Coupled Hidden Markov Models: Computational Challenges

Louis J. M. Aslett and Chris C. Holmes

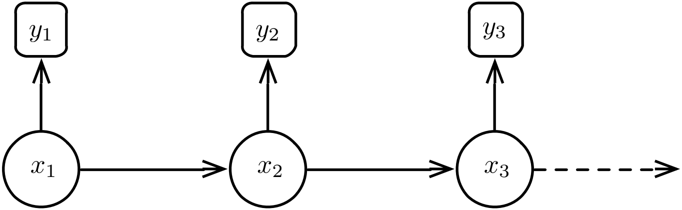
i-like Research Group
University of Oxford

Warwick Algorithms Seminar
7$^{\text{th}}$ March 2014

i-like.org.uk

Hidden Markov Models (HMM)

The standard HMM is well known and extensively studied:



Here, take finite discrete hidden state space and arbitrary observation state space.

$$
\begin{aligned}
Y_t \mid X_t &\sim F_Y(\cdot; X_t) & Y_t &\in \Omega \\
X_t \mid (X_{t-1} = i) &\sim \text{Discrete}(a_{i1}, \ldots, a_{iN}) & X_t &\in \{1, \ldots, N\}
\end{aligned}
$$

Hidden Markov Models (HMM)

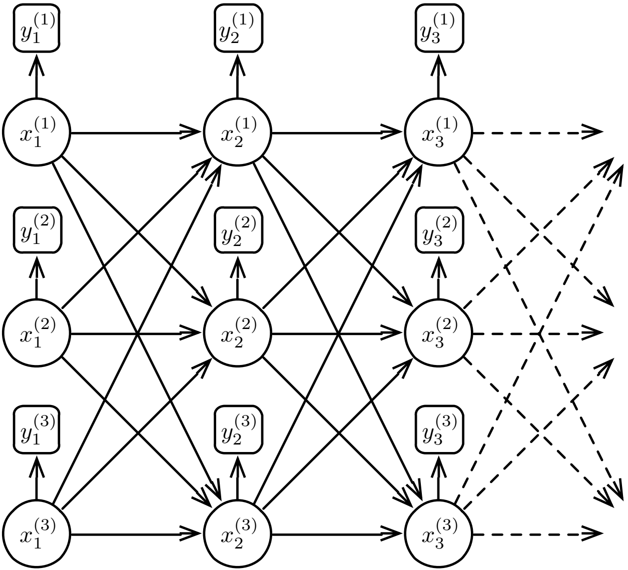
Traditionally forward and backward algorithms play a central role in inference for HMMs.

$$\begin{aligned}\alpha_t(i) &= \mathbb{P}(Y_1 = y_1, \ldots, Y_t = y_t, X_t = i) \\ &= \left(\sum_{j=1}^{N} \alpha_{t-1}(j)\ a_{ji}\right) f_Y(y_t; X_t = i)\end{aligned}$$

$$\begin{aligned}\beta_t(i) &= \mathbb{P}(Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \,|\, X_t = i) \\ &= \sum_{j=1}^{N} a_{ij}\, f_Y(y_{t+1}; X_{t+1} = j)\ \beta_{t+1}(j)\end{aligned}$$

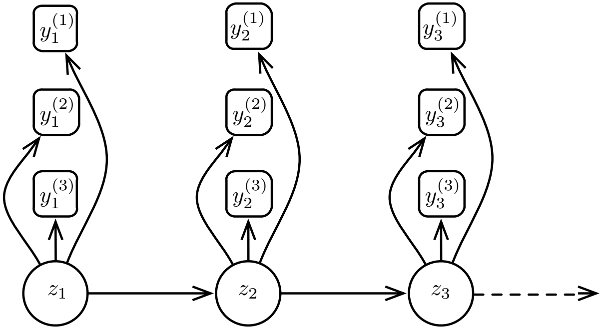
$$\mathbb{P}(X_t = i \,|\, \mathbf{y}) = \frac{\alpha_t(i)\beta_t(i)}{\sum_j \alpha_t(j)\beta_t(j)}$$

Coupled Hidden Markov Models (CHMM)



Coupled Hidden Markov Models (CHMM)

Can naïvely reformulate as:



Where $Z_t = (X_t^{(1)}, \dots, X_t^{(3)})$ and $Y_t^{(i)} \mid Z_t = Y_t^{(i)} \mid X_t^{(i)}$

$\Longrightarrow$ for C chains with $X_t^{(i)} \in \{1, \dots, N\}$, $|Z_t| = N^C$.

