Nature Methods

PhymmBL expanded: confidence scores, custom databases, parallelization and more

Arthur Brady & Steven Salzberg

Supplementary Table 1	New features added to PhymmBL
Supplementary Table 2	List of genomes added to PhymmBL for the woolly mammoth classification experiment
Supplementary Note 1	Comparisons to recent methods on classification of short reads
Supplementary Note 2	Parallelization
Supplementary Note 3	Computational resources used in the classification of the mammoth bone metagenome
Supplementary Note 4	Usage notes and algorithm details for confidence scores

Supplementary Table 1: New features to make PhymmBL more flexible and easier to use include the following. Beyond the major items listed here, a collection of minor bug fixes and performance improvements is documented in the release history on PhymmBL's website.

▪ **Update RefSeq data:** The initial release of the program allowed users to update their local database only by completely overwriting the existing data with a current copy of RefSeq's bacterial and archaeal collection, rebuilding all IMMs, recomputing all supplementary data structures and taxonomic metadata from scratch. New options have been added to the installer script to give users the option to refresh their local databases with only those elements of the current RefSeq collection that have been added or updated since they first installed (or last updated) the software, greatly easing computational and temporal burdens associated with maintaining a current local database.
▪ **Mac OS** is now formally supported.
▪ A script to **manually regenerate the local BLAST database** has been included in the distribution. This is useful for users who want to create models for multiple custom genomes without rebuilding the BLAST database after each addition.
▪ The output format of the Phymm component of PhymmBL has now been coded so as to **reduce the size of raw output files by more than an order of magnitude** over the original.
▪ We have standardized PhymmBL's internal, ad hoc system for resolving discrepancies between species names and taxonomic data given in GenBank sequence files and the corresponding data present in the global NCBI taxonomy, from which the taxonomic labels used in PhymmBL's predictions are ultimately derived.

Supplementary Table 2: List of genomes added to PhymmBL for the woolly mammoth classification experiment.

Species	Common name

Arabidopsis thaliana	Thale cress
Canis lupus familiaris	Dog
Equus caballus	Horse
Homo sapiens	Human
Mus musculus	House mouse
Oryza sativa	Rice
Saccharomyces cerevisiae	Yeast

Supplementary Note 1: Comparisons to recent methods on classification of short reads

Two recent papers[4,5] describe new methods (or new versions of existing methods) aimed at classification of short reads. In most cases and for most read lengths, PhymmBL remains more accurate than these methods. In one recent paper[4], a new method called RAIphy was introduced and compared to PhymmBL. Fig. 5 reports that PhymmBL performs better for read lengths of 100 and 400 bp, and that a very slight improvement over PhymmBL was observed for RAIphy for lengths of 800 and 1000 bp. The absolute accuracy numbers they present for PhymmBL are considerably lower than we reported in our original paper[1], however, and it is unclear to us why the authors were unable to obtain better results than what was reported.

In another paper describing a new version of PhyloPythia[2] called PhyloPythiaS[4], Fig. 1 (comparing PhymmBL, MEGAN[3], and PhylopythiaS), shows that PhymmBL clearly outperforms MEGAN on longer contigs (with respect to accuracy reported for shorter read lengths, please see below for a discussion of comparisons between PhymmBL and other methods which don't label all input reads). For short reads, Supplementary Tables 1 and 13 show again that PhymmBL outperforms the PhylopythiaS system on many read lengths and for multiple levels of the phylogenetic classification system: e.g., in Supplementary Table 1, describing results on simulated acid mine data, PhymmBL outperforms both MEGAN and PhylopythiaS at the family, order, class, and phylum levels. Most of the other data in the paper describing PhyloPythiaS refers to results from long contigs, which are tens to hundreds of times longer than single reads. Single reads constitute the optimal target length for ab initio metagenomics analysis – that is to say, analysis that doesn't require assembly prior to classification. This is the length for which PhymmBL was explicitly designed.

On accuracy comparisons in general, we would like to address some confusion that has repeatedly arisen since PhymmBL's release, due in large part to two common misapprehensions of PhymmBL's design philosophy. The first issue concerns comparisons between other methods and Phymm (not PhymmBL). Phymm was never intended for use by itself: as we made clear in the article describing PhymmBL's release, Phymm, alone, does not classify reads as accurately as BLAST does. The only reason to use Phymm at all, then, is that in concert with BLAST – i.e., when used as part of the hybrid method PhymmBL – better accuracy is produced than when using either method on its own. Accuracy comparisons to Phymm alone are therefore misleading at best. The second cause of confusion concerns reported accuracy comparisons between PhymmBL, which labels all input reads, and methods like MEGAN, which do not: below some confidence threshold, reads classified by the latter group of methods are labeled "unclassified" or "unknown." It should be obvious that a direct comparison between PhymmBL and these methods is a comparison between two sets of results which are not of the same fundamental type, and are, again, misleading at best. PhymmBL now produces (and has produced, via up-to-date stable software versions available on the website since mid-2010) confidence scores for all predictions. Canonical use of PhymmBL **requires** that researchers (through methods which are explicitly and deliberately left to individual users, in order to meet their local research goals, idiosyncratic statistical preferences, and confidence constraints) identify a threshold within the range of confidence scores PhymmBL produces, below which all labels **must** be considered meaningless. Only once this threshold has been selected – and attention has been restricted solely to predictions whose confidence scores exceed this threshold – can comparisons to methods like those mentioned above be considered fair.

