Statistical Methods in Applied Computer Science

Magic Word

Author
Salma El Alaoui Talibi

Professor
Jens Lagergren

January 2016

1. Introduction

The magic word model generates N sequences of length M, $s^1, \ldots, s^N$, where $s^n = s^n_1, \ldots, s^n_M$, where all the sequences are over an alphabet $[K]$. Each sequence has a magic word of length W hidden in it, while the rest of the sequence is called background. Our goal is to find $R = r_1, \ldots, r_N$, where r_n is the start position of the magic word in the n:th sequence s^n.

An interesting application of this model can be found in biosequence analysis. The alphabet that we consider is the genomic alphabet: $K = \{A, C, G, T\}$. The sequences $s^1, \ldots, s^N$ become a set of aligned DNA sequences and the positions $[1 \ldots M]$ are the DNA columns which represent locations along the genome. Our problem can therefore be reformulated as finding the unknown magic word that appears at different unknown starting positions in those sequences.

In order to do that, we will implement a collapsed Gibbs sampler to sample from the posterior $p(r_1, \ldots, r_N|D)$ where D is the set of DNA sequences generated by the model.

2. Generation

The magic word generative model proceeds as follow:

- For each sequence s^n in $s^1 \dots s^N$, we sample a start position $r_n \sim U(M - W + 1)$
- Then for each sequence, we sample the letters in the positions j in the magic word :

  $$x_j^n \sim Cat(\theta_j)$$

  Where $\theta_j \sim Dir(\alpha)$ and $j \in [r_n, r_n + W - 1]$
- Finally, we sample the letters in the background positions for all sequences:

  $$x^n \sim Cat(\theta) \quad \text{Where} \quad \theta \sim Dir(\alpha')$$

Since we have no prior specific knowledge about the background of the DNA sequences, we will use a uniform prior for θ with parameter $\alpha' = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}$. All sequences will be generated from the same categorical distribution. For the magic words in the sequences however, the j:th positions are generated from distinct categorical distributions parameters θ_j having the same prior $Dir(\alpha)$, which complicates accurate inference. We generate set of 5 sequences ($N = 5$) and 30 columns ($M = 10$), with magic words of length 10 ($W = 5$), with $\alpha' = (1, 1, 1, 1)$ and $\alpha = (12, 7, 3, 1)$. The start positions are highlighted:

$$\begin{array}{cccccccccc}
A & T & T & A & \boxed{C} & G & C & A & A & T \\
T & A & T & T & \boxed{A} & A & C & G & A & C \\
\boxed{C} & G & C & C & A & C & A & C & C & A \\
C & G & C & T & A & \boxed{A} & A & G & A & A \\
A & T & A & A & T & \boxed{A} & G & A & G & A
\end{array}$$

3. Gibbs Sampler

We use a Gibbs sampler to estimate the posterior $p(r_1, \ldots, r_N|D)$ where D is the set of sequences $s^1, \ldots, s^N$ generated by the magic word model and r_n is the start position of the magic word in the n:th sequence s^n.

We use a *collapsed* Gibbs sampler, which means collapse out the Dirichlet distributions(the prior distributions over the categorical variables). The result of this collapsing introduces dependencies among all the categorical variables dependent on a given Dirichlet prior.

Each state $R^{(s)}$ of a the Markov chain has the following form : $R^{(s)} = r_1^{(s)} \ldots r_N^{(s)}$.

The first T samples (burn-in period) are discarded to allow for the Markov chain to reach stationarity. We also collect subsequent samples after a lag to ensure their correlation is low.