Coupled Hidden Markov Models (CHMM)

So, given C chains with N hidden states, the natural forward variable becomes:

$$\begin{aligned}\alpha_t(i_1,\ldots,i_C) &= \mathbb{P}(Y_{1:t}^{(1:C)} = y_{1:t}^{(1:C)}, X_t^{(1)} = i_1, \ldots, X_t^{(C)} = i_C) \\ &= \left(\sum_{j_1=1}^{N} \cdots \sum_{j_C=1}^{N} \alpha_{t-1}(j_1,\ldots,j_C) \prod_{k=1}^{C} \mathbb{P}(X_t^{(k)} = i_k \,|\, x_{t-1}^{(1:C)} = j_{1:C}) \right) \\ &\quad \times \prod_{k=1}^{C} f_Y(y_t^{(k)}; X_t^{(k)} = i_k)\end{aligned}$$

with a corresponding backward variable, so that:

$$\mathbb{P}(X_t^{(1)} = i_1, \ldots, X_t^{(C)} = i_C \,|\, \mathbf{y}) = \frac{\alpha_t(i_1,\ldots,i_C)\beta_t(i_1,\ldots,i_C)}{\sum_{j_1} \cdots \sum_{j_C} \alpha_t(j_1,\ldots,j_C)\beta_t(j_1,\ldots,j_C)}$$

CHMM: the computational challenge

The initial objective is to be able to perform inference in absolutely minimal setting of $N = 10$, $C = 100$, $T = 10^5$.

However, under the naïve formulation, there are numerous issues:

- computation of the forward variable involves N^C additions and C multiplications at each of T time steps;
- even assuming computation were possible, each forward variable requires $8N^C$ bytes of memory to store, and all T of them must be stored;
- the transition matrix itself is of dimension $N^C \times N^C$.

CHMM: the computational challenge

The initial objective is to be able to perform inference in absolutely minimal setting of $N = 10$, $C = 100$, $T = 10^5$.

However, under the naïve formulation, there are numerous issues:

- computation of the forward variable involves N^C additions and C multiplications at each of T time steps;
  $\geq 10^{105}$ computations
- even assuming computation were possible, each forward variable requires $8N^C$ bytes of memory to store, and all T of them must be stored;
  $\geq 7.45 \times 10^{96}$ GB memory
- the transition matrix itself is of dimension $N^C \times N^C$.
  $\geq 9.31 \times 10^{190}$ GB memory

Transitions: mixture model

A popular approach is to use the mixture model formulation of Saul and Jordan (1999). This replaces the $N^C \times N^C$ transition matrix with the transition model:

$$\mathbb{P}(X_t^{(i)} \mid x_{t-1}^{(1:C)}) = \sum_{k=1}^{C} \omega_{ki} \, \mathbb{P}(X_t^{(i)} \mid x_{t-1}^{(k)})$$

ω_{ki} can be viewed as mixing weights, or strength of effect of chain k on chain i.

This now involves only NC^2 parameters, but does not solve the computational issue.

Computation: marginal composite likelihood

Zhong and Ghosh (2002) made additional simplifications. After calculating forward variables for each chain, k,

$$\alpha_t^{(k)}(i) = \mathbb{P}(Y_{1:t}^{(k)} = y_{1:t}^{(k)}, X_t^{(k)} = i)$$

they use a marginal composite likelihood

$$\mathbb{P}(Y_{1:T}^{(1:C)}) \approx \prod_{k=1}^{C} \mathbb{P}(Y_{1:T}^{(k)}) = \prod_{k=1}^{C} \sum_{i=1}^{N} \alpha_T^{(k)}(i)$$

as part of an iterative self-mapping transformation algorithm (Baum *et al.*, 1970).

