Gene Prediction with Glimmer for Metagenomic Sequences Augmented by Classification and Clustering

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Published Version</td>
<td>doi:10.1093/nar/gkr1067</td>
</tr>
<tr>
<td>Citable link</td>
<td><a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:11248787">http://nrs.harvard.edu/urn-3:HUL.InstRepos:11248787</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <a href="http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA">http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA</a></td>
</tr>
</tbody>
</table>
Gene prediction with Glimmer for metagenomic sequences augmented by classification and clustering

David R. Kelley1,2,3,*, Bo Liu1, Arthur L. Delcher1, Mihai Pop1 and Steven L. Salzberg4

1Center for Bioinformatics and Computational Biology, Institute for Advanced Computer Studies, Department of Computer Science, 3115 Biomolecular Sciences Building 296, University of Maryland, College Park, MD 20742, 2Department of Stem Cell and Regenerative Biology, 7 Divinity Avenue, Harvard University, Cambridge, MA 02138, 3Broad Institute, 7 Cambridge Center, Cambridge, MA 02142 and 4McKusick-Nathans Institute of Genetic Medicine Johns Hopkins University School of Medicine Baltimore, MD, USA

Received July 29, 2011; Revised September 19, 2011; Accepted October 28, 2011

ABSTRACT

Environmental shotgun sequencing (or metagenomics) is widely used to survey the communities of microbial organisms that live in many diverse ecosystems, such as the human body. Finding the protein-coding genes within the sequences is an important step for assessing the functional capacity of a metagenome. In this work, we developed a metagenomics gene prediction system Glimmer-MG that achieves significantly greater accuracy than previous systems via novel approaches to a number of important prediction subtasks. First, we introduce the use of phylogenetic classifications of the sequences to model parameterization. We also cluster the sequences, grouping together those that likely originated from the same organism. Analogous to iterative schemes that are useful for whole genomes, we retrain our models within each cluster on the initial gene predictions before making final predictions. Finally, we model both insertion/deletion and substitution sequencing errors using a different approach than previous software, allowing Glimmer-MG to change coding frame or pass through stop codons by predicting an error. In a comparison among multiple gene finding methods, Glimmer-MG makes the most sensitive and precise predictions on simulated and real metagenomes for all read lengths and error rates tested.

INTRODUCTION

Prokaryotes inhabit a diverse array of environmental niches and account for most of the world’s biomass (1–3). They play an integral role in many ecosystems, including the human body where a typical individual carries 10–100 times more prokaryotic cells than human cells (4). The DNA sequences of these microorganisms provide us with important information about their identities, capabilities and evolution. Traditional methods for obtaining these sequences have required scientists to select a single microbe of interest, isolate it in culture and sequence its genome to high coverage (5).

Because many microbes cannot be isolated and grown in culture, researchers have increasingly turned to sequencing DNA directly from environmental samples, an approach often called ‘metagenomics’ (6,7). Metagenomics is an effective tool for exploring natural environments (e.g. acid mine drainage (8), ocean water (9), and soil (10)) and environments on and within the human body (11). With the development of improved sequencing technologies from companies such as 454 Life Sciences and Illumina, DNA sequence reads can be obtained at increasingly higher throughput and lower cost. These transformative technologies have made it possible to use metagenomics to simultaneously analyze the genomes of entire communities of microbes in an ever-broadening collection of environments.

Identifying the protein-coding genes in an organism’s genome is a fundamental step in any genome sequence analysis, and metagenomics is no different. Whether the assembly produces large contigs (8) or is highly fragmented (9), many new and interesting genes can be extracted (12). The goal of some metagenomic experiments is to compare microbial communities across environments (13,14). In these cases, accurate gene prediction is critical to perform a functional comparison (15,16).

It has long been known that sequences coding for proteins have statistical properties that differentiate them from non-coding sequences, which allows gene finding programs to identify open reading frames

*To whom correspondence should be addressed. Tel: +1 301 405 3234; Email: dkelley@umiacs.umd.edu

© The Author(s) 2011. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
Sequence composition is the most important discriminative feature, due primarily to the fact that the triplet patterns of coding DNA differ from non-coding DNA. These patterns can be captured by Markov chain models, which need to be trained on a set of ORFs from the same species or a close relative. State of the art prokaryotic gene finding softwares typically achieve >99% sensitivity and high precision on finished genomes (20).

As environmental shotgun sequencing has become more prevalent, computational gene prediction approaches have adapted to the particular challenges of these data. Since the source organisms of the sequences are unknown, the foremost challenge is training the statistical gene feature models. Following training, the gene finder must make predictions on short sequence fragments that frequently contain only part of a gene. Further complicating matters, metagenomic assemblies have many low-coverage contigs and unassembled singleton reads (21,22), in which sequencing errors are prevalent and create difficulties for gene prediction (23). In contrast, finished genomes have many fewer sequencing errors thanks to their deeper and more uniform coverage.