Supplementary Note 2: Parallelization

After extensive testing done both in-house and on other systems by PhymmBL users, we are now formally supporting parallel processing (specifically, simultaneous batch processing) of input reads. If you have multiple processors available and wish to parallelize your classification runs, scoreReads.pl is designed to allow multiple copies to run in parallel; you can split your input data (FastA sequences) into as many files as your system can comfortably handle, then run scoreReads.pl separately (and simultaneously) on each one. This will substantially improve processing time.

Example scenario: Your sequence data is in a file called "queryReads.fasta". You have six processors available on your system, and you want to use, say, four of them for classification.

Break up "queryReads.fasta" into four roughly equally-sized multi-FastA files, say "queryReads_1.fasta", "queryReads_2.fasta", etc., one for each processor to be used. Then run scoreReads.pl separately on each of the four input chunks, e.g.:

```
   nohup scoreReads.pl queryReads_1.fasta &
   nohup scoreReads.pl queryReads_2.fasta &
   nohup scoreReads.pl queryReads_3.fasta &
   nohup scoreReads.pl queryReads_4.fasta &
```

(The 'nohup' keyword at the beginning of each line directs the system not to kill the PhymmBL scoring process if you log out or lose your terminal connection before the process is finished; this is recommended, wherever possible, when launching long-running processes like PhymmBL. The '&' character at the end will send each process to the background, so you can regain control of your terminal before the process is finished, which is necessary if you don't want to open four separate terminals just to launch four parallel processes.)

In addition to single-machine batch processing, we have successfully deployed PhymmBL to a multi-machine computing grid, for more automated and more powerful parallel classification than can be done on a single machine (or for that matter on a single disk). Since implementation details of each grid or cloud are different – often in substantial ways – we can only provide a logical outline to this approach, but we expect that the process will be conceptually straightforward on most systems:

- Install PhymmBL on a single machine.
- Once installation is finished and all the data comprising the reference databases has been generated, copy everything in the installation directory to each of your grid nodes, preferably on a locally-accessible scratch disk to reduce data latency.
- Separate your input reads into however many components you feel are appropriate, creating a new multi-FastA file for each component.
- Schedule your batch job using whatever protocols are offered by your local grid software: each process in the batch job should
    - copy one component to the local scratch drive,
    - classify it by calling the local copy of scoreReads.pl, and
    - move the result files for the current component back to a central location for later aggregation once the batch job has finished.
    -

Supplementary Note 3: Computational resources used in the classification of the mammoth bone metagenome

The entire set of 2,278,901 input reads (avg. length 276 bp) was classified in approximately 2.5 days, on a grid consisting of 24 identical machines, with approximately 20 components (chunks of input reads) being simultaneously processed at any given time. PhymmBL was deployed to this grid as described in Supplementary Note 2. Each 12-core machine in the grid ran 64-bit Linux as its primary operating system and computing environment. 8GB of memory and one (2GHz AMD Opteron) core were allocated to each process.

Supplementary Note 4: Usage notes and algorithm details for confidence scores

When interpreting results, confidence scores should not be taken to be probabilities: as an example, it is not guaranteed that 75% of all predictions which are assigned a confidence score of 0.75 will be correct. In fact, the functions which generate PhymmBL's confidence scores were deliberately designed to be more conservative than actual probabilities in most circumstances: the true proportion of accurate predictions in such a set is expected to be greater than 75%. Confidence scores can, however, reliably serve as linearly-scaled estimators of accuracy. For example, when considering different portions of the same input set, one group of predicted labels with confidence scores of 0.8 will contain approximately twice as many correct predictions as a different group with scores of 0.4.

Scores are output from five distinct polynomial functions of two variables, namely read length and raw PhymmBL score. These polynomials (one each for the phylum, class, order, family and genus classification levels) were determined via iterative 3D curve fitting, on result data generated from synthetic classification runs, where the correct labels were known in advance so that predictive accuracy could be directly measured, plotted and fitted as a function of read length and raw score.

The polynomial function defining confidence score estimation for genus-level predictions is given below; the others have the same form, with their constants adjusted to fit the corresponding accuracies observed for each taxonomic level.

$$0.86 - \frac{\left[-rawScore - (1.25 \times readLength - 550)\right]^{\left(6 + \frac{12}{readLength}\right)}}{1.5 \times 10^{17}} + \frac{8}{readLength}$$

1. Brady, A. and S.L. Salzberg, *Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models.* Nat Methods, 2009. **6**(9): p. 673-6.
2. McHardy, A., et al., *Accurate phylogenetic classification of variable-length DNA fragments*. Nat Methods, 2007. **4**(1): p. 63-72.
3. Huson D.H., et al., *MEGAN analysis of metagenomic data.* Genome Res., 2007. **17**(3): p. 377-86.
4. Nalbantoglu, O.U., et al., *RAIphy: Phylogenetic classification of metagenomics samples using iterative refinement of relative abundance index profiles.* BMC Bioinformatics, 2011. 12:41.
5. Patil, K.R., et al., *Taxonomic metagenome sequence assignment with structured output models.* Nature Methods, 2011. **8**: p. 191-2.