In practise, they only had $C = 2$ — scalable? In fact, same forward variable computation issue, but less memory required (only need $\alpha_T^{(\cdot)}(\cdot)$)

Transitions: structured matrix & logistic regression

Sherlock *et al.* (2013) replaced the full transition matrix with a structured transition matrix for each chain where probabilities were modelled with a logistic regression including others chains (and external factors) as covariates.

The computational approach was an adaptive random walk Metropolis-within-Gibbs algorithm.

The implementation was in C and had a run-time of 3 hours for 100000 iterations (9.3 it/sec) with $C = 6$, $2 \leq N \leq 4$ and $T \approx 20(?)$ with 1841 such sequences.

Transitions: logistic regression

Choi *et al.* (2013) used logistic regression directly for the transition probabilities in a CHMM with $N = 2$.

- $2C$ logistic regressions: each chain at $t-1$ acts as a categorical explanatory variable in predicting the state of a chain at t;
- lasso shrinkage with AIC selection of penalty weights;
- emission distribution parameters fitted as a mixture model via EM and left fixed;
- IRLS to fit regression parameters on a subsample of 50000 transitions;
- forward-backward to deterministically select hidden state sequence one chain at a time.

$$C = 39,\ N = 2,\ T = 15.4 \times 10^6$$

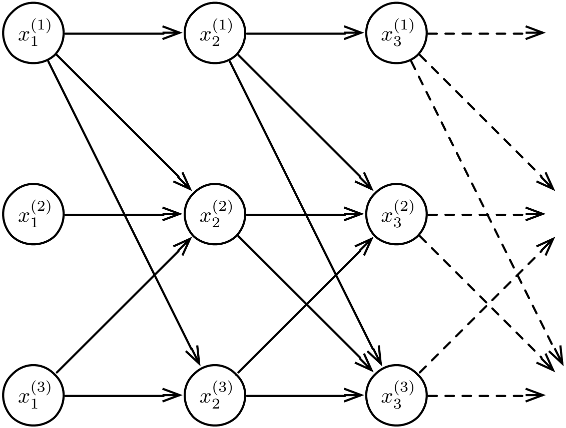
Summary of existing approaches

- Mixture model reduces number of parameters;
- Marginal composite likelihood also avoids need to store all the forward variables;
- Structured transition matrices per chain improves computation, but modelling becomes cumbersome as number of chains grows and assumes some scientific knowledge of the hidden process to achieve sparsity;
- Direct logistic regression transitions promise all these advantages, but has not been implemented in a principled statistical fashion.

However, none of these offer a solution to huge computational challenge of joint forward variable calculation as number of chains grows.

The question of interest

In the context of our genomics application, the primary question of interest is to infer the graphical structure which it is expected will be sparse.



The question of interest

In the context of our genomics application, the primary question of interest is to infer the graphical structure which it is expected will be sparse.

- use multinomial logistic/probit regression for transition probabilities $\implies$ $C(N-1)+1$ parameters instead of N^C;
- use blocked spike-and-slab prior construction to perform Bayesian variable selection as a means of inferring hidden layer structure;
- employ MCMC to properly quantify uncertainties.

The model

Observation model

$$Y_t^{(i)} \mid X_t^{(i)} \sim \text{Poisson}\left(\theta_{X_t^{(i)}}\right) \qquad \text{where } \theta_1 < \cdots < \theta_N$$

$$\theta_i = \sum_{j=1}^{i} \lambda_i$$

$$\lambda_i \sim \text{Gamma}(\alpha, \beta)$$

Hidden state model

$$X_t^{(i)} \mid X_{t-1}^{(1:C)} \sim \mathcal{M}\left(p_{t1}^{(i)}, \ldots, p_{tN}^{(i)}\right) \qquad \text{for } t \in \{2, \ldots, T\}$$

$$p_{tj}^{(i)} = \frac{\exp(\mathbf{x}_{t-1}^{(1:C)} \boldsymbol{\beta}_j^{(i)})}{\sum_{n=1}^{N-1} \exp(\mathbf{x}_{t-1}^{(1:C)} \boldsymbol{\beta}_n^{(i)})}$$

$$\beta_{jk}^{(i)} \sim \text{N}(0, v^2)$$

$$X_0^{(i)} \sim \mathcal{M}(N^{-1}, \ldots, N^{-1})$$

Top block Gibbs sampler

MCMC sampler from the block updates:

