PHE	0.033	0.096	0.090	0.093	0.077
ARG	0.069	0.014	0.011	0.010	0.012
LEU	0.063	0.162	0.156	0.145	0.164
PRO	0.051	0.046	0.041	0.034	0.028
MET	0.025	0.035	0.039	0.043	0.035
GLU	0.073	0.015	0.015	0.011	0.005
ILE	0.028	0.096	0.084	0.100	0.105
VAL	0.043	0.099	0.086	0.106	0.109
GLY	0.082	0.068	0.096	0.083	0.093
LYS	0.062	0.009	0.005	0.008	0.006
GLN	0.047	0.018	0.019	0.015	0.023
THR	0.062	0.046	0.046	0.051	0.041
ASP	0.073	0.017	0.013	0.011	0.008
CYS	0.013	0.005	0.009	0.010	0.011

Table 5.2: **Amino acid emission probabilities in the transmembrane helix non-cytoplasmic side cap.** POS 0 refers to the non-cytoplasmic residue position (outer cell), while positions 1 to 4 are within the helix. POS0 also corresponds to position-4 of the top and position-0 of the bottom "non-cyto" caps shown in Figure 5.1.

near the non-cytoplasmic regions of membrane spanning helices feature significantly different amino acid composition. Consistent with this idea, I calculated amino acid emission probabilities for helix middle regions and helix "cap" regions independently. I extended both helix cap regions to overhang outside of the helix by one residue, where a total of 5 amino acid positions are considered.

If the symbol "i" represents the "inner" cytoplasmic part, "M" the membrane, and "o" the outer, non-cytoplasmic region residues, then one can write the possible two configurations that alpha helices can be in, in terms of these letters, as follows:

1. ...iiiMMMMM...MMMMooo...

2. ...oooMMMMM...MMMMiii...

Residue	KL	Residue	KL
TYR	-0.0064	ASP	-0.01727
MET	-0.0110	ASP	-0.01727
LEU	-0.0134	PRO	-0.01922
ILE	-0.0043	TRP	-0.01052
GLN	-0.0188	CYS	0.00751
PHE	0.0080	SER	-0.02420
THR	-0.0142	HIS	-0.00233
ALA	-0.0056	GLU	-0.03438
GLY	-0.0157	**ARG**	**0.16262**
VAL	-0.0024	**LYS**	**0.15494**

Table 5.3: **KL deviations of cytoplasmic side helix termini from the noncytoplasmic side helix termini in relative amino acid abundance rates.** Amongst the other amino acids, Arginine (ARG) and Lysine (LYS) differ the most in terms of their relative KL deviations between the two first non-helical positions on both sides (see text). That is, the first non-helix position of a helix cap on the cytoplasmic side is more enriched in ARG and LYS than the first non-helix position on the other side of the membrane.

The so-called "cyto cap" of Figure 5.1 corresponds to the left-hand side of the first helix topology, and the right-hand side of the second. Similarly, the "non-cyto cap" regions overlap where M is either preceded or followed by "o". Each position in the caps was represented as an independent state as in a profile HMM, and therefore, for each position a separate probability distribution was calculated, rather than using a single, general distribution for the entire cap lengths. This was done after reversing the order of residues in the second type topology. Tables 5.1 and 5.2 summarise the amino acid emission probabilities in the helix caps near the cytoplasmic side, and non cytoplasmic side, respectively. Probability distributions for amino acid position 0 in the tables indicate the first amino acid residue outside of the helix on either side. A good statistical measure to evaluate the difference between two distributions is to use Kullback Leibler (KL) deviation which measures the relative entropy between two distributions:

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{5.1}$$

where P is the "true" distribution from which we measure how another distribution "Q" deviates in terms of relative entropy. KL deviation is not a proper distance measure as it is not symmetric, hence it is called a deviation, although sometimes it may be referred to as "KL distance". Its value is always positive, but some of the individual terms summed up in the above equation could be negative, due to the nature of the logarithmic part.

Amino acid probability distribution of the cytoplasmic cap at position 0 (Table 5.1) deviates from the non-cytoplasmic cap's position zero distribution (5.2) by a

KL value of 0.137 (as computed in logarithm base 2). This number does not make much sense alone, but from the individual values contributing to the sum, which are given in Table 5.3, one can easily spot the differences from the relative values computed for the same pair of amino acid symbols. From Table 5.3 it can be easily noticed that the primary difference between the two distributions is due to the differing abundance rates of the Arginine (ARG or R) and Lysine (LYS or K) residues. That is, we can safely conclude that the first amino acid position outside of the helix that points to the cytoplasmic region is more enriched in ARG and LYS than the first non-cytoplasmic position at the other end of the membrane, protruding into the extracellular space. This is also evident by directly comparing the raw probabilities of both distributions.

