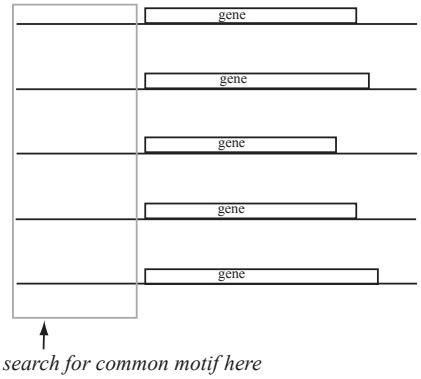
11 Motif Finding

Sources for this section:

- Rouchka, 1997, "A Brief Overview of Gibbs Sampling".
- J. Buhler, M. Tompa: Finding motifs using random projections, RECOMB 2001, 69-75.
- P.A. Pevzner and S.-H. Sze, Combinatorial approaches to finding subtle signals in DNA sequences, ISMB 2000, 269–278.

The goal of motif finding is the detection of new, unknown "signals" in a set of sequences.

For example, assume we have a collection of genes that appear to be co-regulated and have the same pattern of expression. You would like to search for a transcription factor binding site that occurs in the upstream regions of all the genes.



The aim is to compute something like this, where we highlight a motif that occurs in all given DNA sequences:

```
TCAGAACCAGTTATAAATTTATCATTTCCTTCTCCACTCCT
CCCACGCAGCCGCCCTCCTCCCCGGTCACTGACTGGTCCTG
TCGACCCTCTGGAACCTATCAGGGACCACAGTCAGCCAGGCAAG
AAAACACTTGAGGGAGCAGATAACTGGGCCAACCATGACTC
GGGTGAATGGTACTGCTGATTACAACCTCTGGTGCTGC
AGCCTAGAGTGATGACTCCTATCTGGGTCCCCAGCAGGA
GCCTCAGGATCCAGCACACATTATCACAAACTTAGTGTCCA
CATTATCACAAACTTAGTGTCCATCCATCACTGCTGACCCT
TCGGAACAAGGCAAAGGCTATAAAAAAATTAAGCAGC
GCCCCTTCCCCACACTATCTCAATGCAAATATCTGTCTGAAACGGTTCC
CATGCCCTCAAGTGTGCAGATTGGTCACAGCATTTCAAGG
GATTGGTCACAGCATTTCAAGGGAGAGACCTCATTGTAAG
TCCCCAACTCCCAACTGACCTTATCTGTGGGGGAGGCTTTTGA
CCTTATCTGTGGGGGAGGCTTTTGAAAAGTAATTAGGTTTAGC
ATTATTTTCCTTATCAGAAGCAGAGAGACAAGCCATTTCTCTTTCCTCCCGGT
AGGCTATAAAAAAATTAAGCAGCAGTATCCTCTTGGGGGCCCCTTC
CCAGCACACACACTTATCCAGTGGTAAATACACATCAT
TCAAATAGGTACGGATAAGTAGATATTGAAGTAAGGAT
ACTTGGGGTTCCAGTTTGATAAGAAAAGACTTCCTGTGGA
TGGCCGCAGGAAGGTGGGCCTGGAAGATAACAGCTAGTAGGCTAAGGCCAG
CAACCACAACCTCTGTATCCGGTAGTGGCAGATGGAAA
CTGTATCCGGTAGTGGCAGATGGAAAGAGAAACGGTTAGAA
GAAAAAAAATAAATGAAGTCTGCCTATCTCCGGGCCAGAGCCCCT
TGCCTTGTCTGTTGTAGATAATGAATCTATCCTCCAGTGACT
GGCCAGGCTGATGGGCCTTATCTCTTTACCCACCTGGCTGT
CAACAGCAGGTCCTACTATCGCCTCCCTCTAGTCTCTG
CCAACCGTTAATGCTAGAGTTATCACTTTCTGTTATCAAGTGGCTTCAGCTATGCA
GGGAGGGTGGGGCCCCTATCTCTCCTAGACTCTGTG
CTTTGTCACTGGATCTGATAAGAAACACCACCCCTGC
```

11.1 The Gibbs sampling method

We will first describe an algorithm based on the *Gibbs-sampler*.