The implementation is as follows:

```
Random initialization $R^{(0)}$
for i from 0 to iterations do
    for n from 0 to N do
        $P(r_n^{(i)}|R_{-n}^{(i)}, D) \propto \cup_{l=0}^{M-W-1} p(D_{background}|R^{(i)}, \alpha') \prod_{j=l}^{l+W} p(D_j|R^{(i)}, \alpha)$
        normalize the above vector of probabilities
        $r_n^{(i)} \sim Cat(P(r_n^{(i)}|R_{-n}^{(i)}, D))$    ▷ $r_n^{(i)}$ is sampled from the categorical with parameter $P(r_n^{(i)}|R_{-n}^{(i)}, D)$, which is the vector of posterior probabilities for values of $r_n^{(i)}$ in $[0, M - W - 1]$
    end for
end for
for n from 0 to N do
    samples = samples[i]    ▷ samples is the vector containing the Markov chain states (samples) that were collected
    $r_n = argmax(count_{t=T:k:iterations}(R_n^{(t)}))$    ▷ For each sequence, the starting position is the mode of the posterior, i.e. the position that was sampled most frequently . The $T$ first samples are discarded and we only consider every $k$:th iteration in the samples, where $k > 1$.
```

```
12: end for
```

With :

$$p(D_{background}|R^{(i)},\alpha') = \frac{\Gamma(\sum_k \alpha'_k)}{\Gamma(B\sum_k \alpha'_k)} \prod_k \frac{\Gamma(B_k+\alpha'_k)}{\Gamma(\alpha'_k)} \tag{3.1}$$

where $B = N(M-W))$ is the number of background positions and B_k is the count of symbol k in the background.

$$p(D_j|R^{(i)},\alpha) = \frac{\Gamma(\sum_k \alpha_k)}{\Gamma(N\sum_k \alpha_k)} \prod_k \frac{\Gamma(N_k^j+\alpha_k)}{\Gamma(\alpha_k)} \tag{3.2}$$

where N_k^j is the count of symbol k in the j:th column of the magic word.

4. Results

We consider a model with 5 sequences ($N = 5$) and 30 columns ($M = 30$), with magic words of length 10 ($W = 10$).

4.1 Convergence

We first want to estimate the number of iterations that are necessary for the Markov chain to converge to the target distribution. We run the sampler for 200 iterations, with $\alpha' = (1, 1, 1, 1)$ and $\alpha = (12, 7, 3, 1)$ and obtain the results in figure 4.1. We can see that the burn-in period is almost undetectable and the chain converges rapidly. Nonetheless, to estimate the accuracy, we discard the first 100 samples and save the subsequent samples with a lag of 10. We obtain an accuracy of 80% : the true starting positions being $[11, 11, 18, 5, 4]$ and the estimated positions being $[11, 12, 18, 5, 4]$. We can also note that for the second sequence, the estimated start position 12 is close to the true start position 11, and inside the magic word.

Another way to check the convergence of the Markov Chain is to estimate how far the samples are from perfect mixing. We can therefore compute the R.hat (potential scale reduction factor). For that, we run 3 chains, compute the variance of the samples from each chain (after the halves of each have been discarded. We also compute the variance of all the chains mixed together.

$$R.hat = \sqrt{\frac{\text{mixture variance}}{\text{average within the chain variances}}}$$

After 200 iterations of the chain described above, we obtain : $R.hat = 1.02$, which indicates that the chain has converged, since the distributions between and within the chains are identical.

4.2 Effect of α

The accuracy of the estimation depends on whether the categorical distributions of the magic words and of the background are distinguishable. It would therefore follow that the more skewed