Despite these challenges, a number of methods for predicting genes in metagenomic sequences have been published, reporting varying degrees of success (24–28). All these previous methods incorporate sequence GC-content into their prediction algorithms to choose model parameters. GC-content is a simple way to identify training genomes that are likely to be evolutionarily related, and whose genes might have similar sequence composition. This task’s goal overlaps considerably with that of computationally assigning sequences a phylogenetic classification, which implicitly identifies close relatives. Phylogenetic classification of metagenomic sequences is a well-studied problem for which much better statistics than GC-content have been developed (29–32). In this article, we recommend using a more sophisticated classification scheme, based on the Phymm system (29), to parameterize gene prediction models for metagenomic sequences and show that it works much better than GC-content. We further enhance model parameter estimation using unsupervised sequence clustering, in which the relationships between sequences are elucidated via a partition of the sequences into clusters, generally without the use of reference (or ‘training’) genomes (33–35). In previous work, sharing information between sequences within clusters during training improved the identification of translation initiation sites (36). We demonstrate that the use of a more advanced clustering method SCIMM (34) to allow an unsupervised retraining step significantly boosts full gene accuracy as well.

In previous work, our group demonstrated that the Glimmer gene prediction software is highly effective, routinely identifying >99% of the genes in complete prokaryotic genomes (20). However, Glimmer was not designed for the highly fragmented, error-prone sequences that typify metagenomic sequencing projects today. In this article, we develop a metagenomics gene prediction system called Glimmer-MG based on the Glimmer framework. As described above, Glimmer-MG implements a metagenomics pipeline that incorporates classification and clustering of the sequences prior to gene prediction.

In addition, we addressed the other metagenomics gene prediction challenges with novel and effective solutions. Glimmer-MG incorporates a thorough probabilistic model for gene length and start/stop codon presence to aid prediction of truncated genes on short sequence fragments. Glimmer-MG predicts insertion, deletion and stop codon-introducing substitution errors in order to more closely track coding frames in raw error-prone sequences. Our implementation of these features produces the most sensitive and precise gene predictions on realistically simulated metagenomes. On a set of real 454 reads from the human gut microbiome (37), Glimmer-MG makes many more gene predictions matching known proteins than other programs.

Glimmer-MG is freely available as open-source software from www.cbcb.umd.edu/software/glimmer-mg under the Perl Artistic License (www.perl.com/pub/a/language/misc/Artistic.html).

**MATERIALS AND METHODS**

**Glimmer**

Glimmer’s salient feature is its use of interpolated Markov models (IMMs) for capturing gene composition (18). IMMs are variable-order Markov chain models that maximize the model order for each specific oligonucleotide window based on the amount of training data available. IMMs then interpolate the nucleotide distributions between the chosen order and one greater. Thus, IMMs construct the most sophisticated composition model that the training data sequences support. To segment the sequence into coding and non-coding sequence, Glimmer uses a flexible ORF-based framework that incorporates knowledge of how prokaryotic genes can overlap and upstream features of translation initiation sites (TIS) like the ribosomal binding site (RBS). Glimmer extracts every sufficiently long ORF from the sequence and scores it by the log-likelihood ratio of generating the ORF between models trained on coding versus non-coding sequence. The features included in the log-likelihood ratio are composition via the IMMs, RBS via a position weight matrix (PWM) and start codon usage. For simplicity, features are assumed to be independent so that the overall score can be computed as a sum of the individual feature log-likelihood ratios. A dynamic programming algorithm finds the set of ORFs with maximum score subject to the constraint that genes cannot overlap for more than a certain threshold, e.g. 30 bp.

**Additional features**

Glimmer is ineffective on metagenomic sequences because its gene composition model is trained under the assumption that the sequences all originated from a single genome. Recent approaches both relax this assumption and add new features used to discriminate between coding and non-coding sequence. One approach called MetaGeneAnnotator (MGA) uses a similar framework to Glimmer by scoring ORFs and choosing a high
scoring set using dynamic programming (25). MGA incorporates additional gene features, of which we add three—ORF length, adjacent gene orientation, and adjacent gene distance—to Glimmer. Below, we describe how to compute models for these features given an annotated genome. In the sections to follow, we further explain how such genomes are obtained.

First, we seek probability distributions for the length of coding and non-coding ORFs. For the coding model, our sample data are the lengths of annotated genes in the training genome. For the non-coding model, the lengths of non-coding ORFs that meet a minimum length threshold (75 bp) and a maximum overlap threshold with a gene (30 bp) are considered. One can estimate the distributions using a non-parametric method based on the histogram of lengths or a parametric method where one assumes a well-studied probability distribution and computes the maximum likelihood parameters (38). We use both methods to obtain our estimate. Where training data are plentiful, such as for common gene sizes, a non-parametric approach (such as kernel smoothing) offers greater modeling accuracy than any parameterized distribution. But when data are sparse, such as for very long ORFs, the non-parametric approach fails. For example, we cannot assign a useful probability to an ORF larger than any in our training set though it should obviously receive a large log-likelihood ratio score. A parameterized distribution can assign meaningful probabilities to ORFs of any length. We analyzed a number of distributions and found that a Gamma distribution most accurately modeled the gene length distributions examined and produced the highest accuracy gene predictions.

To combine the two versions, we use a histogram after kernel smoothing with a Gaussian kernel (38) for the first quartile (as determined by the raw counts), a Gamma distribution with maximum likelihood parameters for the last quartile and a linear combination of the two with a linearly changing coefficient in between (e.g. Figure 1). Performance was robust to other blending schemes and to the points at which the model changes. We score an ORF with the log-likelihood ratio that the feature was generated by the coding versus non-coding model and add it to the ORF’s overall score.

ORFs truncated by the end of their fragments require an adjustment to the length model. We know that the total length of a truncated ORF with \(X\) bp on a fragment is at least \(X\) and should therefore be scored higher than a complete \(X\) bp ORF. We accomplish this by modeling the joint distribution of the length and the presence of start and stop codons (Supplementary Methods).