Basic assumptions:

- Input is a set of DNA sequences $s_1, \ldots, s_m$.
- Each sequence contains a copy or *instance* of the motif.
- The motif has a fixed length ℓ.
- The motif is generated by a *position weight matrix*.

11.1.1 Sequence generating models

We describe two models for randomly generating DNA sequences:

1. The *background model* B is given by a vector  that specifices a probability for each of the four character states.
2. The *motif model* M specifies for each position $i = 1, 2, \ldots$ a probability $M(i, c)$ for each of the four states c for a word of length ℓ:

	1	2	3	...	ℓ
A					
C					
G					
T					

   We will refer to this as a *position weight matrix*, although usually it is required that a position weight matrix contains the logarithms of the probabilities rather than the probabilities themselves.

11.1.2 Simple computations

If we know both M and B then we can score any word w of length ℓ. In more detail, with

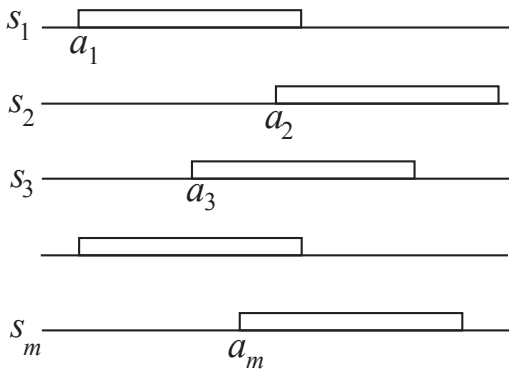
$$P(w) = \prod_{i=1}^{\ell} B(w[i]) \quad \text{and} \quad Q(w) = \prod_{i=1}^{\ell} M(i, w[i])$$

we can define the *log-odds ratio* as

$$R(w) = \log \frac{Q(w)}{P(w)}.$$

If $R(w) > 0$, it is more likely that the word w was generated by the motif model. If $R(w) < 0$, then it is more likely that w was generated by the random model.

If both M and B are given, then we can make a maximum likelihood prediction of the locations of the motif in sequence of the input sequences:



In each sequence s_i, choose a location a_i such that the word w starting at a_i maximizes $R(w)$.

Vice versa, if we are given the locations $a_1, \ldots, a_m$ of the instances of the motif in $s_1, \ldots, s_m$, then we can setup M and B for all states c and positions i as follows:

$$B(c) = \frac{\#\text{ occurences of state } c \text{ outside of motif}}{\#\text{ all bases outside of motif}}$$

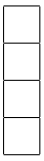
and

$$M(i,c) = \frac{\#\text{ occurrences of } c \text{ at the } i\text{-th position of the motif}}{m}.$$

For example consider the following sequences and motif instances:

A C G **T A A**
G C **G T T** A
A C T T T G

We have

$B =$ 

and

$M =$

	1	2	3	...	ℓ
A					
C					
G					
T					

Note: to avoid zeros in either distributions, *pseudo counts* are added to all counts.

The *maximum likelihood* score of a choice of motif locations $a_1, \ldots, a_m$ is defined as follows:

$$\begin{aligned} L(a_1,\ldots,a_m) &= L(a_1,\ldots,a_m,M,B) = \\ &= P(s_1,\ldots,s_m \mid a_1,\ldots,a_m,M,B) \\ &= \prod_{i=1}^{m} \left(\prod_{j=1}^{a_i-1} B(s_i[j])\right) \left(\prod_{j=a_i}^{a_i+\ell-1} M(j,s_i[j])\right) \\ &\qquad \left(\prod_{j=a_i+\ell}^{|s_i|} B(s_i[j])\right), \end{aligned}$$

where $|s_i|$ = length of s_i. In this formula, we use either the background model or the motif model to assign a probability to a given position in a given sequence depending on whether the position is outside of, or inside of, the current placement of the motif, respectively.

11.1.3 Motif-finding dilemma

Motif-finding dilemma:

- If we know M and B, then we can determine the most likely motif locations $a_1, \ldots, a_m$ in $s_1, \ldots, s_m$.

- If we know $a_1, \ldots, a_m$ then we can determine M and B.

In motif-finding we know *neither* and therefore want to infer *both*.

Idea: Construct M' and B' from a_i's from $m-1$ sequences and then use M' and B' to determine a value for a_i for remaining sequence s_i. Repeat.