The other cap positions (1 to 4) obviously differ from the rest of the sequence positions in that they tend to be rich in hydrophobic residues. However, comparing the corresponding positions in the cytoplasmic and non-cytoplasmic cap regions in terms of KL deviation, we see that they are more similar to each other. While position zero distributions of the caps deviate by a KL of 0.137 as mentioned above, the KL deviations (specifically, of the cyto-cap distributions from the non-cyto cap ones) for the other positions are:

1. 0.037
2. 0.044
3. 0.030
4. 0.037

Appendix D lists the individual KL distances between all corresponding pairs of amino acid distributions within the cytoplasmic side, within the non-cytoplasmic side, and finally amongst cytoplasmic and non-cytoplasmic helix cap positions. The computed KL deviations implied that within the same helix cap, there is not much difference among the different cap positions, except for when compared with the ones in the edges. For example, the KL deviation for the distributions at non-cytoplasmic side helix cap positions 3 and 4 is 0.013, which is pretty small. For this reason, it may not be necessary to use separate distributions for such similar two positions; however, this does not introduce any extra complication to the algorithm in optimising their transition probabilities in the HMM, as they have a fixed probability of moving to the adjacent state of 1. Furthermore, any small distribution difference across the same position of different caps can be more valuable than the possible differences within a particular cap's positions in figuring out the correct overall state path. That is why all cap positions were modeled independently, while this does not increase the burden of the state path optimiser.

5.2.1.2 Duration HMM states

Even though it will not be biologically possible to have a transmembrane helix of, say length of 3aa, HMMs which are probabilistic methods, can generate predicted state labels corresponding to biologically unfeasible lengths no matter how high the self-transition probabilities of the associated states are. This can be prevented by setting a "minimum number of self-visits" to the problematic states. As mentioned earlier in the Introduction chapter (page 15), such HMMs are referred to

as "duration HMMs". Unfortunately, the current version of Biojava (BioJava, 2007), the Java libraries collection which was used to implement this prediction system, does not have such a duration concept. However, I implemented a duration HMM package that allows users to create Markov models having states with a certain number of minimum self-transitions. The probability of staying in a certain "duration" state remains constant while the number of transitions are less than the user-set minimum number, and then goes into an exponential decay in accordance with its classical self-transition probability (see Figure 1.2).

The states representing the helical regions (see Figure 5.1) were not allowed to be any shorter than 6 amino acids. This makes the minimum allowed length of a transmembrane alpha helix 14 amino acids, when both cap regions are considered (in Phobius this was taken as 15). Note that, although the cap regions were of length 5aa each, only 4 positions are spanning helices in the designed model. Minimum durations for all the states of the HMM are given in Table 5.4.

5.2.2 Datasets and training of the model

For training the emission probabilities of the HMM states, I used the same sequence sets used in Phobius. These sequences are provided in labelled fasta format (Krogh, 2002) to indicate what type of region (i.e. α-helix, n-region of signal peptide, loop protruding towards the extracellular space etc.) they fall into. Table 5.5 lists the number of sequences from each category of sequences. As the table shows, the HMM was trained using sequences that include both transmembrane regions and signal peptides, sequences having only one type of these features, and finally sequences not having any of these structures. HMM state

HMM State	Minimum duration
n-region of SPs	6
h-region of SPs	6
c-region of SPs	5 (including cleavage site)
Non-cytoplasmic long loop	15
Non-cytoplasmic short loop	1
Cytoplasmic loop	1
Globular (cytoplasmic)	15
Globular (non-cytoplasmic)	15
Transmembrane alpha helices	6 (14, with both caps)

Table 5.4: **Minimum allowed emission state occupancy numbers for the transmembrane topology predicting HMM.** The n, h, and c regions refer to the sub-regions in N-terminal Signal Peptides (SP). The c-region includes the cleavage site which is modeled as a profile HMM, based on a NestedMICA motif for this region. Transmembrane alpha helices can hardly be shorter than 14, so the corresponding helical states in the HMM allow a minimum of stay of 6 emissions in these states, which is then added with durations of the N- and C-terminal helix caps regions to yield a total length of 14 amino acids.

emission probabilities were determined from the amino acid frequency distributions of only the corresponding sequence segments. That is, the "membrane" states shown in Figure 5.1, for example, have amino acid emission probabilities calculated only from the transmembrane segments of sequences featuring those regions.

Type of sequence	Number of sequences
Transmembrane proteins with signal peptides	45
Transmembrane proteins without signal peptides	247
Non-transmembrane proteins with signal peptides	1773
Non-transmembrane proteins without signal peptides	1520

Table 5.5: **Sequences used in the training of Phobius and of the program developed.**

The HMM was trained using 10-fold cross validation, with the new HMM tran-

sition probability optimisation procedure that is introduced below. The dataset (Table 5.5) was divided into 10 equal portions having more or less the same number of sequences from each type, 9 of which were used in the training while the singled out one was used for testing performance.