Features computed on pairs of adjacent genes also capture useful information. For example, genes are frequently arranged nearby in the same orientation to form transcriptional units called operons (39). Alternatively, consecutive genes with opposing ‘head-to-head’ orientations (where the 5’-ends of the genes are adjacent) tend to be further apart to allow room for each gene’s respective RBS. We added two features of adjacent genes: their orientation with respect to each other and the distance between them. Again, we need distributions for coding and non-coding ORFs to score a pair of adjacent genes by their log-likelihood ratio. The gene model uses all adjacent pairs of annotated genes. For the non-coding model, we consider pairs including non-coding ORFs that satisfy the length and overlap constraints with their adjacent annotated genes.

For adjacent gene orientation, we count the number of times each adjacent arrangement appears in the training data and normalize the counts to probabilities. The adjacent gene distance model is estimated similarly.
to the gene length models described above. However, common parameterized distributions were not a good fit for the distances so we rely solely on a smoothed histogram. Because one gene’s start codon often overlaps another gene’s stop codon due to shared nucleotides, we do not smooth the histogram for distances implying overlapping start or stop codons. We incorporate these features during Glimmer’s dynamic programming algorithm for choosing ORFs by adding the log-likelihood ratios when linking an ORF to its previous adjacent ORF.

Classification

All previously published approaches to metagenomic gene prediction parameterize the gene composition models as a function of the sequence GC-content. For example, MetaGeneMark computes (offline) a logistic regression for each codon frequency as a function of GC-content for a large set of training genomes and sets its hidden Markov model parameters (online) according to the GC-content of the metagenomic sequence (28). For whole genomes, gene composition model training has traditionally been performed on annotated close evolutionary relatives rather than genomes with similar GC-content (40). Many methods for assigning a taxonomic classification to a metagenomic sequence are currently available (29–32). Here, we suggest using one of these methods called Phymm (29), rather than GC-content, to find evolutionary relatives of the metagenomic sequences on which to train. Phymm trains an IMM on every reference genome in GenBank (41), scores each input sequence with all IMM and assigns a classification at each taxonomic level according to the reference genome of the highest scoring IMM. Phymm’s IMMs are single-periodic and trained on all genomic sequence, in contrast to Glimmer’s IMMs which are three-periodic and trained only on coding sequences.

Thus, before predicting genes, we run Phymm on the input sequences to score each sequence with each reference IMM. To train the gene prediction models, we use gene annotations for the genomes corresponding to the highest scoring IMMs. These annotations are taken from NCBI’s RefSeq database (42). Though classification with Phymm is very accurate, the highest scoring IMM is rarely from the sequence’s exact source genome. For this reason, we found that training over multiple genomes (e.g. 43) captured a broader signal than improved prediction accuracy. Though most of the training can be performed offline, the models over multiple genomes must be combined online for each sequence. Features such as the length, start codon and adjacent gene distributions are easy to combine across multiple training genomes by simply summing the feature counts.

IMMs cannot be combined quickly, and saving trained IMMs for all combinations of two or three genomes would require too much disk space. In practice, pairs of genomes with similar composition are far more likely to be top classification hits together and we can restrict our offline training to only these pairs (Supplementary Methods).

Glimmer-MG’s RBS model trains using ELPH (http://cbbc.umd.edu/software/ELPH), a motif finder based on Gibbs sampling, to learn a 6-bp PWM from the 25-bp upstream of every gene in the training set. We train these PWMs offline for each individual reference genome, but like the other features, RBS modeling for metagenomic sequences benefits from the broader signal obtained by combining over multiple training genomes. Averaging PWMs for the top three Phymm classifications can be done quickly, but dilutes the signal. Instead, we generalized the RBS model in Glimmer-MG to score the upstream region of each start codon using a mixture of PWMs in equal proportions. Thus, a gene’s RBS score is the probability that the best 6 bp motif in the 25-bp upstream of the start codon was generated by a mixture of three PWMs normalized by a null model based on GC-content to a log-likelihood ratio.

Two interesting cases warrant further discussion. First, a novel sequence may not be phylogenetically related to any known reference genome in the database. Here, Phymm’s highest scoring IMMs will merely represent the reference genomes with most similar nucleotide composition. Prior work demonstrating the relationship between even simple nucleotide composition statistics and prediction model parameters supports the validity of this strategy (24–28). In addition, we did not detect a significant relationship between prediction accuracy and the divergence of a sequence from the reference genome database (Supplementary Figure S2). Second, some sequences will contain horizontally transferred genes. While single genome gene prediction typically cannot implement a model general enough to predict these genes accurately, Glimmer-MG is more robust because Phymm will likely ‘mis-classify’ the sequence containing the gene by scoring the sequence more highly with IMMs more representative of the genome from which the gene was transferred than the sequence’s true source genome.

Clustering

The following prediction pipeline has been applied successfully on whole prokaryotic genomes. First, train models on a finished and annotated close evolutionary relative. Make initial predictions, but then retrain the models on them and make a final set of predictions (40). By using Phymm to find training genomes, we replicate the first step in this pipeline for application to metagenomics. However, retraining on the entire set of sequences would combine genes from many different organisms and yield a non-specific and ineffective model. If the sequences could be separated by their source genome, retraining could be applied.