The *Gibbs-sampler* is based on this idea. This method can be interpreted as an MCMC approach in which proposed new solutions are always accepted.

11.1.4 Gibbs sampling algorithm

Algorithm 11.1.1 (Gibbs-sampler-based motif finding)

Input: Sequences $s_1, \ldots, s_m$
Output: Motif M*, background* B*, locations* $a_1, \ldots, a_m$
Init.:
Choose $a_1, \ldots, a_m$ *either randomly or using an algorithm, as described later.*
repeat
  for $h = 1, \ldots, m$ **do**
    Compute M' *and* B' *from* $a_1, \ldots, a_{h-1}, a_{h+1}, \ldots, a_m$
      (i.e., from data with h-th sequence and motif removed)
    Using M' *and* B'*, compute a new location* a_h *in* s_h $(*)$
    Compute M *and* B *from* $a_1, \ldots, a_m$
     (i.e., from data with new choice of h-th motif location inserted)
    Compute score $L(a_1, \ldots, a_m)$
until *score* $L(a_1, \ldots, a_m)$ *has converged*
Return M*,* B *and* $a_1, \ldots, a_m$
end

In line $(*)$ choose the new value a for a_h probabistically according to the distribution of normalized log-odds scores:

$$\frac{R(a)}{\sum_{a'=1}^{|s_h|-\ell+1} R(a)}.$$

Here, $R(a)$ is the score obtained for M' and B' applied to sequence s_h and motif location a.

11.1.5 Example

We will now discuss an example taken from Rouchka (1997)[1]. The motif length is $\ell = 6$.

Here we show the input sequences and the initial choice of locations (in red):

[1] Rouchka, 1997, "A Brief Overview of Gibbs Sampling"

Nucleotide	Motif Position (0 = Background)						
	0	1	2	3	4	5	6
A	279	6	12	6	6	11	7
C	280	8	3	5	7	7	7
G	225	9	8	10	7	5	8
T	262	6	6	8	9	6	7

Table 1: Caclulation of observed counts for inital alignment of figure 1

Nucleotide	Motif Position (0 = Background)						
	0	1	2	3	4	5	6
A	.267	.256	.296	.256	.256	.289	.263
C	.267	.263	.230	.243	.256	.256	.256
G	.216	.240	.233	.246	.226	.213	.233
T	.250	.241	.241	.254	.261	.241	.248

Table 2: Calculation of residue frequencies for initial alignment of figure 1

The number of `A`'s in all of the sequences combined is 327, the number of `C`'s is 317, the number of `G`'s is 272, and the number of `T`'s is 304.

To avoid zero entries in the motif matrix M, we will use 10% of each of the background counts as *pseudocounts* that are always added to the different counts obtained for different entries of M when calculating the frequencies.

The pseudo counts to be added to values in M equal that is 32.7, 31.7, 27.2 and 30.4 for `A`, `C`, `G` and `T`, respectively.

We will not look at the details of this, but this explains why the tables containing observed counts don't translate directly into the tables containing frequencies.

In the first iteration of the main loop we remove the first sequence s_1 =`TCAGAACCAGTTATAAATTTATCATTTCCTTCTCCACTCCT`.

Here are M' and B' for the first sequence removed:

Nucleotide	Motif Position (0 = Background)						
	0	1	2	3	4	5	6
A	281	5	12	6	6	10	7
C	280	8	3	5	7	7	7
G	225	9	8	10	7	5	8
T	266	6	5	5	5	6	6

Table 3: Recalculated observed counts

Nucleotide	Motif Position (0 = Background)						
	0	1	2	3	4	5	6
A	.267	.251	.298	.258	.258	.285	.265
C	.265	.264	.231	.245	.258	.258	.258
G	.215	.241	.235	.248	.228	.215	.235
T	.252	.243	.236	.236	.236	.243	.243

Table 4: Recalculated residue frequencies

We then score all $36 = 41 - 6 + 1$ possible words of length ℓ in s_1 =`TCAGAACCAGTTATAAATTTATCATTTCCTTCTCCACTCCT`.