- Hidden states: conditional forward/stochastic-backward

$$\mathbf{X}_{1:T}^{(i)} \mid \boldsymbol{\beta}, \boldsymbol{\lambda}, \mathbf{Y}_{1:T}^{(i)}, \mathbf{X}_{1:T}^{(-i)} \quad \text{for} \quad i \in \{1, \ldots, C\}$$

- Multinomial logistic parameters (Holmes and Held, 2006)

$$\boldsymbol{\beta} \mid \mathbf{X}_{1:T}^{(1:C)}$$

- Observation model parameters

$$\boldsymbol{\lambda} \mid \mathbf{Y}_{1:T}^{(1:C)}, \mathbf{X}_{1:T}^{(1:C)}$$

Conditional forward/stochastic-backward

Use the modified conditional forward variable

$$\alpha_{tjk}^{(l)} = \mathbb{P}(y_t^{(l)}, X_{t-1}^{(l)} = j, X_t^{(l)} = k \,|\, \mathbf{y}_{1:t-1}^{(l)}, \mathbf{x}_{1:T}^{(-l)})$$

$$= \left(\sum_{i=1}^{N} \alpha_{(t-1)ij}^{(l)} \right) \frac{\exp(\mathbf{x}_{t-1}^{*j} \boldsymbol{\beta}_k^{(l)})}{1 + \sum_{n=1}^{N-1} \exp(\mathbf{x}_{t-1}^{*j} \boldsymbol{\beta}_n^{(l)})} \frac{\left(\sum_{n=1}^{k} \lambda_n \right)^{y_t^{(l)}}}{y_t^{(l)}!} e^{-\sum_{n=1}^{k} \lambda_n}$$

Conditional forward/stochastic-backward

Use the modified conditional forward variable

$$\alpha^{(l)}_{tjk} = \mathbb{P}(y^{(l)}_t, X^{(l)}_{t-1} = j, X^{(l)}_t = k \,|\, \mathbf{y}^{(l)}_{1:t-1}, \mathbf{x}^{(-l)}_{1:T})$$

$$= \left(\sum_{i=1}^{N} \alpha^{(l)}_{(t-1)ij}\right) \frac{\exp(\mathbf{x}^{*j}_{t-1}\boldsymbol{\beta}^{(l)}_k)}{1 + \sum_{n=1}^{N-1} \exp(\mathbf{x}^{*j}_{t-1}\boldsymbol{\beta}^{(l)}_n)} \frac{\left(\sum_{n=1}^{k} \lambda_n\right)^{y^{(l)}_t}}{y^{(l)}_t!} e^{-\sum_{n=1}^{k} \lambda_n}$$

Then, the simple factorisation:

$$\mathbb{P}(\mathbf{X}^{(l)}_{1:T} \,|\, \mathbf{y}^{(l)}_{1:T}, \mathbf{x}^{(-l)}_{1:T}) = \mathbb{P}(X^{(l)}_T \,|\, \mathbf{y}^{(l)}_{1:T}, \mathbf{x}^{(-l)}_{1:T}) \prod_{t=1}^{T-1} \mathbb{P}(X^{(l)}_{n-t} \,|\, \mathbf{X}^{(l)}_{n-t+1:T}, \mathbf{y}^{(l)}_{1:T}, \mathbf{x}^{(-l)}_{1:T})$$

Conditional forward/stochastic-backward

Use the modified conditional forward variable

$$\alpha_{tjk}^{(l)} = \mathbb{P}(y_t^{(l)}, X_{t-1}^{(l)} = j, X_t^{(l)} = k \,|\, \mathbf{y}_{1:t-1}^{(l)}, \mathbf{x}_{1:T}^{(-l)})$$

$$= \left(\sum_{i=1}^{N} \alpha_{(t-1)ij}^{(l)} \right) \frac{\exp(\mathbf{x}_{t-1}^{*j} \boldsymbol{\beta}_k^{(l)})}{1 + \sum_{n=1}^{N-1} \exp(\mathbf{x}_{t-1}^{*j} \boldsymbol{\beta}_n^{(l)})} \frac{\left(\sum_{n=1}^{k} \lambda_n\right)^{y_t^{(l)}}}{y_t^{(l)}!} e^{-\sum_{n=1}^{k} \lambda_n}$$

