Using HaMMLET

John Wiedenhoeft, Eric Brugel, Alexander Schliep

February 12, 2016

This document describes the usage of HaMMLET on the biorxiv branch at `https://github.com/wiedenhoeft/HaMMLET/tree/biorxiv`. The paper describing the method has been accepted for oral presentation at RECOMB 2016[1]; a preprint can be found at `http://biorxiv.org/content/early/2015/07/31/023705`.

1 Introduction

HaMMLET is a general-purpose implementation of a Bayesian Hidden Markov Model (HMM) [1]. In contrast to frequentist HMM using the Baum-Welch algorithm [2, 3] for parameter estimation and the Viterbi path [4] to derive the most likely state sequence, HaMMLET provides a full marginal distribution of states for each data point, thus allowing for alternative interpretations of the data. This is achieved using a Markov Chain Monte Carlo technique called Forward-Backward Gibbs sampling [5, 6], which is typically computationally expensive. HaMMLET therefor uses the Haar wavelet transform to dynamically compress the data into different levels of spatially localized resolution (in plain English, it dynamically "zooms in" on what it considers "interesting" regions and treats "boring" regions in a summary fashion), thereby drastically reducing running times and improving convergence behavior. The name of the software is a portmanteau of "HMM" and "wavelet".

Presently, only univariate Gaussian emissions are supported, additional functionality for multivariate data and non-Gaussian emissions will be provided in the near future. A typical biological application for HaMMLET would be the segmentation of array-based Comparative Genomic Hybridization (aCGH) data into segments that can be explained statistically as being noisy measurements of a shared underlying $\log_2$ intensity ratio. Segments of different $\log_2$ indicate the presence of genomic copy-number variants.

2 Preliminaries and Installation

Requirements HaMMLET is implemented in C++ and should compile on all modern platforms (Linux, Mac, Windows). The wrapper script requires Python[2] (at least

[1] `http://recomb2016.bioinformatics.ucla.edu/accepted-papers/`
[2] `https://www.python.org/`

version 2.6, tested with 2.7.3 and 2.7.6) with NumPy[3] (tested with 1.6.2 and 1.8.2). For plotting, Matplotlib[4] is required (tested with 1.1.1 and 1.3.1).

HaMMLET is distributed as a Git[5] repository and currently hosted on GitHub (`https://github.com/wiedenhoeft/HaMMLET/tree/biorxiv`). It is recommended to have Git installed, in order to benefit from all the advantages of a distributed version control system. To pull the repository, open a shell and run

```
git clone https://github.com/wiedenhoeft/HaMMLET.git
cd HaMMLET
git checkout biorxiv
make .
cd evaluation
```

Alternatively, if Git is not installed, ignore the first line and download the files directly as a zip archive from `https://github.com/wiedenhoeft/HaMMLET/zipball/biorxiv` into a directory called HaMMLET instead.

After running the commands above, the directory contains a compiled executable named HaMMLET. However, this file should not be called directly. Instead, the directory contains a Python wrapper script (hammlet.py) which allows for conveniently passing options to the executable, handles the plotting etc.; to get a short manpage, simply run

```
python hammlet.py -h
```

3 Data Preparation

The input to HaMMLET consists of s simple text file containing two tab-separated columns. The first column is ignored by HaMMLET and can be used for annotations such as probe identifiers etc.; the second column should contain a floating-point number representing the value of a data point, cf. the included file `sample.csv` for example:

```
0 0.623454
1 -0.538821
2 0.092649
3 -0.155161
4 0.456472
5 -0.748464
6 0.362765
7 -0.189305
8 -0.409891
9 0.372456
```

To pass model hyperparameters, a model file can be used. For instance, HaMMLET includes an example (`sample_model.txt`) to be used with `sample.csv`:

[3] `http://www.numpy.org/`
[4] `http://matplotlib.org/`
[5] `https://git-scm.com/`

```
AutoNormal,.2,.9,s1
AutoNormal,.2,.9,s2
AutoNormal,.2,.9,s3
AutoNormal,.2,.9,s4
AutoNormal,.2,.9,s5
AutoNormal,.2,.9,s6
Categorical,6,100,10,10,10,10,10,a1
Categorical,6,10,100,10,10,10,10,a2
Categorical,6,10,10,100,10,10,10,a3
Categorical,6,10,10,10,100,10,10,a4
Categorical,6,10,10,10,10,100,10,a5
Categorical,6,10,10,10,10,10,100,a6
Categorical,6,10,10,10,10,10,10,pi
HMM,6,pi,a1,a2,a3,a4,a5,a6,s1,s2,s3,s4,s5,s6
```

The last line defines an HMM in terms of the number of states (`6`), followed by an identifier for the initial state distribution (`pi`), identifiers for the rows of the transition matrix (`a1,...,a6`) *in order* (the i-th entry is the row representing transitions out of state i), and emission states (`s1,...,s6`); the number of matrix rows and states must match the one given after `HMM`. Identifiers can be assigned arbitrarily, but must be unique for each entry.