Sequence	X	A_x	Normalized A_x
TCAGAA	1	0.906	.030
CAGAAC	2	1.283	.042
AGAACC	3	0.905	.030
GAACCA	4	1.131	.037
AACCAG	5	1.093	.036
ACCAGT	6	0.700	.023
CCAGTT	7	0.822	.027
CAGTTA	8	1.142	.037
AGTTAT	9	0.919	.030
GTTATA	10	0.902	.029
TTATAA	11	0.856	.028
TATAAA	12	1.021	.033
ATAAAT	13	0.839	.027
TAAATT	14	0.923	.030
AAATTT	15	0.875	.029
AATTTA	16	0.873	.028
ATTTAT	17	0.787	.026
TTTATC	18	0.757	.025
TTATCA	19	0.781	.025
TATCAT	20	0.997	.033
ATCATT	21	0.723	.024
TCATTT	22	0.698	.023
CATTTC	23	0.907	.030
ATTTCC	24	0.725	.024
TTTCCT	25	0.762	.025
TTCCTT	26	0.743	.024
TCCTTC	27	0.674	.022
CCTTCT	28	0.709	.023
CTTCTC	29	0.791	.026
TTCTCC	30	0.731	.024
TCTCCA	31	0.732	.024
CTCCAC	32	0.864	.028
TCCACT	33	0.696	.023
CCACTC	34	0.761	.025
CACTCC	35	0.904	.030
ACTCCT	36	0.695	.023

Table 5 :Weights for segments within sequence 1

The column headed *normalized* A_x contains the normalized log-odd scores for each choice of motif location in s_1.

The algorithm chooses one of the locations according to the given distribution of log-odds scores.

We then repeat the main loop and remove the second sequence etc.

This is repeated until the score coverges or until a set maximum number of iterations has been performed.

Here is the final choice of motif:

TCAGAACCAGTTATAAATTTATCATTTCCTTCTCCACTCCT
CCCACGCAGCCGCCCTCCTCCCCGGTCACTGACTGGTCCTG
TCGACCCTCTGGAACCTATCAGGGACCACAGTCAGCCAGGCAAG
AAAACACTTGAGGGAGCAGATAACTGGGCCAACCATGACTC
GGGTGAATGGTACTGCTGATTACAACCTCTGGTGCTGC
AGCCTAGAGTGATGACTCCTATCTGGGTCCCCAGCAGGA
GCCTCAGGATCCAGCACACATTATCACAAACTTAGTGTCCA
CATTATCACAAACTTAGTGTCCATCCATCACTGCTGACCCT
TCGGAACAAGGCAAAGGCTATAAAAAAAATTAAGCAGC
GCCCCTTCCCCACACTATCTCAATGCAAATATCTGTCTGAAACGGTTCC
CATGCCCTCAAGTGTGCAGATTGGTCACAGCATTTCAAGG
GATTGGTCACAGCATTTCAAGGGAGAGACCTCATTGTAAG
TCCCCAACTCCCAACTGACCTTATCTGTGGGGGAGGCTTTTGA
CCTTATCTGTGGGGGAGGCTTTTGAAAAGTAATTAGGTTTAGC
ATTATTTTCCTTATCAGAAGCAGAGAGACAAGCCATTTCTCTTTCCTCCCGGT
AGGCTATAAAAAAAATTAAGCAGCAGTATCCTCTTGGGGGCCCCTTC
CCAGCACACACACTTATCCAGTGGTAAATACACATCAT
TCAAATAGGTACGGATAAGTAGATATTGAAGTAAGGAT
ACTTGGGGTTCCAGTTTGATAAGAAAAGACTTCCTGTGGA
TGGCCGCAGGAAGGTGGGCCTGGAAGATAACAGCTAGTAGGCTAAGGCCAG
CAACCACAACCTCTGTATCCGGTAGTGGCAGATGGAAA
CTGTATCCGGTAGTGGCAGATGGAAAGAGAAACGGTTAGAA
GAAAAAAAATAAATGAAGTCTGCCTATCTCCGGGCCAGAGCCCCT
TGCCTTGTCTGTTGTAGATAATGAATCTATCCTCCAGTGACT
GGCCAGGCTGATGGGCCTTATCTCTTTACCCACCTGGCTGT
CAACAGCAGGTCCTACTATCGCCTCCCTCTAGTCTCTG
CCAACCGTTAATGCTAGAGTTATCACTTTCTGTTATCAAGTGGCTTCAGCTATGCA
GGGAGGGTGGGGCCCCTATCTCTCCTAGACTCTGTG
CTTTGTCACTGGATCTGATAAGAAACACCACCCCTGC