We accomplish this goal using SCiMM, an unsupervised clustering method for metagenomic sequences that models each cluster with a single-periodic IMM (34). After initially partitioning the sequences into a specified number of clusters, SCiMM repeats the following three steps until the clusters are stable: train IMMs on the sequences assigned to their corresponding clusters, score each sequence using each cluster IMM and reassign each sequence to the cluster corresponding to its highest scoring IMM. While
SCIMM may not partition the sequences exactly by their source organism, the mistakes that it tends to make do not create significant problems for retraining gene prediction models. In cases where SCIMM merges sequences from two organisms together, they are nearly always phylogenetically related at the family level (34). Though some families can be quite diverse, this shared phylogenetic relationship, combined with the nucleotide composition similarity that SCIMM more directly identifies, is encouraging. SCIMM sometimes separates sequences from a single organism into multiple clusters, but this occurs most often for highly abundant organisms, in which case there will usually still be enough training data in each cluster to be informative. The Phymm classifications that have already been obtained imply an initial clustering at a specified taxonomic level (e.g. family), which can be used as an initial partition for the iterative clustering optimization in a mode of the program referred to as PHYSCIMM (34). Using PHYSCIMM also implicitly chooses the number of clusters, removing this free variable.

After clustering the sequences, we focus on each cluster individually to retrain the coding IMM, RBS and start codon models before making the final predictions within that cluster. The ORF length and adjacent ORF feature distributions are more difficult to estimate from short sequence fragments, so we continue to learn them using the Phymm classifications to whole annotated genomes. If the cluster is too small, retraining may not have enough data to capture the gene features, and prediction accuracy may decrease. We tested various thresholds and requiring at least 80 Kb of predicted coding sequence for retraining produced the highest accuracy predictions. For clusters with less, we do not retrain and instead finalize the gene predictions from the initial iteration. Accuracy may also decrease if the cluster is heterogeneous and does not effectively model some of its sequences. For each sequence, we compute the ratio between the likelihood of the cluster IMM versus its top scoring Phymm IMM generated the sequence. If the ratio is too low, we assume that the cluster does not represent this sequence well enough and finalize its initial predictions. The full pipeline for metagenomic gene prediction is depicted in Figure 2.

### Sequencing errors

Gene prediction on raw sequencing reads or contigs with low coverage must contend with sequencing errors. The most prevalent type of error made by the 454 sequencing technology is an insertion or deletion (indel) at a homopolymer run. Indels cause major problems for gene prediction by shifting the coding frame of the true gene, making it impossible for a method without a model for these errors to predict it exactly. When Glimmer-MG encounters a shifted gene, the most frequent outcome is two predictions, each of which covers half of the gene up to the point of the indel and then beyond (Figure 3). Such predictions have limited utility.

While the problems caused by sequencing errors have been known for some time (23,22), only recently has a good solution been published in the program FragGeneScan (26). FragGeneScan uses a hidden Markov model where each of the three indexes into a codon are represented by a model state, but allows irregular transitions between the codon states that imply the presence of an indel in the sequence. On simulated sequences containing errors, FragGeneScan achieves far greater accuracy than previous methods that ignore the possibility of errors.

Since Glimmer-MG uses an ORF-based approach to gene prediction, we must take a more ad hoc approach to building an error model into the algorithm. First, we...
address 454 indel errors. When Glimmer-MG is scoring
the composition of an ORF using the coding and
non-coding IMMs, we allow branching into alternative
reading frames. More specifically, we traverse the
sequence and identify low-quality base calls (defined
below) that are strong candidates for a sequencing error.
At these positions, we split the ORF into three branches.

One branch scores the ORF as is. The other two switch
into different frames to finish scoring, implying an inser-
tion and deletion prediction. ORFs that change frames are
penalized by the log-likelihood ratio of the predicted cor-
rection’s probability to the original base call probability.
A maximum of two indel predictions per ORF is used to
limit the computation time. After scoring all ORFs, ORFs
with the same start and stop codon (but potentially dif-
ferent combinations of interior indels) are clustered and
only the highest scoring version is kept. All remaining
ORFs are pushed to the dynamic programming stage
where the set of genes with maximum score subject to
overlap constraints is chosen. However, the algorithm is
further constrained to disallow an indel prediction in a
region of overlapping genes.

Focusing on low-quality base calls (typically <5–10% of
the sequence) makes the computation feasible. If quality
values are available for the sequences, either from the raw read output or the consensus stage of an
assembler, Glimmer-MG uses them and designates base
calls less than a quality value threshold as potential
branch sites. For 454 sequences that are missing quality
values, we designate the final base of homopolymer runs
longer than a length threshold as potential branch sites.

In Illumina reads, indels are rare, and the primary
errors are substitutions (44). Most substitutions do not
affect a start or stop codon and are nearly irrelevant to
the gene prediction. We focus on the most detrimental
error, which is a substitution that converts a regular
codon to a stop codon, thus prematurely truncating the
gene. To predict such errors, we consider substitution
errors to remove each stop codon in the sequence. That
is, for every ORF, we consider an altered ORF where the
previous stop codon did not exist, thus combining the
current ORF with the previous one in the same frame.
Similarly to the 454 error model, we penalize these
altered ORFs with the log-likelihood ratio (based on the
quality values) comparing the probability that the stop
codon contains a sequencing error that changed it from
a regular codon to the probability that it truly is a stop
codon. All normal and altered ORFs are considered
during the dynamic programming stage to choose the
maximum scoring set of ORFs.

