BIOINFORMATICS ORIGINAL PAPER

Vol. 30 no. 9 2014, pages 1214–1219
doi:10.1093/bioinformatics/btu010

Sequence analysis

Advance Access publication January 9, 2014

BlindCall: ultra-fast base-calling of high-throughput sequencing data by blind deconvolution

Chengxi Ye[1,2], Chiaowen Hsiao[2,3] and Héctor Corrada Bravo[1,2,3,*]

[1]Department of Computer Science, [2]Center for Bioinformatics and Computational Biology and [3]Applied Mathematics and Scientific Computing, University of Maryland, College Park, USA

Associate Editor: Dr John Hancock

ABSTRACT

Motivation: Base-calling of sequencing data produced by high-throughput sequencing platforms is a fundamental process in current bioinformatics analysis. However, existing third-party probabilistic or machine-learning methods that significantly improve the accuracy of base-calls on these platforms are impractical for production use due to their computational inefficiency.

Results: We directly formulate base-calling as a blind deconvolution problem and implemented BlindCall as an efficient solver to this inverse problem. BlindCall produced base-calls at accuracy comparable to state-of-the-art probabilistic methods while processing data at rates 10 times faster in most cases. The computational complexity of BlindCall scales linearly with read length making it better suited for new long-read sequencing technologies.

Availability and Implementation: BlindCall is implemented as a set of Matlab scripts available for download at http://cbcb.umd.edu/~hcorrada/secgen.

Contact: hcorrada@umiacs.umd.edu

Received on August 1, 2013; revised on December 12, 2013; accepted on January 6, 2014

1 INTRODUCTION

Second-generation sequencing technology has revolutionized high-throughput genomics in life science and clinical research. The sheer scale of sequence generated by these instruments has allowed unprecedented views into a number of molecular phenomena, including population genetics, transcriptomics, epigenetics and translational profiling. Both the throughput and accuracy of second-generation sequencing instruments has increased at an accelerated pace in the last few years due to the use of high-resolution optics and biochemical methods that allow sequencing of billions of DNA fragments in parallel by generating fluorescence intensity signals that can be decoded into DNA sequences. However due to experimental and hardware limitations, these raw signals are inherently noisy (Aird *et al.*, 2011; Bravo and Irizarry, 2010; Dohm *et al.*, 2008; Erlich *et al.*, 2008). Base-calling is the essential step of converting these noisy fluorescent intensity signals into sequences used in downstream analysis. Providing accurate base-calls greatly reduces many difficulties in downstream bioinformatics analysis like genome assembly and variant calling (Alkan *et al.*, 2011; Bravo and Irizarry, 2010).

*To whom correspondence should be addressed.

Sequencing-by-synthesis (Bentley *et al.*, 2008) generates millions of reads of short DNA sequences by measuring in parallel the fluorescence intensity of billions of PCR-amplified and labeled clusters of DNA from a sample of interest. The DNA fragments attach to a glass surface where it is then PCR-amplified *in situ* to create a cluster of DNA fragments with identical nucleotide composition. Sequence reads are generated from these DNA clusters in parallel and by cycles. A single nucleotide is sequenced from all DNA clusters in parallel by adding labeled nucleotides that incorporate to their complementary nucleotide. This synthesizes DNA fragments complementary to the fragments in each cluster as sequencing progresses. A set of four images is created measuring the fluorescence intensity along four channels to detect incorporation at each cycle. These images are then processed to produce fluorescence-intensity measurements from which sequences are then inferred by base-calling. In the default base-calling process for Illumina sequencers, called Bustard, the highest intensity in each quadruplet of intensity measurements determines the base at the corresponding position of the corresponding read. For current Illumina technologies, sequencers can produce up to 600 GB per run (Illumina, 2013).

The raw intensity signals generated by this process are known to be subject to several biases (Aird *et al.*, 2011; Bravo and Irizarry, 2010; Dohm *et al.*, 2008; Erlich *et al.*, 2008) (Fig. 1A and B). (i) Cross talk: there are significant correlations between different nucleotide channels; (ii) phasing/pre-phasing: the signal in one cycle can spread to the cycles ahead and the cycles after it; (iii) signal decay: where signal intensities become lower in later sequencing cycles; (iv) background noise: the signal to noise ratio becomes lower in later sequencing cycles. A significant challenge in base-calling is accounting for these biases.