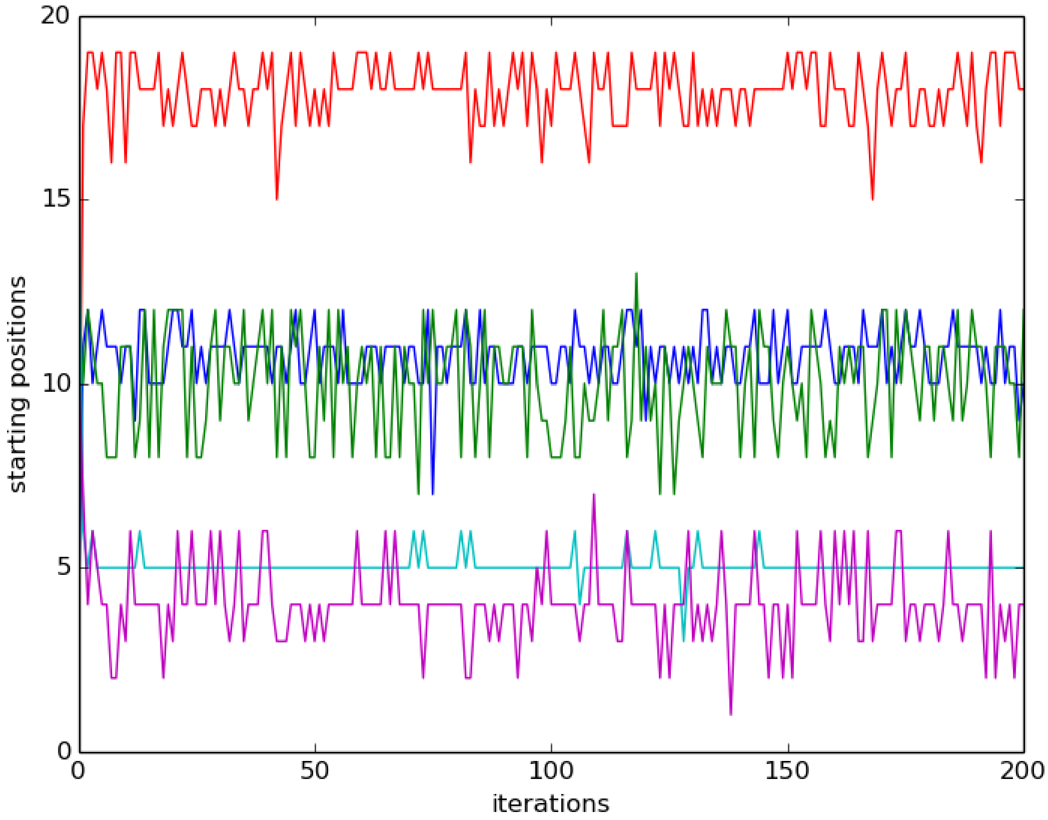

FIGURE 4.1: Markov chain samples for $\alpha' = (1, 1, 1, 1)$ and $\alpha = (12, 7, 3, 1)$

α is, or the more different from α', the higher the accuracy will be.

Since the accuracy can be highly dependent on the sequences we generated, we compute the average accuracy for 500 sets of sequences, for different vectors α. The results are presented in table 4.1

TABLE 4.1: Accuracy for different values of α. model with $\alpha' = (1, 1, 1, 1)$

α	Accuracy
$(12, 7, 3, 1)$	0.61
$(12, 7, 20, 16)$	0.63
$(2, 2, 2, 2)$	0.402
$(12, 1, 1, 1)$	0.74

The lowest accuracy is for $\alpha = (2, 2, 2, 2)$ as expected, since it is very close to the prior for the background. Wee can see the accuracy the highest when the Dirichlet parameter is "tilted" towards one of the letters $\alpha = (12, 1, 1, 1)$. The difference between the components of α and $alpha'$ seems to have less impact on the accuracy than the skewness of $Dir(\alpha)$. Figure 4.2 shows the convergence plot for a set of sequences using $\alpha' = (1, 1, 1, 1)$ and $\alpha = (12, 1, 1, 1)$.

The accuracy for this data set is 80%: the true positions are $[1, 9, 17, 5, 11]$ and the estimated positions are $[1, 9, 17, 5, 10]$.