5.2.3 Transition probability optimisation: a new approach

The optimisation of state transition probabilities was first performed using the standard Biojava (BioJava, 2007) implementation of the Baum-Welch algorithm. This semi-supervised learning method tries to find the best model parameters by maximising the likelihood of the state path, given a set of "emittable" observables and their emission probabilities for each state. It can be considered as semi-supervised, because it tries to optimise the transition probabilities without using the true state labels of a given training data. If the number of parameters to optimise is too large, this method may not produce a good set of parameters at all, particularly for problems where different emission states have similar emission probability distributions. Also, it is often the case during any automatic learning that overfitting of the model can occur, which makes determining the number of iterations in the Baum-Welch algorithm somewhat tricky. Of course, another laborious approach would be to set the model parameters empirically, judging according to what set of parameters maximises performance at the end, where performance would be the number of observables correctly assigned into the associated state label. However, this trial-and-error approach would not be practical for problems involving many emission states.

Here I introduce a new approach to optimise HMM transition probabilities.

It is based on finding a set of state transition probabilities learnt from a given training set with known state labels, by using a probabilistic, generative sampling strategy. "Labels" correspond to sequence annotations such as transmembrane helix, cytoplasmic loop, globular region etc. for each amino acid position. At each iteration of the method, a different set of transition probabilities is tried, and a likelihood score corresponding to the fraction of correctly labelled amino acids in the training dataset is calculated.

This procedure is analogous to other Monte Carlo techniques in that we either accept or reject a proposed set of probabilities, but in order to eliminate the possibility of getting stuck in some local maxima, I use Nested Sampling (see Section 2.2.1 on page 28) that keeps an ensemble of fixed number of proposals, or "probability vectors". Because we sample typically hundreds of different vectors, this ensures finding a globally optimal solution given enough time and a good likelihood function. In the initialisation step, a likelihood score for each vector in the ensemble is calculated. Transition probabilities of HMM states having a single transition are automatically set to a value of 1.0, and these states are omitted in the rest of the parameter optimisation.

Each step in this algorithm starts with the search for the vector generating the least likelihood score. This "worst" vector is removed from the ensemble, and replaced with a newly sampled one according to the following two rules:

1. The new probability vector is generated by modifying the probabilities of the removed vector.

2. The new vector has to have a likelihood score that is greater than that of

the removed. Another random vector is sampled until this condition is met, or the algorithm is terminated under certain termination criteria.

Once an appropriate move is found the ensemble is updated. This ensures that after each step we move into a better set of solutions, and that the total likelihood increases after each accepted step as illustrated in the cumulative likelihood plot in Figure 5.2. Figure 5.3 shows the likelihood curves of the worst and the replacement states after each move, in an example optimisation problem. The overall likelihood continues to get better monotonically until the system converges or termination occurs as determined by some stopping criteria. Convergence slows down when it becomes harder to find "better" moves than those in the current ensemble (Figure 5.4).

In the sampling process, it is necessary to choose a good, relevant likelihood function that will reflect the fact that we are optimising for the number of correct labels in an HMM state path. To this end, I used a simple likelihood function calculated in the log-space:

$$L(m) = \sum_{i=1}^{N} \log \left(\frac{1}{|S_i|} \sum_{j=1}^{|S_i|} k_j^i \right) \tag{5.2}$$

where $|S_i|$ is the length of sequence i, N is the total number of sequences, and k_j^i is a unit function that is non-zero if the j^{th} amino acid position of the i^{th} sequence is correctly identified by the evaluated Markov model m. The total likelihood function is the sum of accepted model likelihoods that constitute the ensemble. It is this cumulative likelihood function that is ensured to increase at each step before accepting a proposed Monte Carlo move. Similarly, the least