Whole genomes

Although we implemented the additional gene features
with metagenomics in mind, they improve accuracy on
whole genomes as well. In Glimmer3.0, the following
pipeline was recommended (20). First, using a program
called long-orfs, find long non-overlapping ORFs in the
sequence with amino acid composition that is typical of
prokaryotic genomes. Train the coding IMM on these se-
quences, and predict genes on the genome. On the initial
predictions, train the RBS and start codon models.
Finally, make a second set of gene predictions
incorporating the new models.

We recommend a similar scheme for a new
whole-genome pipeline, designated as Glimmer3.1. As
before, we use long-orfs to train an IMM and predict an
initial set of genes. Without a length model, these initial
predictions tend to include many erroneous small gene
predictions. We use a log-likelihood ratio threshold to
filter out the lowest scoring ones. On the remaining
genes, we retrain all models—IMM, RBS, start codons,
length and adjacency features—before predicting again.
To eliminate any remaining bias from the initial prediction
and filtering, we retrain and predict one final time.

The preceding pipeline is unsupervised, but we can do
slightly better on average by following Glimmer-MG and
using GenBank reference genomes. In this pipeline,
we first classify our new genome with Phymm to find
similar reference genomes. Alternatively, a researcher
may be able to specify these genomes based on prior
knowledge. We train RBS, start codon, length and adja-
cency models from the RefSeq annotations of these similar
genomes as described above. For the gene IMM, accuracy
is better if we use long-orfs compared with an IMM
trained on relative reference genomes. After making
initial predictions, we retrain the IMM, RBS and start
codon models before predicting genes a final time.

Simulated metagenomes

We constructed simulated datasets from 1206 prokaryote
genomes in GenBank (41) as of November 2010. Since
Glimmer-MG involves clustering the sequences, it is im-
portant to have realistic simulated metagenomes. For each
metagenome, we randomly chose 50 organisms and
included all chromosomes and plasmids. We sampled
organism abundances from the Pareto distribution, a
power law probability distribution that has previously
been used for modeling metagenomes (45). Reference
genomes included in the metagenome were removed
from Phymm’s database so that the sequences appeared
novel and unknown. To simulate a single read, we selected
a chromosome or plasmid with probability proportional
to the product of its length and the organism’s abundance
and then chose a random position and orientation from
that sequence. To enable comparison between experiments
with different read lengths and error rates, we simulated
20 metagenomes (i.e. organisms, abundances, read posi-
tions and read orientations) and used them to derive
each experiment’s dataset. We labeled the reads using
gene annotations that are not described as hypothetical
proteins from RefSeq (42).

In experiments where we considered sequencing errors,
we focused on three prevailing technologies. Two varieties
of high-throughput, short-read Technologies with very dif-
ferent characteristics have become ubiquitous tools for
sequencing genomes, including metagenomics (46). The
Illumina sequencing platform generates 35–150 bp length
reads with sequencing errors consisting almost entirely of
substitutions (44). The 454 sequencing platform generates
400–550 bp length reads where indels make up nearly all
of the errors (47). Less popular in recent studies due to greater expense and lesser throughput is Sanger sequencing with read lengths of 600–1000 bp and both substitution and indel sequencing errors. We include Sanger sequencing both because previous programs were designed and tested with the technology in mind and because the reads resemble contigs assembled from the more prevalent short read technologies with respect to length and errors tending to occur at the fragments’ ends.

To imitate Sanger reads, we used the lengths and quality values from real reads taken from the NCBI Trace Archive (41) as templates. That is, for each fragment simulated from a genome as described above, we randomly chose a real Sanger read from our set to determine the length and quality values of the simulated read. Then we simulated errors into the read according to the quality values and using a ratio of five substitutions per indel. To achieve a specific error rate for a dataset, we multiplied the probability of error at every base by a factor defined by the desired rate. To generate simulated reads to imitate the Illumina platform, we similarly used real 124 bp reads as templates to obtain quality values, but injected only substitution errors. For 454 reads, we used a read simulator called FlowSim, which closely replicates the 454 stochastic sequencing process to generate the sequences and their quality values (43). We conservatively quality trimmed all read ends to avoid large segments of erroneous sequence.

Accuracy
We computed accuracy a few ways to capture the multiple goals of gene prediction. Sensitivity is the ratio of true positive predictions to the number of true genes, and precision is the ratio of true positive predictions to the number of predicted genes. Since the RefSeq annotations tend to be incomplete after the removal of hypothetical proteins, which are unconfirmed computational predictions, we consider sensitivity to be the more important measure as ‘false positive’ predictions may actually be real genes. For this reason, precision values in the experiments we performed are artificially low and should be interpreted carefully. For all experiments, we computed the sensitivity and precision of the 5’- and 3’-ends of the genes separately. Since there is only a single 3’ site, 3’ prediction is generally given more attention. There are frequently many choices for the 5’-end of the gene and a paucity of sequence information to discriminate between them. Adding to the difficulty, most of the 5’ annotations in even the high-quality RefSeq database are unverified.

In experiments with sequencing errors, indels shift the gene’s frame and substitutions can compromise the start and stop codons. To measure the ability of the gene prediction to follow the coding frame, we compute sensitivity and precision at the nucleotide level. That is, every nucleotide is considered a unit and a true positive prediction must annotate the nucleotide as coding in the correct frame. A gene prediction that is correct until a sequencing error indel but predicted in the wrong frame beyond gets partial credit, whereas a gene prediction that identifies the error location and shifts the frame of the prediction gets full credit.

