Supporting material for "MCMC sampling of Markov models for ion channels"

Ivo Siekmann
Auckland Bioengineering Institute,
The University of Auckland, Auckland, New Zealand

Larry E. Wagner II, David Yule
Department of Pharmacology and Physiology,
University of Rochester Medical Center, Rochester, New York 14642, USA

Colin Fox
Department of Physics,
University of Otago, Dunedin, New Zealand

David Bryant
Department of Mathematics and Statistics,
University of Otago, Dunedin, New Zealand

Edmund J. Crampin[1]
Auckland Bioengineering Institute and Department of Engineering Science,
The University of Auckland, Auckland, New Zealand

James Sneyd
Department of Mathematics,
The University of Auckland, Auckland, New Zealand

[1]Corresponding author. Address: Auckland Bioengineering Institute, The University of Auckland, 70 Symonds St, Auckland, New Zealand, Tel.: (+64) 9 373 7599 ext. 88168, Fax: (+64)9 367 7157

1 The forward-backward algorithm for sampling sequences of states

In the following we describe a method for calculating sequences of Markov states which are consistent with both the data I and the model Q. This algorithm was previously used by Rosales et al. (1), Rosales (2), Rosales et al. (3), Rosales and Varanda (4) and was first introduced by Carter and Kohn (5).

Figure S1 shows how a sequence of current samples I is transformed to open and closed events E. Open and closed events (denoted O and C, respectively) are represented in the Markov model by open and closed states. If a state S_i represents an open event, we also write O_i as a shorthand and, analogously, C_j if S_j stands for a closed state. The *forward-backward* algorithm as it is described here assigns a sequence of Markov states M to a given sequence of events E in a suitable way. Following the approach of Rosales et al. (1) and Rosales (2) it is straight-forward to include more advanced methods for idealising the currents I instead of using the sequence of events E which was obtained by simple thresholding. Instead of idealising the currents I, a filter can be included in Eq. 10 of the main text. Such a filter is capable of taking into account characteristics of the noise both from the recording apparatus as from the channel itself. See Rosales et al. (1) and Rosales (2) for further details.

The idea of the algorithm is simple: In the forward phase, probability distributions are calculated that at a position k in the sequence E the model is in a state $M^k \in S$. These probability distributions $\mathbb{P}(M^k|(E^1, ..., E^k), Q)$ can be computed iteratively starting from the first event E^1, which is why this is named the *forward phase* of the algorithm. In the second phase, starting from $\mathbb{P}(M^N|(E^1, ..., E^N), Q)$, a state M^k is then sampled for each position k in the sequence, hence moving backwards from the last to the first element of M. Both parts of the algorithm lean on the fact that transitions from a state $M^k = S_m$ to $M^{k+1} = S_n$ can be looked up in the transition matrix $A_\tau = \exp(Q\tau)$ from the Markov model.

1.1 *forward:* Probabilities for the end of a sequence

In the forward phase of the algorithm probabilities for assigning a state M^k to an event E^k are calculated. An example for a model with three closed states C_1, C_2 and C_3 and two open states O_4 and O_5 is given in Figure 1b of the main text. Assuming that the Markov model is at equilibrium, the probabilities $\mathbb{P}(M^1|E^1, Q)$ are obtained by scaling the stationary distribution π,