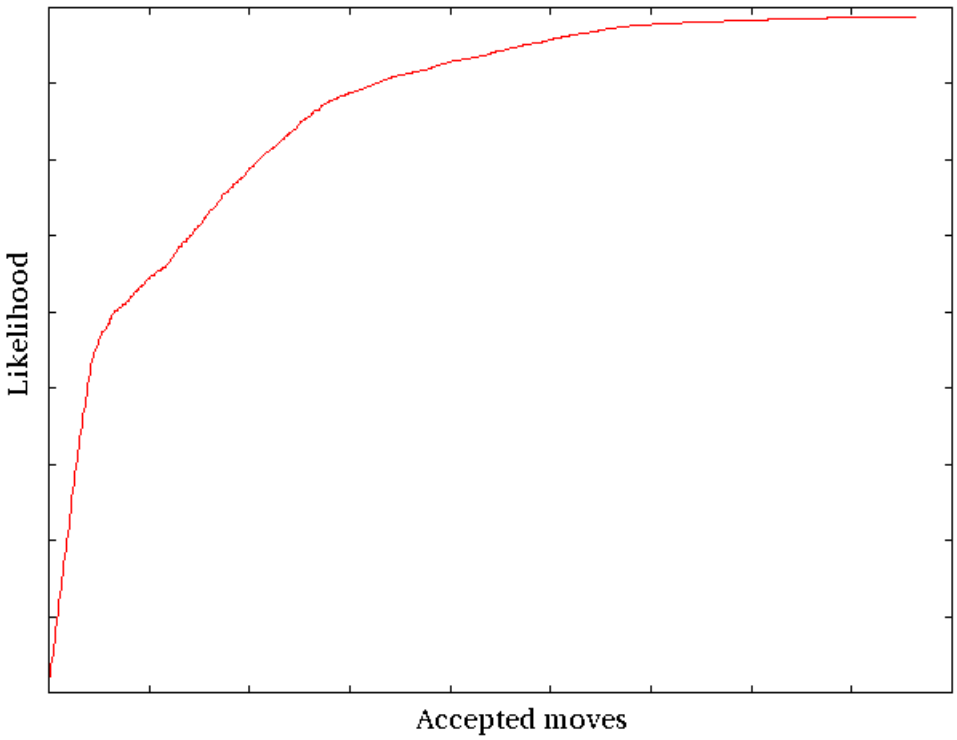


Figure 5.2: **Total likelihood function monotonically increases in nested sampling.** This plot illustrates how the likelihood function (y-axis) used in the developed HMM transition probabilities optimisation technique varies over accepted steps (x-axis).

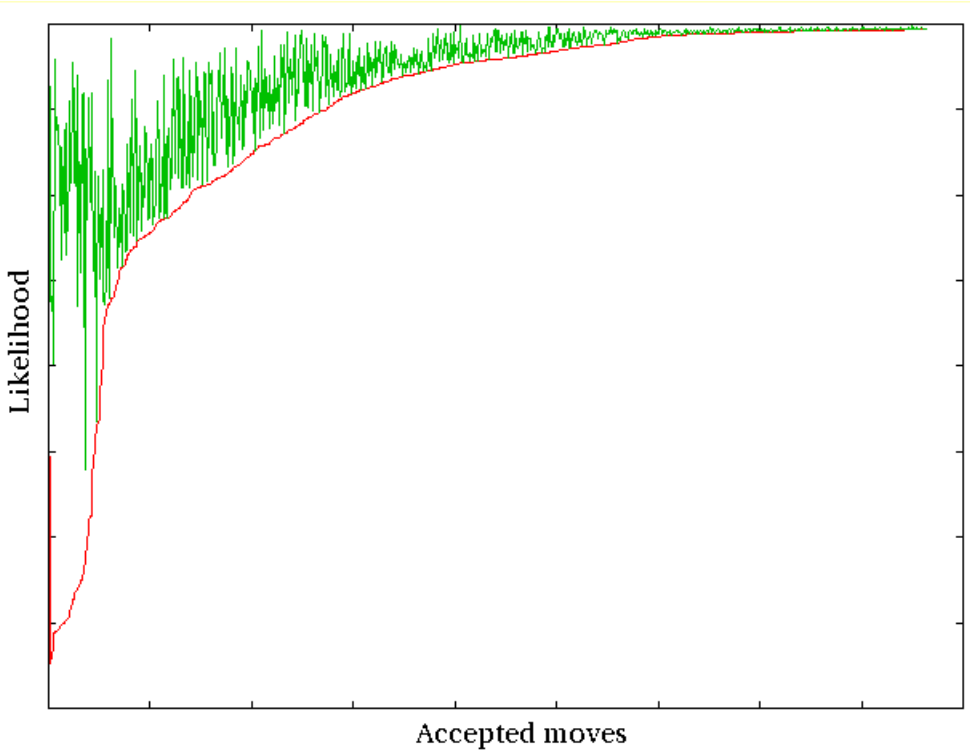


Figure 5.3: **Transition probability set having the least likelihood is replaced by new better one.** A suggested move has to have a likelihood that is larger than the "worst" vector's likelihood to be accepted. The red line represents the least likely probability vector of the ensemble at a particular step. It is replaced by a "better" vector, shown in green colour.