RESULTS
Whole genomes
First, we evaluated the accuracy of the previous Glimmer3.0 iterated pipeline versus the new version with additional features, Glimmer3.1, and the metagenomics pipeline, Glimmer-MG. We predicted genes in 12 reference genomes that cover a wide range of the prokaryotic phylogeny and were previously used to compare Glimmer3.0 with Glimmer2 (20). Results for each of these genomes are displayed in Supplementary Table S1.

Glimmer3.1 maintains the high 3’ sensitivity of Glimmer3.0, but improves the precision by 1.3% on average mainly by predicting fewer short genes (42 predictions <150 bp per genome versus 68) due to the length model. Glimmer-MG increases precision another 1.0% by using additional models in the initial iteration, such as for gene length, learned accurately from close evolutionary relatives. Glimmer3.1 also significantly improves TIS prediction as 5’ sensitivity increases by 1.3% and precision by 1.8%. This improvement is attributable to its ability to assign greater scores to upstream start codons (which are longer genes) and penalize adjacent genes for unlikely arrangements like long overlaps. Glimmer-MG boosts sensitivity and precision relative to Glimmer3.1 by another 0.5 and 1.2%, respectively.

Simulated metagenomes — perfect reads
To compare Glimmer-MG to other methods for metagenomics gene prediction, we first predicted genes on simulated metagenomes with perfect read data without sequencing errors using Glimmer-MG, MetaGeneAnnotator (25), MetaGeneMark (28), and FragGeneScan (26). MetaGeneAnnotator and MetaGeneMark runs used default parameters, and we set FragGeneScan’s parameters for error-free sequences. Table 1 displays the programs’ averaged accuracies over the 20 simulated metagenomes for each read technology. To best explore the programs’ performances, we would ideally plot a series of sensitivity and precision data points similar to a receiver operating characteristic (ROC) curve. However, the other programs lack an accessible parameter to trade off sensitivity versus precision. Since this can easily be done for Glimmer-MG by adding a constant to every ORF’s score, we display accuracy results for two versions of Glimmer-MG. The first, labeled Glimmer-MG, is our recommended mode where a constant 1.0 is added to every ORF’s score. The second, labeled Glimmer-MG (Prec), reports accuracy at the point where Glimmer-MG’s precision passes that of all other programs. Glimmer-MG (Prec) serves to compare Glimmer-MG to the other programs for users whose particular application requires highly precise predictions at some expense to sensitivity.

Overall Glimmer-MG emerged as the clear best method, achieving the greatest sensitivity for every read length. Glimmer-MG’s 3’ sensitivity was 1.3–1.8% greater than
Table 1. Accuracy on simulated metagenomes with perfect reads

<table>
<thead>
<tr>
<th>Technology</th>
<th>Method</th>
<th>3' Sensitivity</th>
<th>3' Precision</th>
<th>5' Sensitivity</th>
<th>5' Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sanger</td>
<td>Glimmer-MG</td>
<td>0.986</td>
<td>0.709</td>
<td>0.900</td>
<td>0.648</td>
</tr>
<tr>
<td>(870 bp)</td>
<td>Glimmer-MG (Prec)</td>
<td>0.986</td>
<td>0.709</td>
<td>0.900</td>
<td>0.648</td>
</tr>
<tr>
<td></td>
<td>MetaGeneMark</td>
<td>0.969</td>
<td>0.707</td>
<td>0.857</td>
<td>0.625</td>
</tr>
<tr>
<td></td>
<td>MetaGeneAnnotator</td>
<td>0.969</td>
<td>0.702</td>
<td>0.846</td>
<td>0.613</td>
</tr>
<tr>
<td></td>
<td>FragGeneScan</td>
<td>0.962</td>
<td>0.667</td>
<td>0.823</td>
<td>0.570</td>
</tr>
<tr>
<td>454</td>
<td>Glimmer-MG</td>
<td>0.984</td>
<td>0.718</td>
<td>0.917</td>
<td>0.669</td>
</tr>
<tr>
<td>(535 bp)</td>
<td>Glimmer-MG (Prec)</td>
<td>0.982</td>
<td>0.721</td>
<td>0.916</td>
<td>0.673</td>
</tr>
<tr>
<td></td>
<td>MetaGeneMark</td>
<td>0.964</td>
<td>0.718</td>
<td>0.877</td>
<td>0.653</td>
</tr>
<tr>
<td></td>
<td>MetaGeneAnnotator</td>
<td>0.966</td>
<td>0.707</td>
<td>0.853</td>
<td>0.625</td>
</tr>
<tr>
<td></td>
<td>FragGeneScan</td>
<td>0.959</td>
<td>0.680</td>
<td>0.859</td>
<td>0.609</td>
</tr>
<tr>
<td>Illumina</td>
<td>Glimmer-MG</td>
<td>0.945</td>
<td>0.695</td>
<td>0.919</td>
<td>0.676</td>
</tr>
<tr>
<td>(120 bp)</td>
<td>Glimmer-MG (Prec)</td>
<td>0.925</td>
<td>0.718</td>
<td>0.901</td>
<td>0.700</td>
</tr>
<tr>
<td></td>
<td>MetaGeneMark</td>
<td>0.901</td>
<td>0.717</td>
<td>0.871</td>
<td>0.693</td>
</tr>
<tr>
<td></td>
<td>MetaGeneAnnotator</td>
<td>0.915</td>
<td>0.686</td>
<td>0.839</td>
<td>0.629</td>
</tr>
<tr>
<td></td>
<td>FragGeneScan</td>
<td>0.932</td>
<td>0.663</td>
<td>0.904</td>
<td>0.643</td>
</tr>
</tbody>
</table>