Nucleotide	Motif Position (0 = Background)						
	0	1	2	3	4	5	6
A	276	3	1	21	0	11	15
C	287	10	0	0	0	18	2
G	256	0	8	8	0	0	0
T	227	16	20	0	29	0	12

Table 6: Final site sampler residue counts

Nucleotide	Motif Position (0 = Background)						
	0	1	2	3	4	5	6
A	.264	.236	.223	.356	.217	.289	.315
C	.273	.276	.210	.210	.210	.329	.223
G	.242	.180	.233	.233	.180	.180	.180
T	.220	.307	.334	.201	.393	.201	.281

Table 7: Final site sampler residue frequencies

11.2 The Projection Method

We will now describe the *projection method*.

Again, the computational problem is to determine a motif by analyzing a set of sequences that contain instances of the motif. Here is another way to formalize the motif-finding problem (Pevzner and Sze):

Planted (ℓ, d)-Motif Problem: Suppose there is a fixed but unknown nucleotide sequence M (the *motif*) of length ℓ. The problem is to determine M, given m sequences, each of length n and each containing a *planted* variant of M. More precisely, each such planted variant is a substring of length ℓ that differs from M in at most d positions.

In the Planted (ℓ, d)-Motif Problem assume the motif is

ACAGGATCA.

The following 4 sequences each contain a planted version of this motif:

AGTTATCGCGGCACAGGCTCCTTCTTTATAGCC
ATGATAGCATCAACCTAACCCTAGATATGGGAT
TTTTGGGATATATCGCCCCTACACAGGATCACT
GGATATACAGGATCACGGTGGGAAAACCCTGAC

Notice that some variants fully agree on a subset of the positions of the full motif:

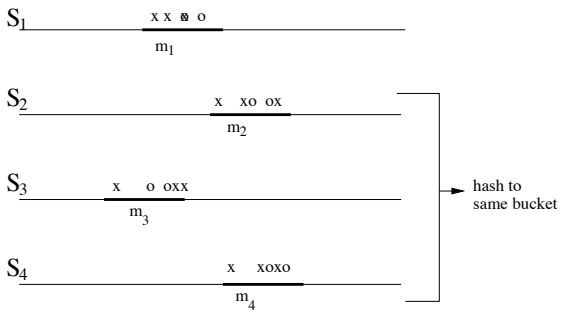
ACAGGcTCc
AtAGcATCA
ACAGGATCA
ACAGGATCA

The key idea is to repeatedly choose k random positions from the ℓ positions of the motif and then to hash words from the sequences using only those positions, in effect *projecting* to those k positions:

$$\text{ACAGGATCA} \xrightarrow{project} \text{AAGTC}$$

If we happen to choose a set of k positions that are well conserved by the motif then we will obtain a hash bucket that contains a higher-than-expected number of hits.

In other words, to address the Planted (ℓ, d)-Motif Problem, the key idea is to choose k of ℓ positions at random, then to use the k selected positions of each ℓ-mer w in a hash function $h(x)$. When a larger-than-expected number of ℓ-mers hash to the same bucket, then it is likely that the bucket is enriched for instances of the planted motif M:



(Here, for each instance m_i of the planted motif M, x's mark the $d = 3$ substitutions and o's mark the $k = 2$ positions used in hashing.)

Like many probabilistic algorithms, the projection algorithm performs a number of independent runs of a basic iteration.

In each such trial, it chooses a random projection h and hashes each ℓ-mer x in the input sequences to its bucket $h(x)$.

Any hash bucket with a high number of entries is explored as a source of the planted motif, in a refinement step, as described below.

If $k < \ell - d$, then there is a good chance that some of the m planted instances of M will be hashed to the same bucket (the *planted bucket*), namely all planted instances for which the k hash positions and d substituted positions are disjoint.

So, there is a good chance that the planted bucket will be enriched for the planted motif, and will contain more entries than an average bucket.

11.2.1 Example

Assume we are given the sequences $\left\{\begin{array}{ll} & \texttt{1234567} \\ s_1 & \texttt{cagtaat} \\ s_2 & \texttt{ggaactt} \\ s_3 & \texttt{aagcaca} \end{array}\right\}$. Let $M = \texttt{aaa}$ be the (unknown) $(3,1)$-motif.