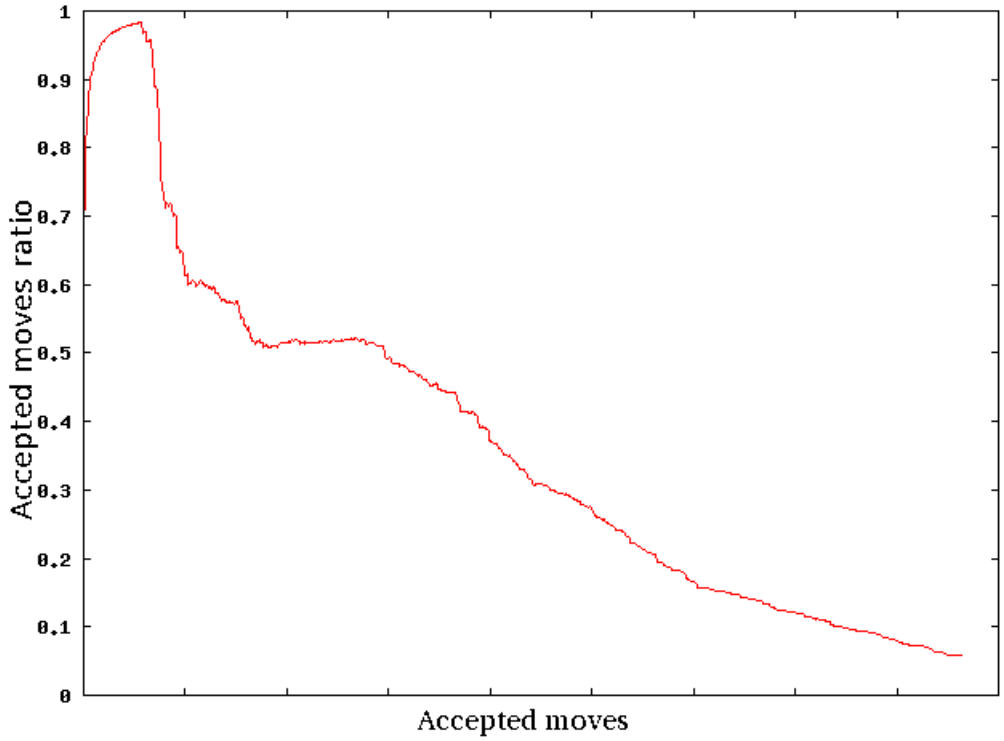


Figure 5.4: **It becomes more difficult over time to find "acceptable" states.** The curve shows the ratio of accepted moves to the total number of proposed moves, which is updated after each accepted move. This accepted/(accepted+rejected) ratio helps to determine whether an optimisation run is close to convergence or not.

likely state's score L_w at each step will increase in parallel with the improving overall goodness of the ensemble.

Increasing the ensemble size may result in better parameter sets. However, the number of sampled vectors in the ensemble hugely affects the program speed and convergence rate. At the end, I observed that setting the ensemble size to around 10 times the number of HMM states being optimised in a particular problem proved to be sufficient for typical HMM transition probability optimisation problems.

The individual probabilities forming the vectors are sampled according to Gaussian functions associated with each of the probabilities. The sampling is carried out using a Gaussian with a mean that is equal to the previous probability value. The main sampled entity is, of course, probability distributions of HMM states. This is achieved through multiple ways:

- All state probabilities at the same time, using a large Gaussian variance
- All state probabilities at the same time, using a small Gaussian variance
- A randomly picked state's probabilities, using a large Gaussian variance
- A randomly picked state's probabilities, using a small Gaussian variance
- All state probabilities at the same time, using a Gaussian variance dynamically updated according to the number of rejected moves
- A randomly picked state's probabilities, using a Gaussian variance dynamically updated according to the number of rejected moves

- All state probabilities at the same time, using a random Gaussian variance
- A Baum-Welch iteration of a model created based on the probabilities of a randomly picked state

That is, the distributions are perturbed by using uniform Gaussian distributions with either pre-determined or dynamically updated standard deviations (see below for how this is achieved). The mean of each Gaussian is equalised to the previous selected probability value of a particular state transition. Two standard deviations, one small and one large, are used. The one that will be utilised in a particular step is determined by a random selection process. These variance values can be later dynamically updated based on the number of accepted and rejected proposals. This reduces the number of rejected proposals, and also allows the system to explore different sets of solutions as much as possible. As the above list shows, sampling could be applied on all states of an HMM simultaneously, or by working on a single distribution in each proposed step. Another possibility is to select a state randomly and then change its transition weights according to a randomly picked variance value around the previous values of the probabilities.

Finally, a Baum-Welch move can be proposed, based on a certain probability (for instance, 15% of the proposed steps are based on moves proposed by the Baum Welch algorithm in the current implementation). In a Baum-Welch move, the actual Markov model is updated with the transition weights of a randomly chosen item from the ensemble. A certain number of Baum-Welch iterations are run on the model characterised by the selected transition weights set, as in a classical transition probability optimisation procedure. If this new model

increases our likelihood function that measures the number of correctly assigned labels, then this move is accepted, and the ensemble is updated with the transition set that the Baum-Welch algorithm fine-tuned.