Then, the simple factorisation:

$$\mathbb{P}(\mathbf{X}_{1:T}^{(l)} \,|\, \mathbf{y}_{1:T}^{(l)}, \mathbf{x}_{1:T}^{(-l)}) = \mathbb{P}(X_T^{(l)} \,|\, \mathbf{y}_{1:T}^{(l)}, \mathbf{x}_{1:T}^{(-l)}) \prod_{t=1}^{T-1} \mathbb{P}(X_{n-t}^{(l)} \,|\, \mathbf{X}_{n-t+1:T}^{(l)}, \mathbf{y}_{1:T}^{(l)}, \mathbf{x}_{1:T}^{(-l)})$$

$\implies$ sampling is straightforward since:

$$\mathbb{P}(X_T^{(l)} = j \,|\, \mathbf{y}_{1:T}^{(l)}, \mathbf{x}_{1:T}^{(-l)}) = \sum_{i=1}^{N} \alpha_{Tij}^{(l)}$$

$$\mathbb{P}(X_{n-t}^{(l)} = i \,|\, X_{n-t+1}^{(l)} = j, \mathbf{y}_{1:T}^{(l)}, \mathbf{x}_{1:T}^{(-l)}) \propto \alpha_{(n-t+1)ij}^{(l)}$$

Holmes and Held (2006)

Uses an auxilliary variable method to ensure conditional conjugacy for logistic models and so automatic sampling without tuning parameters.

The variance component in the Normal latent model formulation of logistic regression has an additional Kolmogorov-Smirnov prior.

In this setting, each chain acts as a categorical predictor variable, contributing $(N-1)$ parameters to the model. Consequently, the design matrix is $(T-1) \times (1 + C(N-1))$.

Observation model

The setup of the observation model essentially follows technique in Scott (2002) to avoid ‘label switching’:

The Poisson parameters $\theta_1 < \cdots < \theta_N$ are based on cumulative sums of $\lambda_i = \theta_i - \theta_{i-1}$, so that when a chain is in state i there are additive Poisson contributions with rate $\lambda_1, \ldots, \lambda_i$.

Observation model

The setup of the observation model essentially follows technique in Scott (2002) to avoid 'label switching':
The Poisson parameters $\theta_1 < \cdots < \theta_N$ are based on cumulative sums of $\lambda_i = \theta_i - \theta_{i-1}$, so that when a chain is in state i there are additive Poisson contributions with rate $\lambda_1, \ldots, \lambda_i$.

Thus, if $x_t^{(l)} = i$ then observation $y_t^{(l)}$ is decomposed as the N vector $\left(y_{t1}^{(l)}, \ldots, y_{ti}^{(l)}, 0, \ldots, 0\right)$ with $\sum_{j=1}^{i} y_{tj}^{(l)} = y_t^{(l)}$ and $Y_{tj}^{(l)} \sim \text{Poisson}(\lambda_j)$. (i.e. 'regimes' $\leq i$ are active)

Observation model

The setup of the observation model essentially follows technique in Scott (2002) to avoid 'label switching':
The Poisson parameters $\theta_1 < \cdots < \theta_N$ are based on cumulative sums of $\lambda_i = \theta_i - \theta_{i-1}$, so that when a chain is in state i there are additive Poisson contributions with rate $\lambda_1, \ldots, \lambda_i$.

Thus, if $x_t^{(l)} = i$ then observation $y_t^{(l)}$ is decomposed as the N vector $\left(y_{t1}^{(l)}, \ldots, y_{ti}^{(l)}, 0, \ldots, 0\right)$ with $\sum_{j=1}^{i} y_{tj}^{(l)} = y_t^{(l)}$ and $Y_{tj}^{(l)} \sim \text{Poisson}(\lambda_j)$. (i.e. 'regimes' $\leq i$ are active)

Given $y_t^{(l)}$, the $y_{t\cdot}^{(l)}$ are simply a multinomial draw with total $y_t^{(l)}$ and probability vector proportional to $(\lambda_1, \ldots, \lambda_i)$.

