The IMMpractical library

Michael E Sparks

July 22, 2006

1 Description

The `IMMpractical` library currently implements various training and testing methods for the appraisal of coding potential in DNA sequences. Top-down and bottom-up deleted interpolated, χ^2-interpolated, and fixed order Markov model methods are implemented. Noncoding sequences are exclusively modeled using a homogeneous Markov chain, whereas coding sequences can be modeled using either a non-periodic (homogeneous) or three-periodic (inhomogeneous) approach, the latter being either restricted to the forward strand only, or optionally modeling both forward and shadow (reverse) strands. Markov chain orders can freely range from 0 to 5 for the above methods.

Version 1.0 has introduced dynamically modulating Markov models and partially modulating mixed models for Markov chains strictly of order 5. Support for G+C composition-based quantile-specific sequence analysis was also added (again, for Markov chains strictly of order 5), wherein sequences can be assayed using quantile-specific transition probability estimates based on overall G+C composition (or that of a sliding window of 100 nucleotides) of a test sequence—quantile-specific behavior is available for all training/testing methods implemented in the library.

Sequence classification methods based on maximum likelihood hypotheses, maximum *a posteriori* (MAP) hypotheses, or the familiar Genmark algorithm of Borodovsky and McIninch are implemented in the library driver program, `use_MM.x`.

2 Getting Started

Noncoding and coding training data is provided for *Arabidopsis* in the training_data/Arabidopsis/ directory. (Actually, these data must be obtained independently of the library sources from the SourceForge project download site. Unpack the training data tarball in the same directory where the source files were unpacked, and they will be placed in the appropriate node of your source tree automatically.) Here, we illustrate a sample application of this software by generating transition probability estimates for *Arabidopsis* using the top-down deleted interpolation method, and then appraising coding potential of some test sequences with these. Assuming your current directory is the root directory of the `IMMpractical` package,

```
cd ./src && make
make dataunzip
cd ..
```

This uncompresses the packaged training data ('make datazip' repacks it) and builds the `train_MM.x` and `use_MM.x` binaries in the bin/ directory.

```
./bin/train_MM.x 1 ./training_data/Arabidopsis/D_data/file0.D \
                   ./training_data/Arabidopsis/D_data/file1.D \
                   ./training_data/Arabidopsis/H_data/file0.H \
                   ./training_data/Arabidopsis/H_data/file1.H \
                   ./parm_files/Arabidopsis | less
```

This call invokes the `train_MM.x` program to generate a binary output file, `Arabidopsis.TDDI` in parm_files/, which stores the top-down DIMM-based parameterization. stdout feedback is piped to the `less` pager so you can track progression of the model's development. Invoking `train_MM.x` (and `use_MM.x`) without command line parameters will elicit messages concerning usage conventions, numeric codes for the various training algorithms, etc.

```
cat training_data/Arabidopsis/T_data/file0.T | \
  ./bin/use_MM.x 1 parm_files/Arabidopsis.TDDI | less
```

This call invokes the `use_MM.x` program to use the transition probability estimates obtained in the previous step to assess the coding potential of the test data.

3 Hints for extending/modifying the software

By default, the software is configured to model noncoding sequences using homogeneous Markov models, and both strands of coding sequences using inhomogeneous chains, in a non-quantile-specific manner, all with a moderate degree of verbosity. For all other applications, see the discussion on preprocessor definitions in the Makefile to select the appropriate combination for your task. (Yes, the Makefile serves as a primary source of documentation for the library, for better or worse!)

There is code to facilitate development of training/testing data in the training_data/utilities/ directory. The `clean_data.pl` script can be used to clean out undesirable sequences in any candidate training or testing data (undesirable inasmuch as the library is concerned), as well as ensure that each sequence of the resulting Fasta file is entirely on one line, a requirement for the included sampling script, `data_partition.sh`. Tune the $FILES variable in `data_partition.sh` so that it locates and processes each model's input sequence file, if necessary; this script will non-destructively split the input sequences into development, held-out, and test partitions of sizes 2/5, 1/10, and 1/10 of the original input file, respectively, sampling randomly and without replacement.

4 Support

This code has been tested in FreeBSD 5.3 and a variety of recent GNU/Linux environments, both 32- and 64-bit; I anticipate the software is portable to most systems with a C compiler supporting C99 extensions including the isinf, isnan, and INFINITY macros. Since it isn't exactly the kind of software very many people will be interested in, if you have questions, concerns, comments, etc., please feel free to contact me at `mespar1@iastate.edu`. Frankly, I'd be interested in knowing who is using the code (and possibly even for what purpose?), period, so feel free to contact me even if you don't have questions! Good luck!