Existing base-calling methods can be classified into two major groups: (i) unsupervised model-based methods that capture the sequencing-by-synthesis process in a statistical model of fluorescence intensity from which base-call probabilities can be extracted directly (Bravo and Irizarry, 2010; Kao and Song, 2011; Kao *et al.*, 2009; Massingham and Goldman, 2012) and (ii) supervised methods that train a statistical model on a set of base-calls whereby fluorescence intensity measurements are classified into base-calls (Erlich *et al.*, 2008; Kircher *et al.*, 2009). The former methods have been shown to significantly improve the accuracy of Bustard base-calls. These model-based methods aim to capture the sequencing process described above in a statistical model from which base-call probabilities are usually obtained. While these probabilistic or machine-learning methods improve

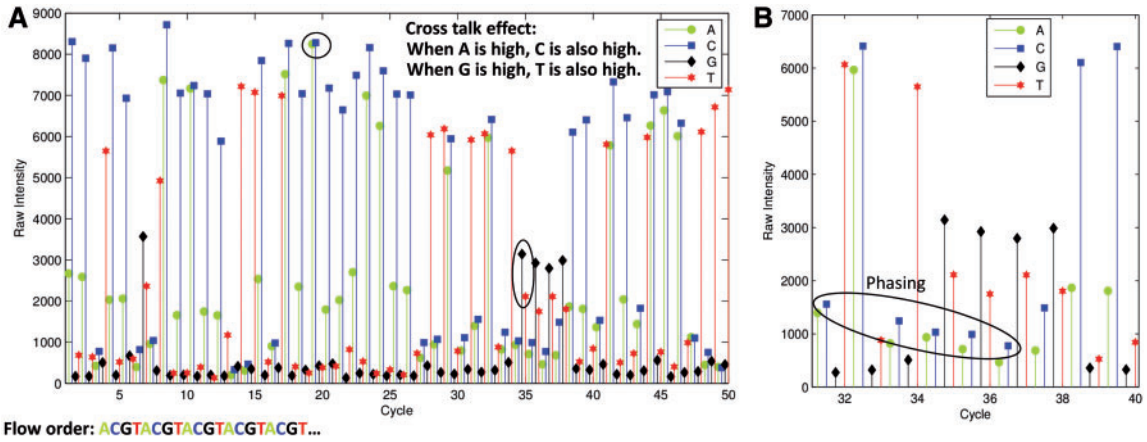

Fig. 1. Signal properties in the base-calling problem. (A) Fluorescence intensity measurements from one cluster for 50 sequencing cycles. Cross-talk and signal decay effects are clearly observed in this data. Background intensity increases as sequencing progresses. (B) The phasing effect demonstrated on a subset of data from (A). High intensity in the C channel in cycle 32 affects background intensity in the C channel in neighboring cycles

the accuracy of base-calls, they are impractical for use due to their computational inefficiency, which usually scales quadratically with read length since most of them resort to dynamic programming for model fitting (Kao and Song, 2011; Kao *et al.*, 2009; Massingham and Goldman, 2012).

In this article, we show that the base-calling problem can be formulated as an optimization problem called blind deconvolution. Based on this observation, we developed BlindCall as a method that treats base-calling as a blind deconvolution problem (Levin *et al.*, 2011; Xu *et al.*, 2013). We model intensity signals (B) output by the sequencer as the convolution of a latent sparse signal of interest X and a convolution kernel k modeling cross-talk and phasing biases, plus background noise N:

$$B = k * X + N.$$

The blind deconvolution problem is to recover the latent signal X given only the observed B. This reduces the base-calling problem into solving an inverse problem that admits computationally efficient solutions. The blind deconvolution problem has been a research hotspot in recent years (Levin *et al.*, 2011; Xu *et al.*, 2013) and we adapt methods for its solution to the base-calling problem (Wang and Yin, 2010).

