Understanding Protein Dynamics with L_1-Regularized Reversible Hidden Markov Models

Robert T. McGibbon, Bharath Ramsundar, Mohammad M. Sultan, Gert Kiss, Vijay S. Pande

Proceedings of the 31st International Conference on Machine Learning

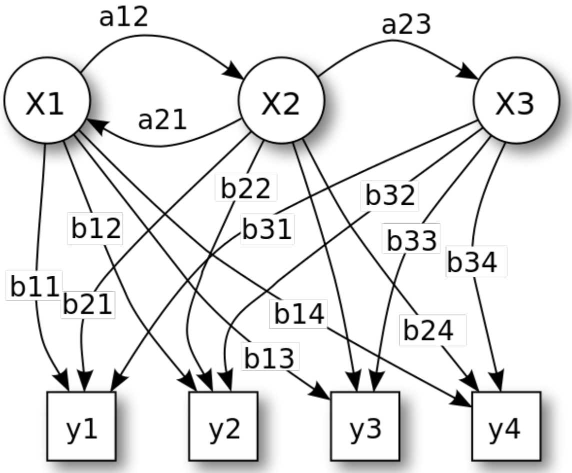
6 May 2014

Overview

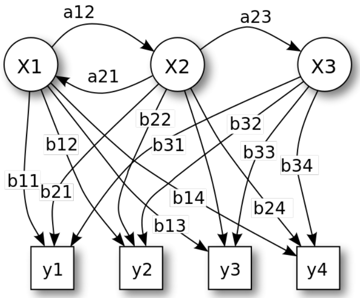
From MD Trajectories to Hypotheses

- Proteins constitute high-dimensional systems
- The majority of those dimensions provide no meaningful information about major conformational changes
- We want to reduce the dimensionality to include only those features that provide information about a small number of major states, and their transition rates

HMM Background

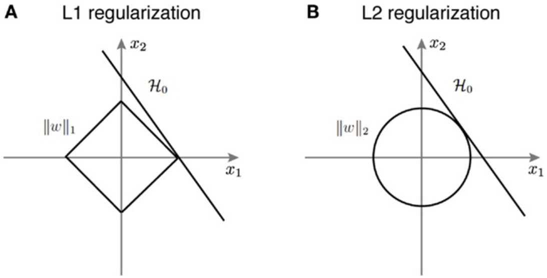


HMM and MD



This formulation applies naturally to analyzing MD trajectories - multiple microstates should correspond to the same macrostate

L_1-Regularization

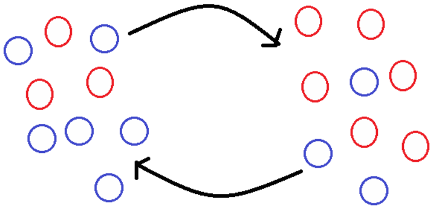


L_1 regularization can be used for feature selection - produces a sparse result because the constraint is able to drive some weights to 0. Because L_2 regularization is rotationally invariant, no benefit can be derived from searching for an extremum along a specific axis; L_1 provides a clear benefit associated with reducing the number of axes along which we search.

L_1-Regularization

The L_2-regularized loss function $F(x) = f(x) + \lambda||x||_2^2$ is smooth; the optimum is a stationary point, which becomes smaller as λ increases but it won't be 0 unless $f'(0) = 0$.
The L_1-regularized loss function $F(x) = f(x) + \lambda||x||_1$ is not smooth, and it is not differentiable at 0. The optimum of a function is either the point with a derivative of 0 or an irregularity (a corner or kink), so the optimum of the L_1-regularized loss function may be 0 even if that is not a stationary point.
The authors further facilitate sparsity by including adaptive weights that increase the penalty for non informative degrees of freedom.

Detailed Balance



No net flows!

$$\forall k, k', \pi_k T_{k,k'} = \pi_{k'} T_{k',k}$$

where π is the stationary distribution of T.