With all $y_{t\cdot}^{(l)}$, the posterior of each λ_i follows easily as a simple conjugate posterior.

Implementation

Current implementation is $\approx 90\%$ in C++, using Armadillo linear algebra libraries.

Current breakdown of CPU usage in a typical small C, N and T run:

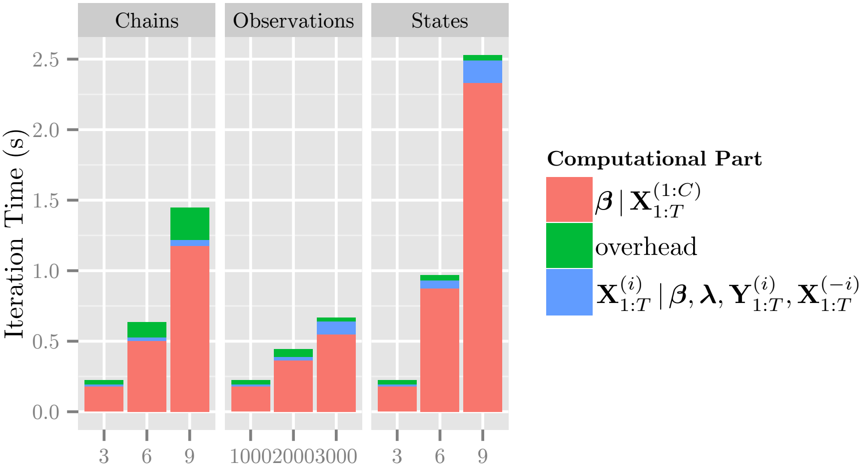
85.2% sampling $\boldsymbol{\beta} \mid \mathbf{X}_{1:T}^{(1:C)}$
of which:

- 60.8% sampling mixing weights;
- 18.8% matrix operations.
- 20.4% $\exp(\cdot)$, random number generation, ...

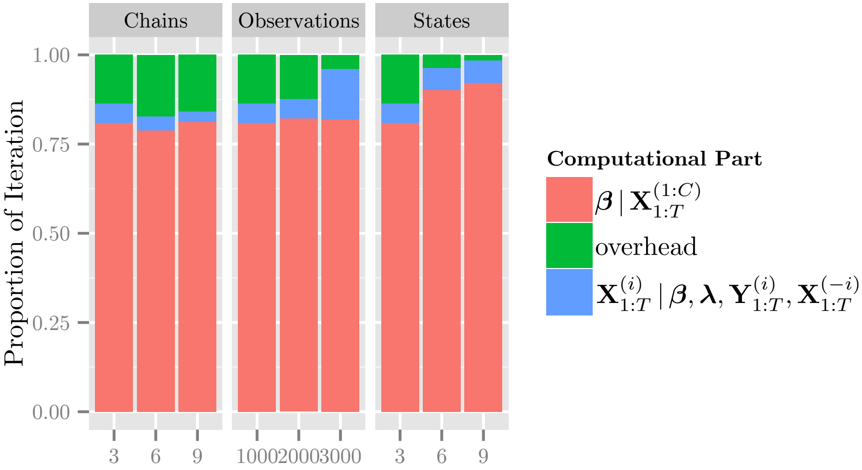
6.2% sampling $\mathbf{X}_{1:T}^{(i)} \mid \boldsymbol{\beta}, \boldsymbol{\lambda}, \mathbf{Y}_{1:T}^{(i)}, \mathbf{X}_{1:T}^{(-i)}$ for $i \in \{1, \ldots, C\}$

8.6% overhead (easy to remove in due course).