Technology refers to the sequencing technology emulated. 3’ Sensitivity/Precision refer to sensitivity and precision based on predicted genes matching at their 3’-ends. Sensitivity is computed as the percentage of true genes that are correctly predicted, and precision is the percentage of predicted genes that are correct. 5’ Sensitivity/Precision is computed similarly using matches at the 5’-end (the start codon) of each gene. Accuracies are bolded when they are significantly greater than all other methods in that experiment using a one-sided sign test and 0.05 P-value cutoff (ignoring Glimmer-MG versus Glimmer-MG (Prec)).

...improvement expected based on prior work (36). Given that retraining increases accuracy, predictions on the few sequences placed in small or heterogeneous clusters will be slightly less accurate on average than predictions that benefited from retraining.

Simulated metagenomes - error reads

Real metagenomic sequences will inevitably contain sequencing errors, and prior work showed that current gene prediction software struggles with these errors (23). The recently published method FragGeneScan specifically models indel sequencing errors and achieves much greater accuracy than other approaches when the sequences are short and error-prone (26). To compare Glimmer-MG to FragGeneScan on reads containing errors, we simulated metagenomes as described using error rates ranging from 0% to 2%. We allowed Glimmer-MG to predict indels for Sanger and 454 reads and substitutions in stop codons for Illumina. We ran FragGeneScan using predefined model parameters meant for the sequencing technology and closest error rate. We included MetaGeneMark with default parameters due to its competitiveness on perfect reads, but excluded MetaGeneAnnotator for clarity because we found its performance lacking, similarly to previous evaluations (26). Table 2 displays the programs’ averaged accuracies at the nucleotide level over the 20 simulated metagenomes for each read technology and error rate.

Glimmer-MG outperformed FragGeneScan with respect to both sensitivity and precision on all read lengths and error rates. The improvement was particularly evident for 454 reads where, for example, Glimmer-MG achieved 5.8% greater sensitivity and 6.0% greater precision than FragGeneScan at a 1.0% error rate. Glimmer-MG’s limit of 2 indels per gene did not hinder gene prediction at a higher rate of 2.0% as accuracy remained greater than FragGeneScan. MetaGeneMark...
We can explain this as follows. Without sequencing errors, Glimmer-MG’s predictions are very accurate so that the potential benefit of retraining and predicting again is limited. However, when there are sequencing errors, predicting coding sequence around indels is far more difficult, and the enhanced ability of Glimmer-MG’s retrained models to identify coding sequence affects accuracy more significantly.

We measured Glimmer-MG and FragGeneScan’s accuracy predicting indels in the 454 simulated reads to determine the degree to which it affected gene prediction accuracy. To do so, we computed a matching between the predicted and true indels in coding regions and called a pair separated by <15 bp a true positive. At a 1.0% error rate, Glimmer-MG correctly predicted 23.2% of the indels, with 63.8% precision. FragGeneScan more readily shifts the gene frame and made 1.9 times more indel predictions. However, they resulted in fewer true positives than Glimmer-MG (19.2% sensitivity) and far lower precision at 28.4%. For indels predicted correctly by both programs, Glimmer-MG’s prediction was 2.3 bp away from the actual position on average compared with 5.2 bp away for FragGeneScan. Thus, by focusing on low-quality nucleotides in the sequences, Glimmer-MG identifies indel positions more effectively than FragGeneScan. Sensitivity for both methods may seem low, but note that, in some cases, the frame of the coding sequence can still be closely followed without predicting the correct error. For example, two nearby insertions will generally result in a deletion prediction, which restores the proper frame more parsimoniously than two insertion predictions.

**Human gut microbiome**

We evaluated Glimmer-MG’s performance on two real metagenomic datasets from a human gut microbiome

Table 2. Accuracy on simulated metagenomes with error reads

<table>
<thead>
<tr>
<th>Method</th>
<th>Sanger</th>
<th>454</th>
<th>Illumina</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sensitivity</td>
<td>Precision</td>
<td>Sensitivity</td>
</tr>
<tr>
<td>Glimmer-MG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.989</td>
<td>0.756</td>
<td>0.988</td>
</tr>
<tr>
<td>0.005</td>
<td>0.972</td>
<td>0.741</td>
<td>0.907</td>
</tr>
<tr>
<td>0.010</td>
<td>0.957</td>
<td>0.729</td>
<td>0.836</td>
</tr>
<tr>
<td>0.020</td>
<td>0.928</td>
<td>0.709</td>
<td>0.732</td>
</tr>
<tr>
<td>Glimmer-MG (Prec)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.989</td>
<td>0.756</td>
<td>0.988</td>
</tr>
<tr>
<td>0.005</td>
<td>0.972</td>
<td>0.744</td>
<td>0.907</td>
</tr>
<tr>
<td>0.010</td>
<td>0.954</td>
<td>0.739</td>
<td>0.836</td>
</tr>
<tr>
<td>0.020</td>
<td>0.922</td>
<td>0.724</td>
<td>0.732</td>
</tr>
<tr>
<td>FragGeneScan</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.977</td>
<td>0.740</td>
<td>0.975</td>
</tr>
<tr>
<td>0.005</td>
<td>0.953</td>
<td>0.699</td>
<td>0.846</td>
</tr>
<tr>
<td>0.010</td>
<td>0.938</td>
<td>0.687</td>
<td>0.778</td>
</tr>
<tr>
<td>0.020</td>
<td>0.914</td>
<td>0.674</td>
<td>0.678</td>
</tr>
<tr>
<td>MetaGeneMark</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.983</td>
<td>0.755</td>
<td>0.979</td>
</tr>
<tr>
<td>0.005</td>
<td>0.954</td>
<td>0.743</td>
<td>0.605</td>
</tr>
<tr>
<td>0.010</td>
<td>0.938</td>
<td>0.739</td>
<td>0.463</td>
</tr>
<tr>
<td>0.020</td>
<td>0.919</td>
<td>0.723</td>
<td>0.322</td>
</tr>
</tbody>
</table>