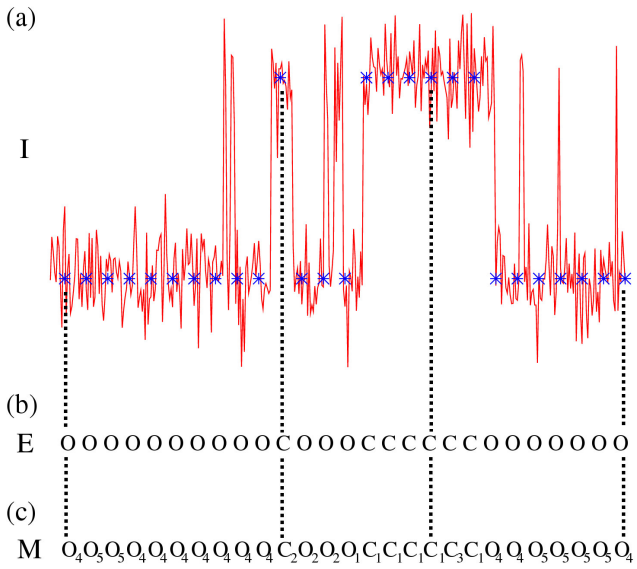

Figure S1: Current samples which are shown in (a) are transformed to a sequence of open and closed events E, see (b). Open and closed events are represented by states of a Markov model which cannot be found directly from E. However, because transition probabilities between consecutive states M^{k-1} and M^k are known from the model Q, a probability distribution $\mathbb{P}(M^k = S_m|(E^1, ..., E^k), Q)$ that the model is in a particular state S_m can be calculated for each position k. The *forward-backward algorithm* allows for generating realisations of the sequence (M^k).

see Eq. 7 in the main text. For the example given in Figure S1c,

$$\mathbb{P}(M^1 = O_4|E^1, Q) = \frac{\pi_4}{\pi_4 + \pi_5}. \tag{S1}$$

In general,

$$\mathbb{P}(M^1 = S_n|E^1, Q) = \begin{cases} \frac{\pi_n}{\sum_{j:M^1=S_j\in E^1} \pi_j}, & M^1 = S_n \in E^1 \\ 0, & M^1 = S_n \notin E^1 \end{cases} \tag{S2}$$

Note that $\mathbb{P}(M^1 = S_n|E^1, Q)$ depends on the Markov model Q due to the stationary probabilities π_j.

Now that the initial probabilities $\mathbb{P}(M^1 = S_m|E^1, Q)$ are calculated, transitions to a state $M^2 = S_n$ can be found by taking into account that the transition probability $\mathbb{P}(M^2 = S_n|M^1 = S_m, E^2, Q)$ is given by the Markov model, see Eq. 6 in the main text, bearing in mind that only transitions to open or closed events are allowed depending on E^2. For the example (Figure S1c), we have

$$\begin{aligned} &\mathbb{P}(M^2 = O_5|M^1 = O_4, (E^1, E^2), Q) \\ &\qquad \propto \mathbb{P}(M^2 = O_5|M^1 = O_4, E^2, Q)\mathbb{P}(M^1 = O_4|E^1, Q) \\ &\qquad = \rho_{45}\mathbb{P}(M^1 = O_4|E^1, Q). \end{aligned} \tag{S3}$$

For a general index k the probability $\mathbb{P}(M^k = S_n|M^{k-1} = S_m, (E^1, ..., E^k), Q)$ can be calculated analogously and Eq. S3 becomes

$$\begin{aligned} &\mathbb{P}(M^k = S_n|M^{k-1} = S_m, (E^1, ..., E^k), Q) \\ &\qquad \propto \mathbb{P}(M^k = S_n|M^{k-1} = S_m, E^k, Q)\mathbb{P}(M^{k-1} = S_m|(E^1, ..., E^{k-1}), Q) \\ &\qquad = \rho_{mn}\mathbb{P}(M^{k-1} = S_m|(E^1, ..., E^{k-1}), Q). \end{aligned} \tag{S4}$$

where, similar to Eq. S2,

$$\mathbb{P}(M^k = S_n|M^{k-1} = S_m, E^k, Q) = \begin{cases} \rho_{mn}, & M^k = S_n \in E^k \\ 0, & M^k = S_n \notin E^k \end{cases} \tag{S5}$$

There still remains one difficulty because $\mathbb{P}(M^k = S_n|M^{k-1} = S_m, E^k, Q)$ depends on the state $M^{k-1} = S_m$ from which the transition to $M^k = S_n$

originated from. However, this can be resolved by summing over all possible preceding states $M^{k-1} = S_m$:

$$\mathbb{P}(M^k = S_n|(E^1, ..., E^k), Q) \propto \sum_{m=1}^{n_S} \mathbb{P}(M^k = S_n|M^{k-1} = S_m, (E^1, ..., E^k), Q). \tag{S6}$$

Using Eq. S2, Eq. S4 and Eq. S6, the probabilities in Eq. S6 are calculated iteratively for each $k = 1, ..., N$. Note that the $\mathbb{P}(M^k = S_n|(E^1, ..., E^k), Q)$ can be interpreted as the probability that the sequence $(E^1, ..., E^k)$ consisting of arbitrary open and closed states ends in the state S_n. This immediately leads to the idea of sampling a realisation of the state sequence M starting from the final state M^N which will be described in the next section.