The other lines define those entities; note that at the end of each line, the identifiers (`id`) used below are assigned. These can be arbitrary alphanumeric strings, but must be unique for each line.

```
AutoNormal,var,p,id
```

defines a Normal distribution `id` using automatic priors with $P(\sigma^2 \leq \texttt{var}) = \texttt{p}$. Alternatively, a Normal-Inverse Gamma prior NIΓ($\mu_o = \texttt{mu0}$, $\nu = \texttt{nu}$, $\alpha = \texttt{alpha}$, $\beta = \texttt{beta}$) can be assigned manually using

```
Normal,mu0,nu,alpha,beta,0,id
```

The penultimate entry 0 indicates whether `mu0` represents an actually known mean; this is used for testing and debugging purposes, and the value should always be set to 0.

```
Categorical,M,alpha1,...,alphaM,id
```

defines a Dirichlet prior of dimension `M` (the number of states) with hyperparameters $(\alpha_1, \ldots, \alpha_M) = (\texttt{alpha1}, \ldots, \texttt{alphaM})$.

With the data and model files in place, we can now run HaMMLET using

```
python hammlet.py -s6 -i 100 sample_model.txt sample.csv
```

`-i` denotes the number of iterations for the sampler, and $-s6$ denotes that the model uses 6 states[6]. Instead of using a model file, hyperparameters can be passed to the command line directly if all states are supposed to use automatic priors, and all

[6] Note that this option will be deprecated in future releases.

self-transitions, other transitions and initial state hyperparameters are each equal, using the option

```
-a M V P S T I
```

to create a model with `M` states, automatic priors $P(\sigma^2 \leq \mathtt{V}) = \mathtt{P}$, prior transitions `T`, prior self-transitions `S` and initial state prior (`I`, ..., `I`). For instance, instead of using the model file above, we could simply run

```
python hammlet.py -i 100 -a 6 0.2 0.9 100 10 10 \
   modelout.txt sample.csv
```

indicating that the model file `modelout.txt` should be created automatically from the values passed to the option `-a`. Notice that HaMMLET, by virtue of compression, is rather robust against choices of hyperparameters for the initial state distribution (`pi`) and transitions (`a1,...,a6`), so the last three of those numbers don't matter too much, unless they are very high, indicating strong prior belief.

4 Output

HaMMLET's output is pretty straightforward. For the example above, it creates a file `sample_statesequence.csv` (notice that the prefix `sample` derives from the input filename, `sample.csv`), containing a matrix representation of the marginal counts for each state, separated by whitespace. Each line represents one state, in the order specified in the model file, and columns contain their respective counts for each position[7]. Another file, `sample_last_statesequence.csv` contains the last sampled state sequence as a single whitespace-separated row. Lastly, `sample_compression_ratio.csv` contains the average compression ratio during sampling.

5 Plotting

The result of a sampling run can be plotted using the option `-p`, cf. Fig. 1. Each state is represented by a color. The top graph shows a scatterplot of the data itself, and each point is labeled with the color of the state with the highest marginal probability. The bottom graph shows the marginal distribution of each state; it is a graphical representation of the matrix in `sample_statesequence.csv`.

Sometimes, one might wish to get a more detailed picture of what is going on during sampling. The `-P` (uppercase) option outputs all sampled values for the parameters specified in the model file, in text files such as `sample_s3.txt`, which contains the sampled means and variances for state `s3`. These are then plotted to a file called `sample_parameters.png` (Fig. 2), which allows to assess the state of convergence of the Gibbs sampler.

[7] In future releases, this matrix is likely to be transposed, so that columns represent the states. It has been kept in this version for reproducibility.