BlindCall was able to provide base-calls at comparable accuracy to state-of-the-art probabilistic methods while processing data at rates ten times or faster in most cases. It scales linearly with read length and is thus better suited for new long-read sequencing technologies. Direct blind deconvolution modeling and the ultra-efficient processing based on optimization methods presented here are essential for bioinformatics analysis workflows to cope with increased throughput and read lengths in new sequencing technologies.

2 METHODS

BlindCall follows the following architecture (Fig. 2A): a training module uses blind deconvolution (Fig. 2B) on a randomly sampled subset (e.g. 1000 reads) of the intensity data to iteratively estimate the convolution kernel k and produce a deconvolved signal from which base-calling is performed. The base-calling module then uses the convolution kernel

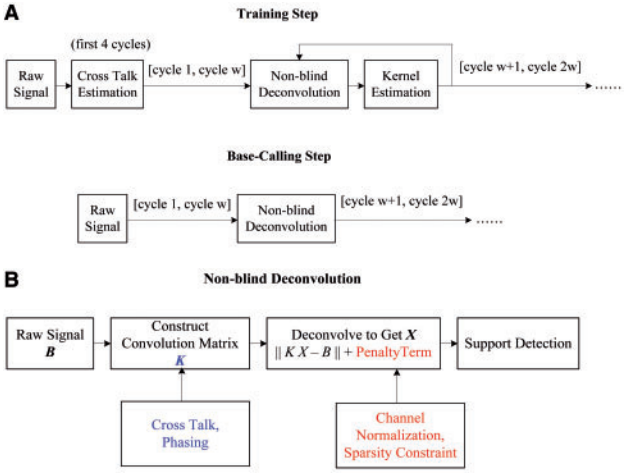

Fig. 2. The BlindCall architecture. BlindCall consists of two modules: (A) the training module uses blind deconvolution and (B) to simultaneously estimate model parameters and produce a deconvolved signal from which base-calling is done. The calling module uses the parameters estimated in the training module to produce a deconvolved output signal

estimated in the training module to produce a deconvolved output signal for the entire dataset and call bases.

2.1 Blind deconvolution

We solve the blind deconvolution problem using an iterative procedure: (i) fixing k and estimating latent signal X using a specific non-blind deconvolution method based on iterative support detection (ISD) (described below) and then (ii) fixing X to estimate convolution kernel k to correct for cross-talk and phasing effects. We divide the signal into non-overlapping windows: in each 20-cycle window we assume an invariant convolution kernel. The discrete convolution can be written as matrix multiplication $B = KX$, where K is a convolution matrix constructed from the kernel k. A normalization procedure is used in each iteration to account for intensity biases across channels.

2.2 Channel intensity normalization

Intensity data for Illumina sequencing show certain biases, specifically (i) signal strength variation across channels, (ii) signal strength variation across clusters and (iii) signal decay over sequencing cycles. For accurate base-calling, these biases must be addressed through normalization. Traditionally, read normalization is applied to tackle the second and third problems first, in order to address the first problem. In our method, we circumvent the read normalization problem by analyzing the relative intensity ratio of successive calls across sequence reads.

After an initial deconvolution in which cross-talk is corrected, we normalize each channel by scaling the intensities across reads by the same quantile (95%) in the respective channels and select the strongest channel after normalization as candidate base-calls. We then select successive candidate calls that are of different bases and construct a set of linear equations of the form $x_{i_k} - r_k x_{j_k} = 0$, where x_{i_k} and x_{j_k} are the relative intensity of channels in the k-th relation and r_k is the observed intensity ratio for the k-th relation. The set of linear equations is then $Rx = 0$, where R is a $M \times 6$ matrix, with M being the total number of base-calls pairs within consideration. To estimate x, we solve a least-squares problem under the constraint that $\|x\|^2 = 1$. The solution is obtained by solving an eigenvalue problem since it can be formulated into the Rayleigh quotient $\min_{\|x\|^2=1}\|Rx\|^2$, and its solution must satisfy the eigenvalue equation $R^t Rx = \lambda x$. Since the number of base-calls across channels varies, the solution of this optimization problem favors channels that are called frequently. We normalize the problem using the number of base-calls and solve the generalized eigenvalue problem $R^t Rx = \lambda Dx$ where D is a diagonal matrix that records the number of base-calls in each channel. This formulation can be interpreted as finding the stable state of a normalized non-linear diffusion, and is used in normalized cut (Shi and Malik, 2000), Laplacian Eigenmaps (Belkin and Niyogi, 2001), and PageRank (Page *et al.*, 1999). The estimated vector x is the relative intensity of each channel and we use it to normalize each channel in subsequent steps.