In the case of rejected Baum-Welch proposals for the same worst state, depending on the number of recent rejections, the iteration number of the Baum-Welch training is increased. This iteration number is randomly chosen from numbers up to a maximum value equal to the number of local rejections, provided that it does not exceed an empirically set 20 iterations per move.

Whenever needed, the standard deviations that are allowed to change are dynamically altered, as stated above (some remain fixed during the entire optimisation). This is performed by multiplying or dividing the previous standard deviation value by $e^{1.0/rejected}$, making sure it will be in the interval (0,1). This way, when there is room for large gains in the total likelihood, this is achieved faster by using a larger standard deviation, and when the system begins to reject more and more proposals, the probability distributions are less perturbed when sampling from the ensemble.

Individual transition probabilities for each state are sampled from their Gaussians by using the "online" standard deviation divided by the number of total transitions a state possesses. If a negative value is obtained from a particular sampling of a state, all other transition probabilities in that state are shifted to the positive side by an amount equal to the minimum probability obtained (plus some very small number, to avoid absolute zero probabilities). The values sampled from the Gaussians in each step are then re-normalised to obtain sensible probabilities that will sum up to 1 for each state.

5.3 Results

Tables 5.6 and 5.7 compare transmembrane (TM) predictors Phobius and the developed prototype program, in terms of their performance in predicting TM topology for i) proteins having both TM and signal peptides (SP), ii) proteins having only TM helices, iii) proteins having only SPs, and finally, iv) proteins having neither a TM nor an SP. For a prediction to be counted as correct, all annotated individual TM helices and loop regions must be predicted correctly. An overlap of at least 5 residues was considered a "correct" prediction for each helix, as done in the CASP competitions (Cozzetto *et al.*, 2005, 2007; Soro & Tramontano, 2005; Valencia, 2005), or as in other studies reporting TM accuracies (Jones, 2007; Käll *et al.*, 2004). In the presence of a signal peptide (SP), whether a program predicted the SP or not in a particular protein was not taken into account in determining the overall correct TM topology.

SP prediction performance is evaluated separately, and the results for predictors that are capable of detecting SPs are summarised in Table 5.7. The developed predictor was compared with two versions of SignalP, in addition to Phobius which is both a TM and SP predictor. TMHMM is not designed to predict SPs directly.

The results generally suggest that the developed HMM program is better in predicting SPs than the other compared programs, although its architecture is quite similar to that of Phobius. On the other hand, the prototype program performs relatively badly in correct transmembrane (TM) topology prediction, although it performs reasonably well in predicting individual TM helices.

	This predictor	Phobius	TMHMM2.0
TM and SP proteins			
Correct topology	66.7%	91.1%	71.1%
Correct TMs	76.1%		
Correct SPs	88.9%		
False positive TMs	18.1%		
TM-only proteins			
Correct topology	36.0%	63.6%	65.2%
Correct TMs	76.1%		
False positives	3.6%	7.7%	
Non-SP and non-TM proteins			
Correct topology	99.87%	98.2%	98.7%

Table 5.6: **Prediction performance summary for "TM-and-SP", "TM-only" and "non-SP, non-TM" proteins.** A prediction was taken as correct when all the predicted Transmembrane (TM) helices overlap all the annotated TM helices of the protein over at least 5 amino acids, and when the loops were correct. In the evaluation of predictions for proteins having no TM helices, the reported "correct topology" corresponds to not having any predicted TM helices for that protein. Incorrect signal peptides (SPs) were not considered in determining correct topology prediction rates. The available prediction rates for Phobius and TMHMM2.0 were taken as reported in Käll *et al.* (2004), where TMHMM results were not cross validated.