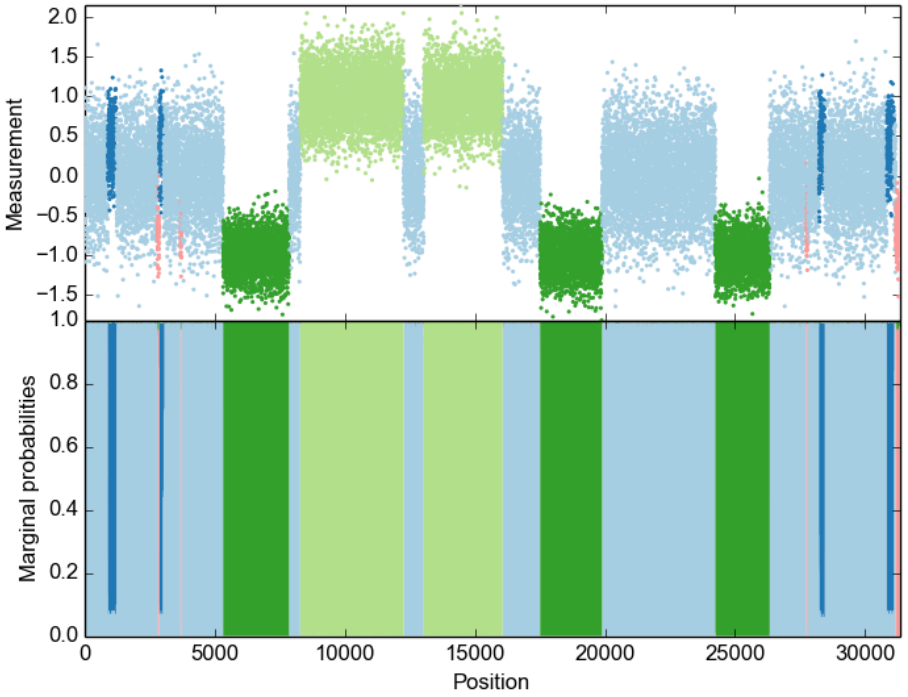

Figure 1: Example of HaMMLET's output plot (option `-p`) for `sample.csv`.

6 Future releases

Future versions of HaMMLET will include support for multivariate data, non-Gaussian emissions and different priors. Specialized implementations for SNP arrays, read-depth data from exome sequencing and WGS are underway. We also plan to implement Dirichlet process priors, so the number of states does not have to be set in advance. While HaMMLET appears to work well treating aCGH data as equidistant, more advanced transition kernels are also planned. Notice that the input format and command line options are likely to change to reflect these extensions.

References

[1] Baum LE, Petrie T. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. The Annals of Mathematical Statistics. 1966;37(6):1554–1563. Available from: `http://www.jstor.org/stable/2238772`.

[2] Bilmes J. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models; 1998. Available from: `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.613`.

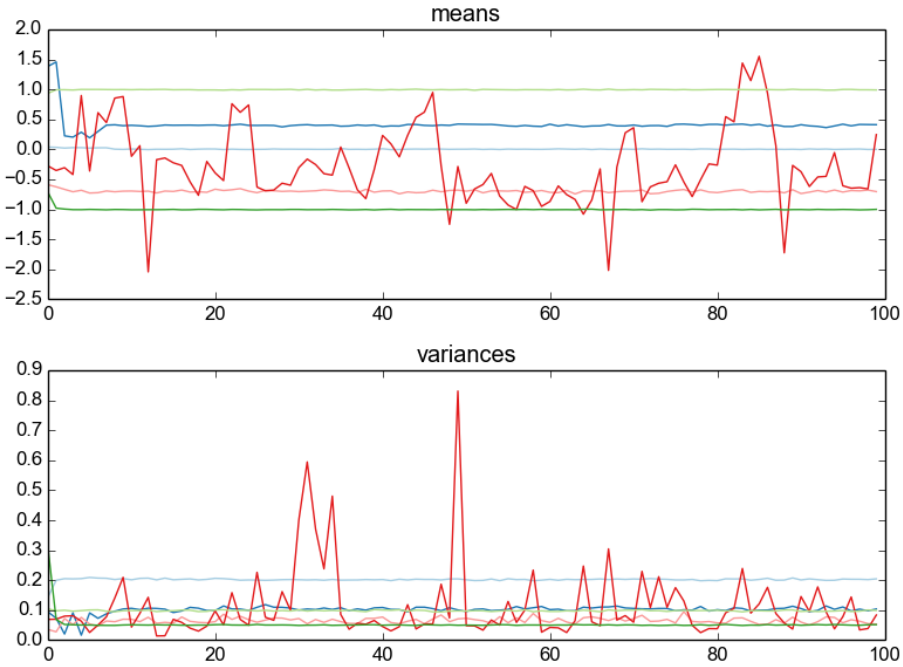

Figure 2: Example of parameter plots using the –P option, for running HaMMLET with a 6-states on data containing only 5 true states (Fig. 1). Notice that five of the means and variances converge almost instantly, while the sixth moves around randomly. This indicates that no observations are assigned to this state and it is sampled solely from its prior. In other words, superfluous states are ignored in HaMMLET (or any Bayesian HMM for that matter).

[3] Rabiner LR. A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE. 1989;77:257–286. Available from: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=18626`.

[4] Viterbi A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory. 1967 Apr;13(2):260–269. Available from: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1054010`.

[5] Chib S. Calculating posterior distributions and modal estimates in Markov mixture models. Journal of Econometrics. 1996;75(1):79–97. Available from: `http://www.sciencedirect.com/science/article/pii/0304407695017704`.

[6] Scott SL. Bayesian Methods for Hidden Markov Models: Recursive Computing in the 21st Century. Journal of the American Statistical Association. 2002;97(457):337–351. Available from: `http://www.jstor.org/stable/3085787`.