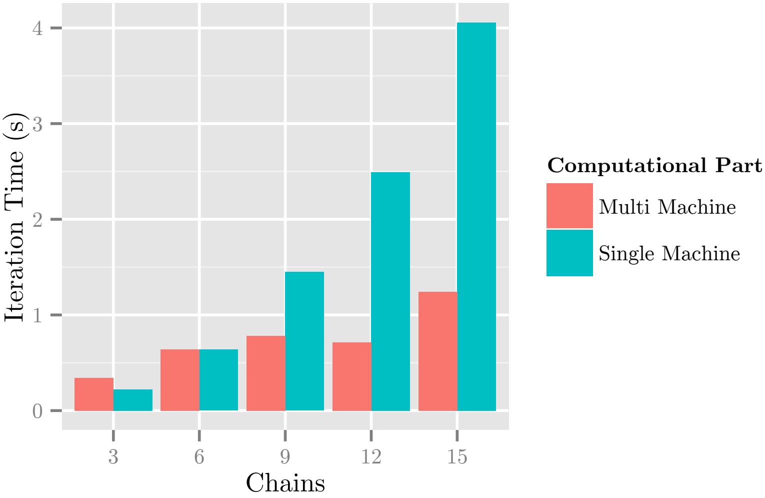
Growth in C, T and N (time)



Growth in C, T and N (proportion)



‘Free’ parallelism in $\boldsymbol{\beta} \mid \mathbf{X}_{1:T}^{(1:C)}$



Alternative dynamic programming

Are there better approaches to the dynamic programming when dealing with joint forward variables?

Do we really need the forward variable or are there other dynamic programming methods of getting at the likelihood in this model?

e.g.

$$\mathbb{P}(\mathbf{y} \mid \boldsymbol{\theta}) = \sum_{i=1}^{N} \alpha_T^{(C)}(i) \text{ where } \alpha_t^{(k)}(i) = \mathbb{P}(\mathbf{y}_{1:t}^{(1:k)}, x_t^{(k)} = j)$$

$$\mathbb{P}(\mathbf{y} \mid \boldsymbol{\theta}) = \prod_{k=1}^{C} \sum_{i=1}^{N} \alpha_T^{(k)}(i) \text{ where } \alpha_t^{(k)}(i) = \mathbb{P}(\mathbf{y}_{1:t}^{(k)}, x_t^{(k)} = j \mid \mathbf{y}_{1:t}^{(1:k-1)})$$

$$\mathbb{P}(\mathbf{y} \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} \psi(t) \text{ where } \psi(t) = \mathbb{P}(\mathbf{y}_t^{(1:C)} \mid \mathbf{y}_{1:t-1}^{(1:C)})$$

etc etc ...

Blocked conditional forward/stochastic backward

Sampling using the conditional forward/stochastic backward should of course be blocked.

- Need to find a partition of chains $\mathcal{J}_i$ which is in some sense 'optimal':
  - coarser $\implies$ better mixing, higher memory, higher compute
  - finer $\implies$ poorer mixing, lower memory, lower compute
- Partition may vary one iteration to the next depending on graphical structure chaging, so decision algorithm must be efficient too.
- Extent of mixing issue varys from data set to data set.

Is there a principled metric for deciding how to block?

Mixture model

Perhaps improve upon the mixture model approach already seen in the literature for small problems.

$$\mathbb{P}(X_t^{(i)} \mid x_{t-1}^{(1:C)}) = \sum_{k=1}^{C} \omega_{ki}\, \mathbb{P}(X_t^{(i)} \mid x_{t-1}^{(k)})$$

An auxilliary variable MCMC scheme where a single additional chain is selected to have influence and estimate ω_{ki} as the empirical distribution of how often chain k is selected to influence chain i in the sampling.

Developing a scheme which enforces sparsity in the number of non-zero ω_{ki} would be useful.

Graphical models literature

Much computational pain is due to lack of sparsity. Can we learn the structure before (or as part of) the inference?

Large literature on learning structure of graphical models, but only with observed data(!) e.g. PC algorithm. Could they be adapted?

Or, there are many options for variable selection if sticking with a logistic transition model, such as spike-and-slab priors, Bayesian lasso, etc.

Improve current multinomial logistic regression

Adopt more recent logistic regression sampling schemes (e.g. Polson *et al.*, 2013).

Since we're not that interested in the value of the coefficients (save for equality to zero for structure), drop logistic regression and use the probit link instead for faster computation (recall: $\approx 52\%$ of computation time on sampling from KS distribution).