Hashing using the first two positions of every word of length 3 we get the following hash table:

$h(x)$	pos.
aa	(1,5), (2,3), (3,1)
ac	(2,4), (3,5)
ag	(1,2), (3,2)
at	(1,6)
ca	(1,1), (3,4), (3,6)
cc	
cg	
ct	(2,5)
ga	(2,2)
gc	(3,3)
gg	(2.1)
gt	(1,3)
ta	(1,4)
tc	
tg	
tt	(2,6)

The motif M is planted at positions $(1,5)$, $(2,3)$, $(3,1)$ and $(3,5)$ and in this example, three of the four instances hash to the planted bucket $h(M) = \texttt{aa}$.

11.2.2 Choosing the parameters

Obviously, the algorithm does not know which bucket is the actual planted bucket. So it considers every bucket that contains at least s elements, where s is a suitable threshold.

The algorithm has three main parameters:

- the *projection size* k,
- the *bucket (inspection) threshold* s, and
- and the *number of independent runs* t.

In the following, we will discuss how to choose each of these parameters.

projection size: Ideally, the algorithm should hash a significant number of instances of the motif into the planted bucket, while avoiding contamination of the planted bucket by random background ℓ-mers.

To minimize the contamination of the planted bucket, we must choose k large enough. What size must we choose k so that the average bucket will contain less than 1 random ℓ-mer?

Since we are hashing $m(n - \ell + 1)$ ℓ-mers into 4^k buckets, if we choose k such that

$$4^k > m(n - \ell + 1),$$

then the average bucket will contain less than one random ℓ-mer.

For example, in a Planted $(15, 4)$-Problem, with $m = 20$ and $n = 600$, we must choose k to satisfy:

$$k < \ell - d = 15 - 4 = 11 \quad \text{and}$$

$$k > \frac{\log(m(n - \ell + 1))}{\log(4)} = \frac{\log(20(600 - 15 + 1))}{\log(4)} \approx 6.76.$$

If the total number of sequences is very large, then it may be that one cannot choose k to satisfy both $k < \ell - d$ and $4^k > m(n - \ell + 1)$. In this case, set $k = \ell - d - 1$, as large as possible.

Bucket threshold: A bucket size of $s = 3$ or 4 is practical, as we should not expect too many instances to hash to the same bucket in a reasonable number of runs.

Number of independent runs: We want to choose t so that the probability is at least $q = 0.95$ that the planted bucket contains s or more planted motif instances in at least one of the t runs.

Let $\hat{p}(\ell, d, k)$ be the probability that a given planted motif instance hashes to the planted bucket, that is:

$$\hat{p}(\ell, d, k) = \frac{\binom{\ell-d}{k}}{\binom{\ell}{k}}.$$

Then the probability that fewer than s planted instances hash to the planted bucket in a given trial is $B_{m,\hat{p}(\ell,d,k)}(s)$.

Here, $B_{m,p}(s)$ is the probability that there are fewer than s successes in m independent Bernoulli trials, each having probability p of success.

If the algorithm is run for t runs, the probability that s or more planted instances hash to the planted bucket is:

$$1 - \left(B_{m,\hat{p}(\ell,d,k)}(s)\right)^t$$

For this to be $\geq q$, choose

$$t = \left\lceil \frac{\log(1-q)}{\log(B_{m,\hat{p}(\ell,d,k)}(s))} \right\rceil. \tag{11.1}$$

Using this criterion for t, the choices for k and s above require at most thousands of runs, and usually many fewer, to produce a bucket containing sufficiently many instances of the planted motif.

11.2.3 Motif refinement

The main loop of the projection algorithm finds a set of buckets of size $\geq s$. In the subsequent *refinement step* each such bucket is explored in an attempt to recover the planted motif.

The idea is that, if the current bucket is the planted bucket, then we have already found k of the planted motif residues. These, together with the remaining $\ell - k$ residues, should provide a strong signal that makes it easy to obtain the motif in only a few iterations of refinement.

We will process each bucket of size $\geq s$ to obtain a candidate motif. Each of these candidates will be "refined" and the best refinement will be returned as the final solution.

Candidate motifs are refined either using the Gibbs-sampler or a related approach called the *EM algorithm*.

In essence, the projection method is used to compute a good starting point and thus greatly speed-up the convergence of a method such as the Gibbs-sampler algorithm.