Error refers to the average rate at which errors were simulated into the sequences. Sensitivity/Precision refer to sensitivity and precision based on predicted genes matching at the nucleotide level. Sensitivity is computed as the percentage of true gene nucleotides that are correctly predicted, and precision is the percentage of predicted gene nucleotides that are correct. Accuracies are bolded when they are significantly greater than all other methods in that experiment using a one-sided sign test and 0.05 P-value cutoff [ignoring Glimmer-MG versus Glimmer-MG (Prec)].
study of obese and lean twins (37). Sample TS28 consists of 303K reads sequenced by the 454 GS FLX Titanium with average length of 335 bp, and sample TS50 consists of 550K reads sequenced by the 454 GS FLX with average length of 204 bp. Evaluating prediction accuracy is more difficult for real metagenomes where there is no gold standard to compare against. We aligned the translated gene predictions made by Glimmer-MG, FragGeneScan and MetaGeneMark against the NCBI non-redundant protein database with BLAST (41,48), and considered a prediction to be a true positive if it matched a database protein with BLAST alignment with E-value <0.001 was found to a non-redundant protein database. Following the count in parentheses is the proportion of predictions with such an alignment.

In this article, we introduced a number of novel and effective techniques for metagenomics gene prediction in the software package Glimmer-MG. By modeling gene lengths and the presence of start and stop codons, Glimmer-MG successfully accounts for the truncated genes so common on metagenomic sequences. Where previous approaches parameterize prediction models using only the GC-content of the sequence, Glimmer-MG first classifies the sequences using a leading phylogenetic classifier Phymm and trains models using the results. By clustering the reads using the unsupervised method SCIMM, we elegantly allow retraining of prediction models on the sequences themselves. Augmenting gene prediction with classification and clustering produces the most accurate predictions in our experiments. Furthermore, Glimmer-MG produces highly accurate predictions on whole genomes, by automatically identifying related training genomes, which boosts the accuracy of initial predictions used for retraining.

Insertion and deletion sequencing errors in real metagenomics data wreak havoc on gene sequences. In Glimmer-MG, we take a novel approach to predict indels in error-prone sequences by focusing the search for frameshifts at low-quality positions. This results in more accurate identification of simulated indel positions than a previous method FragGeneScan. Glimmer-MG is also the first gene prediction method to predict substitution errors affecting stop codons, which improves accuracy for Illumina reads. In our experiments with real gut microbiome reads and simulated metagenomes with multiple sequencing technologies, Glimmer-MG predicts genes on error-prone sequences far more accurately than all other methods.

Despite Glimmer-MG’s contributions, predicting genes on error-prone sequences, particularly those with indels, remains a difficult problem. On simulated 454 reads with a 2.0% error rate, Glimmer-MG’s sensitivity is 73.2% and precision is 54.7%, indicating that many genes are predicted incorrectly. While further computational improvements may appear, substantial gains are more likely to come from reducing the error rate via improvements to assembly of the metagenomic reads and the base-calling accuracy of the sequencing technologies. Highly diverged organisms where the optimal model parameters are unclear present another challenge to metagenomics gene prediction. However, our results here and previous work indicate that nucleotide composition models generalize well to these sequences. Finally, Glimmer-MG has the
caveat that it is more computationally expensive than existing software. Classification and clustering of the sequences are required before prediction, but we expect most researchers will not find this disruptive given that these are common components of metagenomic analysis pipelines. The gene prediction step in Glimmer-MG is also more expensive than previous software due to the use of a more sophisticated probabilistic model, modeling of sequencing errors and multiple iterations. Users will need to evaluate the trade-off between greater accuracy and computational expense for their particular data. But because gene prediction with any of the programs can be easily parallelized and significant computational resources are increasing accessible (e.g. via cloud computing (49,50)), we expect many users will find the additional computation worthwhile.

Overall, Glimmer-MG represents a substantial advance in metagenomics gene prediction on a number of fronts. The software should prove useful for a variety of applications.

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online: Supplementary Table 1, Supplementary Figures 1 and 2, and Supplementary Methods.

ACKNOWLEDGEMENTS

We thank Derrick Wood for helpful TIS discussion and manuscript comments.

FUNDING

National Institute of Health grants (R01-LM083873 and R01-HG006677). Funding for open access charge: National Institute of Health grants (R01-LM083873 and R01-HG006677).

Conflict of interest statement. None declared.

REFERENCES