1.2 *backward*: Sampling a sequence of states

Taking a look at the example again, from Figure S1b it can be seen that the sequence E ends in one of the open states, i.e., O_4 or O_5. By drawing a uniformly distributed random number it is decided which of the two is chosen. Figure S1c shows that for our example it has been decided that the sequence M ends in O_4. Thus, with $M^N = O_4$, the probability distribution for the preceding state $M^{N-1} = S_m$ is given by

$$\begin{aligned}\mathbb{P}(M^{N-1} = S_m|M^N = O_4, (E^1, ..., E^N), Q) \\ \propto \rho_{m,4}\mathbb{P}(M^{N-1} = S_m|(E^1, ..., E^{N-1}), Q),\end{aligned}$$

which generalises to

$$\begin{aligned}\mathbb{P}(M^k = S_m|M^{k+1} = S_n, (E^1, ..., E^{k+1}), Q) \\ \propto \rho_{mn}\mathbb{P}(M^k = S_m|(E^1, ..., E^k), Q)\end{aligned} \tag{S7}$$

for an arbitrary index k. Again, starting by sampling $M^N = S_n$ from $\mathbb{P}(M^N = S_n|(E^1, ..., E^N))$ and generating samples for each $k = N-1, ..., 1$ using Eq. S7 gives an iterative algorithm.

Together, the forward and the backward phase allow for assigning a sequence of Markov states (M^k) to a sequence of events which is consistent both with (E^k) as well as with the transition probabilities given by the Markov model Q. In the Methods section of the main text, a method for

drawing samples Q from the probability distribution $\mathbb{P}(Q|M,E)$ will be developed. Each time, a new sample Q is generated, the sequence (M^k) must be updated.

2 Quantitative comparison of MH and MHG with the Gibbs sampler by Rosales et al.

In Table 1, arithmetic mean estimates for A_τ^{MH} and A_τ^{MHG} are compared with the estimates A_τ^G from Rosales' Gibbs sampler (1) using the relative error

$$\rho_{ij} := \frac{\hat{a}_{ij} - a_{ij}}{a_{ij}} \tag{S8}$$

where $\hat{a}_{ij}$ are the estimates for the components a_{ij} of the matrix A_τ given by the arithmetic mean of the samples for the matrices A_τ^{MH} or A_τ^{MHG}, respectively. Table 1 demonstrates that the new approach leads to clearly better estimates. Especially the smaller entries seem to be generally overestimated by the Gibbs sampler.

3 Numerical methods and implementation of MH and MHG

Here, we supplement the description of the MCMC algorithms MH and MHG with some remarks on the implementation: Evaluating the likelihood Eq. 10 in the main text requires the calculation of the matrix exponential A_τ. As explained by Moler and van Loan (6), approximation of the matrix exponential is a difficult numerical problem. In our implementation and whenever matrix exponentials are calculated in this article, we use our own implementation of an algorithm described in (6, Method 3), a combination of scaling and squaring and a Padé approximation.

The MHG algorithm requires the sampling of the stationary distribution π from a Dirichlet distribution. As stated above, only half of the rate constants are sampled by a Metropolis-Hastings step whereas the other half is calculated from these rate constants and ratios of stationary probabilities, an approach which is based upon Eq. 8 in the main text. If a stationary probability is very small this might lead to numerical difficulties when applying Eq. 8 while a low stationary probability signifies that the corresponding state occurs only infrequently. Therefore, new samples are drawn whenever one

A_τ				
0.997124	0.00245388	0.000109095	0.000311789	$1.60819 \cdot 10^{-6}$
0.0126925	0.71308	0.0709044	0.201711	0.00161154
0.00019916	0.0250251	0.971564	0.00319554	$1.65222 \cdot 10^{-5}$
0.000263299	0.0329325	0.00147821	0.950746	0.0145803
$4.52693 \cdot 10^{-7}$	$8.77028 \cdot 10^{-5}$	$2.54764 \cdot 10^{-6}$	0.00486009	0.995049

relative error of: A_τ^G (percent)				
+0.06	-70.83	+726.49	+72.81	+9668.35
-61.00	-0.88	-14.07	-5.56	+2186.92
+501.14	-36.60	+0.39	+109.49	+5294.20
+233.92	+5.05	+251.24	-0.19	-28.78
+7291.41	+1878.40	+7562.69	-54.84	+0.08