Basic EM

Given Y observed data, X missing data, and unknown parameters θ, we can compute a maximum likelihood estimate of the parameters using the marginal likelihood of the observed data

$$L(\theta; Y) = p(Y|\theta) = \sum_{X} p(Y, X|\theta)$$

If X is a sequence of events (e.g. state transitions), this likelihood grows exponentially with the length of the sequence. Instead we can search for the maximum likelihood estimate iteratively:
E step: Calculate the expected value of the log likelihood function using the current estimate of the parameters

$$Q(\theta|\theta^{(t)}) = E_{X|Y,theta^{(t)}}[logL(\theta; Y, X)]$$

M step: Find the parameter values that maximize Q

$$\theta^{(t+1)} = argmaxQ(\theta|\theta^{(t)})$$

Their M-Step

$$Q_P(\theta, \theta^{(t)}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_k(i) log \phi(x_i; \mu_k, \Sigma) - \lambda \sum_{k,k'} \sum_{j} \tau_{k,k'}^{(j)} |\mu_{k,j} - \mu_{k',j}|$$

where $\phi(x_i; \mu_k, \Sigma_k)$ is a Gaussian density with mean vector $\mu_k = (\mu_{k,1}, ..., \mu_{k,p})$ and covariance matrix Σ_k.
We want to update the parameter estimates via $\theta^{(t+1)} = argmax Q_p(\theta, \theta^{(t)})$, equivalent to

$$\mu_k^{(t+1)} = \underset{\mu_k}{\operatorname{argmin}} \sum_{i}^{N} \sum_{k}^{K} \gamma_k(i) \frac{(x_i - \mu_k)^2}{2(\sigma_k^2)^{(t)}} + \lambda \sum_{k,k'} \sum_{j} \tau_{k,k'}^{(j)} |\mu_{k,j} - \mu_{k',j}|$$

Timescale

We care about the slow dynamics, corresponding to observable dynamical modes, given by

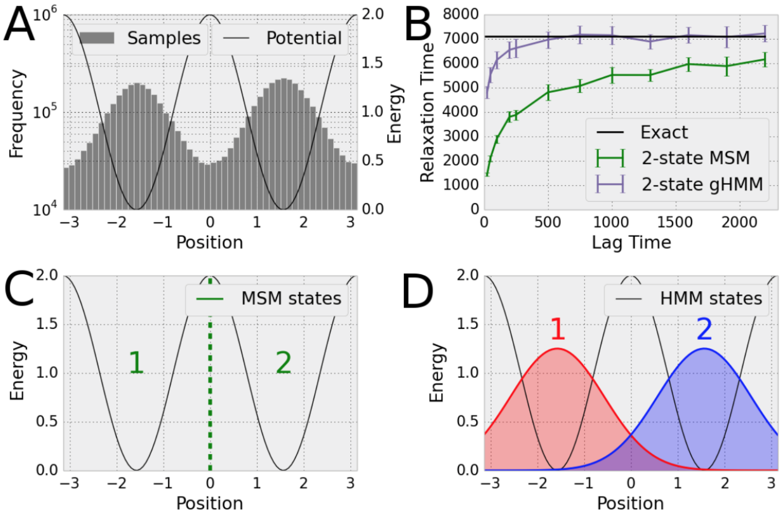
$$\tau_i = -\frac{1}{ln \lambda_i}$$

Thus the HMM framework enables us to estimate physical rates so we can predict values such as the mean passage time.

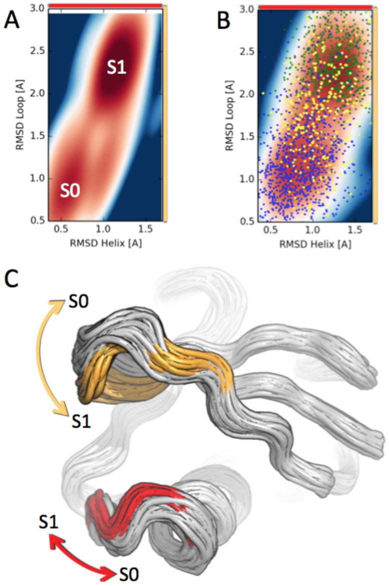
GPU Implementation is Best

- GPU implementation is 15X faster than CPU parallelized implementation
- Double-precision required to avoid floating point errors during computation of the forward-backward algorithm

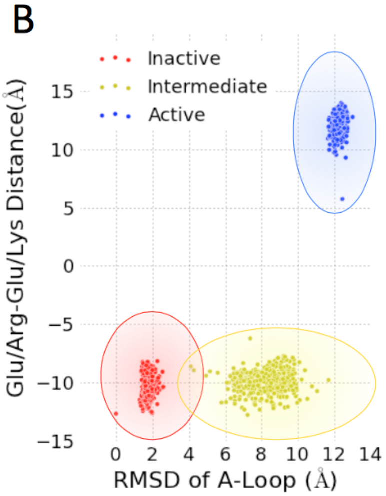
Double Well Potential



Ubiquitin



c-Src Tyrosine Kinase



The End