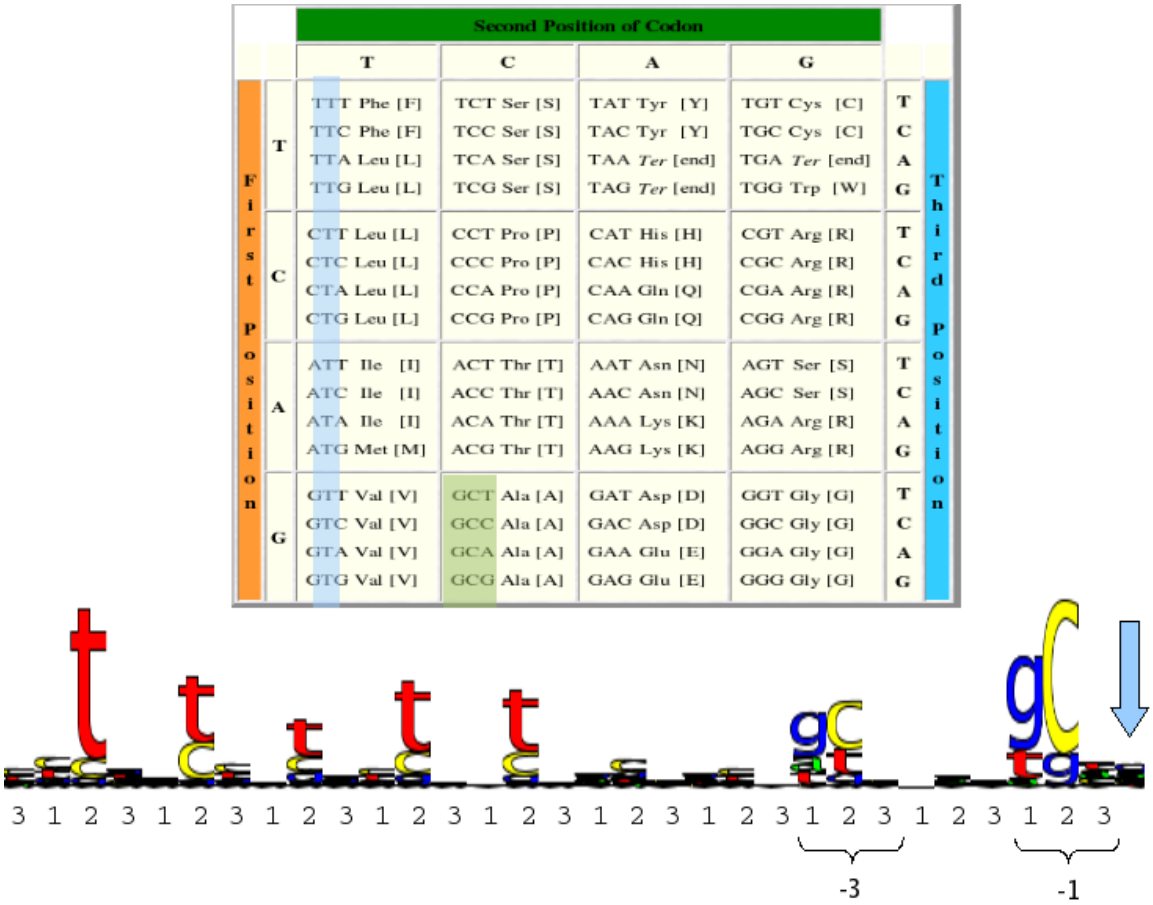
	This predictor	Phobius	SignalP-NN	SignalP-HMM
Correct SPs	89.8%	96.5%	97.7%	98.6%

Table 5.7: **Correct prediction rates for SP-only proteins.** Correct SP prediction rates are shown for the program developed, Phobius, Neural network version of SignalP (Nielsen *et al.*, 1997a) and the HMM version of SignalP (Nielsen & Krogh, 1998). The available prediction rates for Phobius and the two version of SignalP were taken as reported in Käll *et al.* (2004), where SignalP results were not cross validated.

Using a duration-enabled HMM which was trained by the new optimisation procedure improved the results dramatically. When no minimum-durations were set in the HMM model, the correctly predicted TM helices rate, for example, decreased to 12.5% from 76.1% for the TM and SP containing protein dataset, while it decreased to 19.6% from the same initial value of 71.6% for the TM-only proteins. On the other hand, optimising transition probabilities by Baum-Welch without using any initial "clever guesses" resulted in very poor TM helix prediction accuracies (<1%), after letting it to run for about a dozen, 50 and finally a few hundreds of cycles, even when it was trained using the entire protein set.

5.3.1 Signal peptides at the DNA level

As mentioned in Section 3.1.1, most of the signal peptides are at least 20 aa long. Inspecting the available annotated protein sequences having signal peptides, we observed that this signal could stretch out to as long as 50 aa. One question that arises for such long N-terminal sequence patterns is how their untranslated form might look at the DNA level, given that their corresponding mRNAs would be three times longer than the amino acid signal. Because NestedMICA is a DNA discovery tool at the same time, using the motif finder in the DNA mode, I had a chance to investigate how conserved the hydrophobic regions are at the genomic level. Interestingly, as Figure 5.5 shows, only certain residues in the codons that translate into this signal are conserved. The hydrophobic part of the signal is generated from codons having a conserved second position which is usually a "T", while the cleavage site, having small residues like Alanine (A), Glycine (G)