2.3 Sparse signal reconstruction through ISD

To perform base-calling we need to reconstruct latent sparse signal X, corresponding only to nucleotide incorporation measurements given a convolution kernel k. A straightforward l^2 optimization problem to estimate latent signal X minimizes $\|B - k * X\|^2$. We know the latent signal is sparser than the observed signal, so we add this property as a constraint to the least squares problem and use an iterative procedure to solve the problem under the sparsity constraint. This idea is termed ISD in the mathematical community (Wang and Yin, 2010), and can also be applied to deconvolution problems stemming from image deblurring applications. In our case, the support (non-zero entries) detected for latent signal X corresponds exactly to base-calls. Assuming X_{Supp} is the signal taking non-zeros only in the support set obtained using our support detection algorithm, we want to find an X that minimizes $\|B - k * X\|^2 + \lambda\|X - X_{\text{Supp}}\|^2$. This optimization outputs a corrected signal subject to the support set constraint. The support detection procedure is critical to the output accuracy—if the support set is correct, we are close to our solution. At the beginning, we have no knowledge of the support set, since that directly tells us the answer. To tackle this, we use an increasing series $\{\lambda_{\text{itr}}\}$ that puts increasing weight on the second constraint. This weight is low at first since the support set is not accurate. As we gradually refine the estimates we increase this weight. In our implementation, support detection is conducted by incorporating the channel-normalization method discussed in the previous section and picking the strongest normalized channel.

We provide further mathematical justification as to why this iterative procedure recovers the clear intensity signals of incorporation events. For reference to the applications in image deblurring we refer to the convoluted signal B as the blurred signal, and to the latent signal X, the clear signal.

Observation 1: Assume the clear signal is a non-negative signal with spikes, the convolution (blur) kernel is non-negative and $\|k\|^1 = 1$, then the convoluted (blurred) signal is denser than the latent (clear) signal.

This observation holds for all blurs since the blur spreads the spikes thus creates more non-zero intensities, so the support set becomes larger with the blurred signal. This observation hints us to design an optimization that favors sparse solutions:

$$\min_X \|B - k * X\|^2 + \lambda\|X\|^p, 0 \le p \le 1.$$

The second term is a sparse-inducing penalty. This sparse regularization problem is well known in wavelet analysis (Mallat, 2009). We also have the following observation.

Observation 2: By comparing the l^p norm ($0 \le p \le 2$) of the clear/blurred signal, we discover that the sparse norm penalty favors the clear signal.

As special cases:

- l^1 norm measures the total variation of the signal, thus the blurred signal and clear signal have the same l^1 norm.
- The l^2 norm of the blur signal is smaller than that of the clear signal.
- The support set for the blurred signal is larger than the clear signal, therefore it has larger l^0 cost.

The above observations suggest that we use a sparse norm to penalize the blur signal and make it resemble the clear signal. Thus, we analyze the deconvolution model with an l^0 penalty:

$$\min_X \|B - k * X\|^2 + \alpha\|X\|^0.$$

By introducing an auxiliary variable and using an exterior penalty technique, the above minimization problem is equivalent to solving the following optimization problem:

$$\min_X \|B - k * X\|^2 + \alpha\|w\|^0 + \lambda\|w - X\|^2, \lambda \to +\infty.$$

One strategy to solve the above optimization is the alternating minimization technique (Wang *et al.*, 2008) and cast the problem into two sub-problems: (i) fixing X and analyzing the terms containing w, we have the w sub-problem:

$$\min_w \|w - X\|^2 + \frac{\alpha}{\lambda}\|w\|^0.$$

The solution can be found by entry-wise comparison (Mallat, 2009; Xu *et al.*, 2013) and the result is the so-called hard thresholding:

$$w_i = \begin{cases} X_i, \text{if } |X_i| > \sqrt{\frac{\alpha}{\lambda}} \\ 0, \text{otherwise} \end{cases}.$$

Then (ii) fix w, and analyze the terms containing X, we have

$$\min_X \|B - k * X\|^2 + \lambda\|w - X\|^2.$$

This optimization problem has the same form with our deconvolution model when $w = X_{\text{Supp}}$. In our ISD method, X_{Supp} is obtained by adaptive hard thresholding, where α is set adaptively to select strictly one non-zero element into the support set by selecting the channel with maximum intensity. Thus, our ISD method solves an optimization problem with an l^0 penalty favoring sparse signals corresponding to nucleotide incorporation.