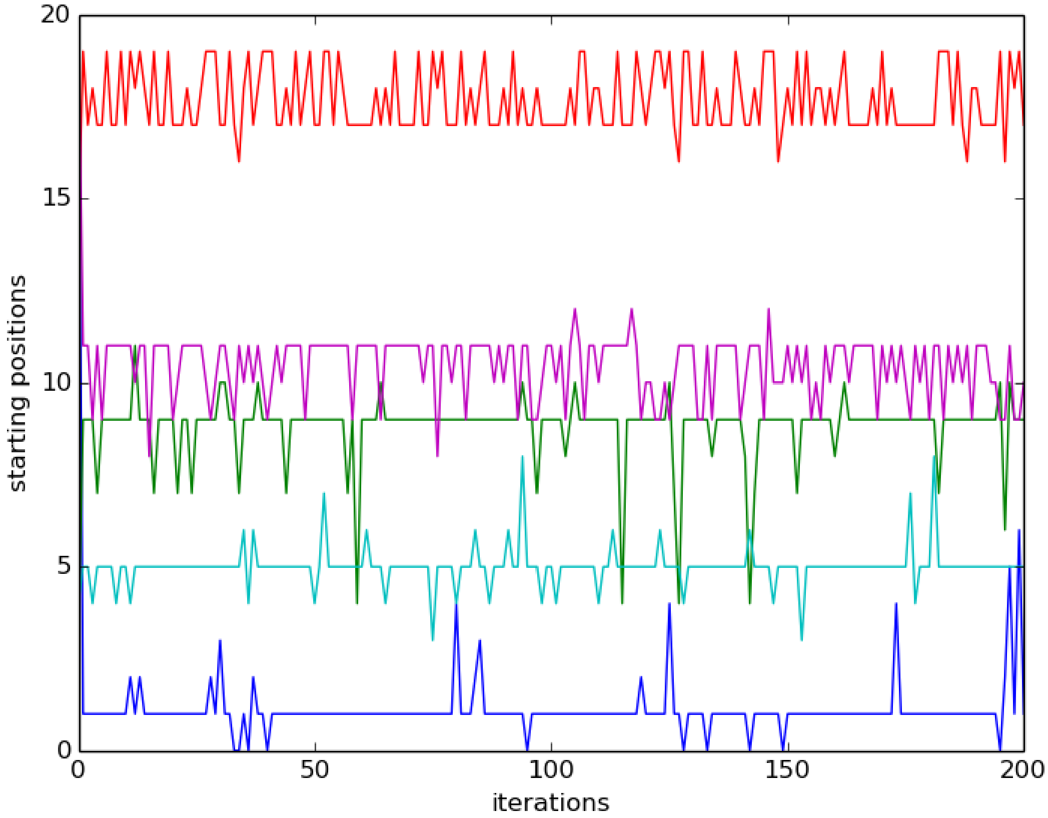

FIGURE 4.2: Markov chain samples for $\alpha' = (1, 1, 1, 1)$ and $\alpha = (12, 1, 1, 1)$

4.3 Effect of the lag

We have considered that the estimated starting position is the mode of the posterior distribution $p(r_n|R_{-n}, D)$. It seems reasonable to think that the less correlated the samples we draw (after convergence) are, the more accurate our estimation of the node will be. To verify this, we compute the average accuracy for 500 sets of sequences, for different values of the lag. To get a better estimation of the mode, we run the Gibbs sampler for 500 iterations but only discard the first 100 samples, since we have seen that the convergence is achieved rapidly. The results are presented in table 4.2

As we can see, the difference between no lag and a lag of 10 samples is not significant, and with a lag of 30 samples, the average accuracy is even slightly lower. we conclude that the purpose of saving every kth iteration for our application is not statistical, since the correlation doesn't seem to influence the accuracy. However, it could have computational benefits if we run

TABLE 4.2: Accuracy for different lag lengths, model with $\alpha' = (1, 1, 1, 1)$ and $\alpha = (12, 7, 3, 1)$

Lag	Accuracy
1 (no lag)	0.66
10 samples	0.67
30 samples	0.63

multiple chains with a very high number of samples in parallel and do not want to carry them all around in our simulation.