First position	Second Position of Codon: T	C	A	G	Third position
T	TTT Phe [F]	TCT Ser [S]	TAT Tyr [Y]	TGT Cys [C]	T
	TTC Phe [F]	TCC Ser [S]	TAC Tyr [Y]	TGC Cys [C]	C
	TTA Leu [L]	TCA Ser [S]	TAA *Ter* [end]	TGA *Ter* [end]	A
	TTG Leu [L]	TCG Ser [S]	TAG *Ter* [end]	TGG Trp [W]	G
C	CTT Leu [L]	CCT Pro [P]	CAT His [H]	CGT Arg [R]	T
	CTC Leu [L]	CCC Pro [P]	CAC His [H]	CGC Arg [R]	C
	CTA Leu [L]	CCA Pro [P]	CAA Gln [Q]	CGA Arg [R]	A
	CTG Leu [L]	CCG Pro [P]	CAG Gln [Q]	CGG Arg [R]	G
A	ATT Ile [I]	ACT Thr [T]	AAT Asn [N]	AGT Ser [S]	T
	ATC Ile [I]	ACC Thr [T]	AAC Asn [N]	AGC Ser [S]	C
	ATA Ile [I]	ACA Thr [T]	AAA Lys [K]	AGA Arg [R]	A
	ATG Met [M]	ACG Thr [T]	AAG Lys [K]	AGG Arg [R]	G
G	GTT Val [V]	GCT Ala [A]	GAT Asp [D]	GGT Gly [G]	T
	GTC Val [V]	GCC Ala [A]	GAC Asp [D]	GGC Gly [G]	C
	GTA Val [V]	GCA Ala [A]	GAA Glu [E]	GGA Gly [G]	A
	GTG Val [V]	GCG Ala [A]	GAG Glu [E]	GGG Gly [G]	G

Figure 5.5: **Signal peptide motif at the RNA level.** Conservation of only certain positions in the codons corresponding to the SP signal suffices to generate hydrophobic amino acid chains, as the genetic code table above illustrates having a 'T' at the middle of a codon usually generates hydrophobic amino acid residues. The sequence logo shows 10 residues before the actual cleavage residue position which is indicated by an arrow. Codons beginning with the "GC" dinucleotides correspond to the small amino acids of the "-3 and -1 positions " rule (see text), with respect to the cleave site. Numbers right below the sequence logo correspond to the actual positions of amino acids within the codons in the associated correct reading frame.

etc in the protein level, is dominated by Guanine and Cytosine at positions 3 and

1, respectively, at the DNA level.

5.4 Discussions

In this chapter, I introduced a new, fully supervised methodology for training HMM transition probabilities and applied it on TM topology prediction as an example. While this method is open to further improvements and also to some more testing, the essential idea behind it, the use of Monte Carlo strategies to fine tune transitions, seems to be a promising approach. These kind of optimisation problems normally involve huge parameter landscapes where each optimised parameter can take any probability value. Another hurdle in this type of search heuristics is that, in principle it is not uncommon for a sampled property to move towards some local maxima and get stuck there. However, with the use of nested sampling, a fruitful strategy that has proven itself in biological sequence motif discovery before, such possibilities can be avoided. With this approach multiple possible solutions representing different maxima from the entire probability landscape are considered at the same time, instead of trying to make a single sample better during the entire process. This is simply to eliminate the greediness of sampling at each step that could possibly miss the real solution set at the end, had it not moved to the locally "best"condition in previous steps. Removal of the "weakest" state having the least likelihood probability from a large population, and re-sampling from the "fitter" entities to replace the worst, is conceptually nothing but a genetic algorithm way of optimising, with the exception that the space is continuous here – entities are not simply of type that either exist or not.

Transmembrane topology prediction is a well established field for many years now, and there are many good TM topology prediction programs. However, due

to the similarities between N-terminal TM helices and SPs (see Sections 3.1.1 and 3.3.1), they tend to misclassify SPs as TMs and vice versa. Programs such as Phobius have recently reduced this by using a combinatorial approach where SPs and TMs are predicted in the same model. This resulted in a reduced number of false positive predictions and cross-misclassifications (Käll *et al.*, 2007).

As mentioned in the methods, this prototype program differs from Phobius in that in the HMM I directly used a cleavage site motif that was discovered by NestedMICA, and I use a new optimal transition probability estimisation method. A relatively low correct topology prediction with respect to the correctly identified TM helices indicates that the program tends to invert the orientations of the topologies it predicts. Apart from the difficulties inherent in the biology of the problem, this could possibly be due to immature termination of the transition probability optimisation procedure, which has not been fully optimised for termination criteria yet.

Both the parameter optimisation approach and the prototype TM topology predictor that was developed to demonstrate that this approach can actually be used successfully are promising. The optimisation method is open to further development, which, in turn, can significantly improve the TM predictor's performance.