A_τ^{MH} (percent)				
+0.06	-21.14	-15.90	-20.65	-14.11
-14.27	-0.48	+6.32	+0.30	+8.59
-15.42	-1.89	+0.06	-1.29	+6.78
-15.68	-1.97	+4.55	-0.06	+8.10
-5.39	+10.04	+17.30	+12.08	-0.06

A_τ^{MHG} (percent)				
+0.06	-22.01	-17.36	-21.91	-15.96
-15.81	-0.30	+5.83	-0.06	+7.55
-17.77	-2.62	+0.08	-2.48	+4.92
-17.49	-2.20	+3.71	-0.04	+7.50
-7.68	+9.44	+16.01	+11.78	-0.06

Table 1: Fits of test data generated from the model shown in Figure 1b, parameter values from Table 1b which both can be found in the main text. Colours are used for indicating the magnitude of the relative error as follows: Relative error is above 100 %: red, relative error is below 100 % but above 10 %: black, relative error below 10 %: green.

of the stationary probabilities has a value below a certain ϵ. This process is repeated until a sample is found which contains only stationary probabilities above ϵ. If this is unsuccessful after more than K iterations the algorithm

is stopped. In our implementation we chose $\epsilon = 10^{-12}$ and $K = 10,000$; different choices of ϵ and K gave similar results. This behaviour will be observed below if a model is fitted to test data which was generated from a less complex model with fewer states. We will interpret this as a hint that the model is overparametrised and a model with fewer states should be used for fitting.

For smaller data sets (about 40,000 data points), the simple Metropolis-Hastings algorithm as described by Eq. 17 and Eq. 18 in the main text is sufficient. However, if the number of data points increases, the values of the likelihood (Eq. 10 in the main text) become very small which results in slow convergence and low acceptance ratio. This can be improved by using adaptive MCMC methods—we successfully used our own implementation of the `t-walk` by Christen and Fox (7).

The algorithms developed in this article were implemented in ANSI C for the compiler `gcc`. For vector arithmetics and linear algebra the GNU Scientific Library (GSL), version 1.14, was used (8). All figures were produced with the plotting software `gnuplot` (9).

References

1. Rosales, R., J. A. Stark, W. J. Fitzgerald, and S. B. Hladky, 2001. Bayesian Restoration of Ion Channel Records using Hidden Markov Models. *Biophysical Journal* 80:1088–1103.

2. Rosales, R., 2004. MCMC for Hidden Markov Models incorporating aggregation of states and filtering. *Bulletin of Mathematical Biology* 66:1173–1199.

3. Rosales, R. A., M. Fill, and A. L. Escobar, 2004. Calcium regulation of single ryanodine receptor channel gating analyzed using HMM/MCMC statistical methods. *The Journal of General Physiology* 121:533–553.

4. Rosales, R. A., and W. A. Varanda, 2009. Allosteric Control of Gating Mechanisms Revisited: The Large Conductance Ca^{2+}-Activated K^+ Channel. *Biophysical Journal* 96:3987–3996.

5. Carter, C. K., and R. Kohn, 1994. On Gibbs sampling for state space models. *Biometrika* 81:541–553.

6. Moler, C., and C. van Loan, 2003. Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later. *SIAM Review* 45:3–49.

7. Christen, J. A., and C. Fox, 2010. A General Purpose Sampling Algorithm for Continuous Distributions (the t-walk). *Bayesian Analysis* 5:263–282.

8. Galassi, M., J. Davies, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth, and F. Rossi, 2009. GNU Scientific Library Reference Manual. GNU Manual. Network Theory Limited, 3rd edition. `http://www.gnu.org/software/gsl/`.

9. Janert, P. K., 2009. Gnuplot in Action—Understanding Data with Graphs. Manning, Greenwich.