Other less expensive classification methods which fit into the framework nicely?

In particular, want fast mixing samplers because want a single sample from stationarity of transition model.

Spike-and-slab prior

A spike-and-slab prior formulation which enforces sparsity in the logistic models fitted provides a natural way of learning the graphical structure in the hidden layer.

This requires a Lebesgue measure zero spike, rather than the now more common Ishwaran and Rao (2005) formulation. How to handle Lindley (1957) paradox?

Too many models to compute model posterior directly as in Mitchell and Beauchamp (1988). Natural ordering on models for a Metropolis jump?

NB: crucially important that whatever formulation is used it includes/excludes entire blocks of parameters: can't have just some states in another chain having effect.

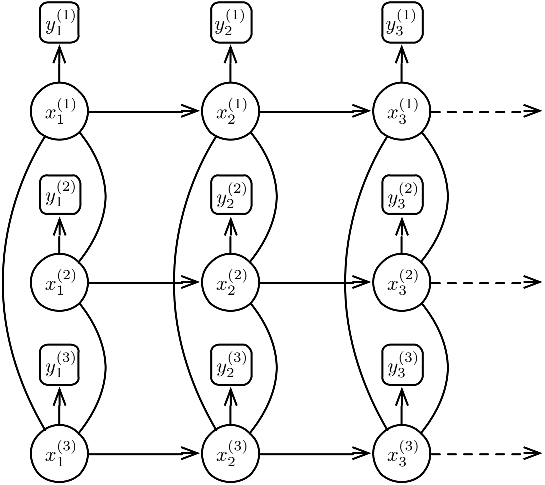
Missing covariate regression methods

There are methods for using prior predictive distributions on missing covariates in regression problems ...

... although here everything (including response!) is missing.

Something else entirely?

In terms of application, we really believe the relationship is a chain graph:



An exciting paper from MCMSki

If time permits, a short discussion of Pakman and Paninski (2013)

References I

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970), 'A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains', *The Annals of Mathematical Statistics* **41**(1), 164–171.

Choi, H., Fermin, D., Nesvizhskii, A. I., Ghosh, D. and Qin, Z. S. (2013), 'Sparsely correlated hidden Markov models with application to genome-wide location studies', *Bioinformatics* **29**(5), 533–541.

Holmes, C. C. and Held, L. (2006), 'Bayesian auxiliary variable models for binary and multinomial regression', *Bayesian Analysis* **1**(1), 145–168.

Ishwaran, H. and Rao, J. S. (2005), 'Spike and slab variable selection: frequentist and Bayesian strategies', *The Annals of Statistics* **33**(2), 730–773.

Lindley, D. V. (1957), 'A statistical paradox', *Biometrika* **44**(1), 187–192.

Mitchell, T. J. and Beauchamp, J. J. (1988), 'Bayesian variable selection in linear regression', *Journal of the American Statistical Association* **83**(404), 1023–1032.

Pakman, A. and Paninski, L. (2013), 'Auxiliary-variable exact Hamiltonian Monte Carlo samplers for binary distributions', *arXiv* (1311.2166).

Polson, N. G., Scott, J. G. and Windle, J. (2013), 'Bayesian inference for logistic models using Pólya-Gamma latent variables', *Journal of the American Statistical Association* .

Saul, L. K. and Jordan, M. I. (1999), 'Mixed memory Markov models: decomposing complex stochastic processes as mixtures of simpler ones', *Machine Learning* **37**(1), 75–86.

Scott, S. L. (2002), 'Bayesian methods for hidden Markov models: Recursive computing in the 21st century', *Journal of the American Statistical Association* **97**(457), 337–351.

Sherlock, C., Xifara, T., Telfer, S. and Begon, M. (2013), 'A coupled hidden Markov model for disease interactions', *Journal of the Royal Statistical Society, Series C* **62**(4), 609–627.

Zhong, S. and Ghosh, J. (2002), HMMs and coupled HMMs for multi-channel EEG classification, *in* 'Proceedings of the International Joint Conference on Neural Networks', Vol. 2, pp. 1154–1159.