2.4 Convolution kernel estimation

Given latent signal X we use a least-squares method to estimate the convolution kernel k modeling cross-talk and phasing effects by solving:

$$\min_k \|B - k * X\|^2.$$

Table 1. Base callers accuracy and runtime comparison

	Bustard		AYB		BlindCall slow		BlindCall fast		freeIbis	
Perfect reads	1 446 079		1 532 000		1 509 451		1 508 779		1 530 099	
Error rate (%)	0.29		0.21		0.23		0.23		0.21	
Time (minimum)	17		217		8/12		4/8		9/126	
Assembly results	N50	Maximum	N50	Maximum	N50	Maximum	N50	Maximum	N50	Maximum
5×	610	1122	628	1155	629	1164	623	1167	649	1184
10×	3 375	3469	3198	3322	3382	3487	3389	3485	3306	3418
20×	4466	4478	4627	4637	4511	4523	4470	4483	4333	4357

AYB, accuracy and run times for Bustard. freeIbis and BlindCall for a dataset of 1.9 million reads from a HiSeq 2000 run of PhiX174. BlindCall Fast corresponds to non-iterative version of the blind-deconvolution method. Running times for BlindCall are reported as (processing time/total time), where the total time includes reading intensity data from disk and writing base-calls to disk. For freeIbis, we report the time as (predicting time with single thread/ training time with 10 threads). BlindCall was able to produce base-calls of comparable accuracy to AYB and freeIbis at significantly faster computational time (8 min/12 min versus 217 min and 126 min, respectively). It is also faster than Bustard (8 min/12 min versus 17 min). AYB, freeIbis and BlindCall all improve on Bustard base calls. We also compared assemblies of the PhiX174 genome using reads generated by Bustard, BlindCall, freeIbis and AYB. The reported N50s and Max contig lengths are averages >100 random samples with the corresponding coverage (5×, 10× or 20×). While BlindCall is able to process data at a significantly lower computational cost, the assemblies obtained using BlindCall are of comparable quality to those obtained using AYB or freeIbis.

We estimate convolution kernel k in two distinct steps: we use data from the first four cycles and only model cross-talk in the convolution kernel and use the blind-deconvolution iterative procedure to estimate cross-talk effects. We then fix the components of the convolution kernel corresponding to cross-talk effects for the remaining windows and estimate the components of the convolution kernel corresponding to phasing effects only. We assume the phasing effect is the same across channels.

2.5 Deriving quality scores from deconvolved signal

We measure the quality of a base-call by the ratio of the intensity of the strongest channel and the sum of the two strongest channels after the deconvolution procedure. This number ranges between 0.5 and 1.0 and is used as the raw quality score. This scheme is similar to the one in Illumina's Bustard basecaller. Like most existing base-callers, we calibrate these raw quality scores by aligning reads to the reference genome and mapping raw quality scores to the alignment error rate.

2.6 Validation methods

The following datasets were used to test the accuracy and computational efficiency of BlindCall and state-of-the-art probabilistic methods:

Illumina HiSeq 2000 phiX174: 1 926 928 single-end reads of 101 cycles from a single tile. Data was sequenced at the University of Maryland, College Park and is available for download at http://cbcb.umd.edu/~hcorrada/secgen.
Ibis Test: 200K single-end reads of phiX174 >51 sequencing cycles.
Bordetella pertussis: 100 tiles of 76-cycle single-end reads from the coccobacillus *B.pertussis*, using the complete genome of the Tohoma I strain as a reference.
AYB phiX174: released with AYB and contains human sequence with a PhiX174 spike-in.
The last three datasets were downloaded from the AYB authors' website (http://www.ebi.ac.uk/goldman-srv/AYB/#data).

To calculate accuracy we align the reads based on the phiX174 reference using Bowtie2 (Langmead and Salzberg, 2012) with `--end-to-end` and `--sensitive` settings. Reported error rates are based on reads with no more than five substitution errors, following the methodology in Massingham and Goldman (2012). We used SparseAssembler (Ye *et al.*, 2012) to obtain assemblies from base-calls obtained by each method. To derive assembly statistics, we sub-sampled 100 datasets from the complete set of reads at 5×, 10× and 20× coverage, and perform assemblies on each of these. We report N50 and maximum contig length for each resulting assembly.

Version 1.9.4 of the Off-line basecaller was downloaded from Illumina to run Bustard. Version 2 of AYB was downloaded from http://www.ebi.ac.uk/goldman-srv/AYB. We ran AYB for 5 iterations as per its default setting.

3 RESULTS

BlindCall is implemented as a set of Matlab scripts available at http://cbcb.umd.edu/~hcorrada/secgen. As an example of its computational efficiency, running BlindCall on a single-core Matlab instance on an Intel i7 3610QM laptop with 2.3–3.3 GHz processor and 8 GB of memory, we found that it was able to process 1 million bases/s, or >85 billion bases/CPU day. We note that a significant portion of its running time (50%) is spent on disk IO to read intensity data and write the fasta/fastq outputs. To the best of our knowledge, BlindCall is one of the fastest base-callers available at this time, even though it is implemented in a scripting language. A port of this algorithm into a lower-level language (C/C++) will give further improvements on speed over the current Matlab version.

We compared the running time of BlindCall to the state-of-the art probabilistic base-caller AYB (Massingham and Goldman, 2012) and the state-of-the-art supervised learning method freeIbis (Renaud *et al.*, 2013) on a dataset of 1.9 million reads from a PhiX174 run on an Illumina HiSeq 2000 (Table 1). We found that BlindCall was able to process this dataset ~20 times faster than AYB and 10 times faster than freeIbis while retaining similar accuracy. A plot of per cycle error rate of these base-callers (Fig. 3) shows that all methods produce significant improvements over Bustard, especially in later sequencing cycles. We observed a similar pattern when testing other datasets (Table 2).

We also obtained better assemblies, especially at low coverage, using BlindCall, AYB and freeIbis relative to Bustard

base-calls (Table 1). We also found that the calibrated quality values obtained from BlindCall are very accurate (Fig. 4).

We next compared each base-calling method's ability to scale to longer read lengths by calculating running time as a function of read length for the same dataset (Fig. 5). Like most probabilistic model-based base callers, AYB resorts to a dynamic programming strategy with quadratic running time complexity with respect to the read length. In contrast, BlindCall scales linearly with read length. freeIbis uses supervised learning approach, and while it also scales linearly with read length, its training time is much slower than BlindCall (even using 10 threads for freeIbis, compared to a single thread for BlindCall). Base-callers based on the blind deconvolution framework will be able to scale as sequencers produce longer reads.

4 CONCLUSION

BlindCall is a simple and ultra-fast non-probabilistic base-calling method for Illumina high-throughput sequencing data based on blind deconvolution. We have shown that it provides comparable

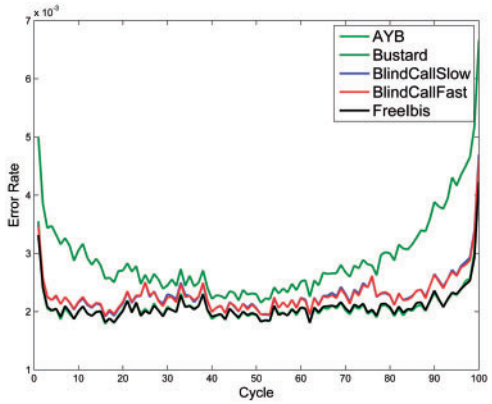

Fig. 3. Third-party base callers improve Bustard per-cycle error rate. We plot error rate of each base-caller per sequencing cycle on the PhiX174 test data. All three base callers significantly improve accuracy over Bustard, especially in later cycles. BlindCall is able to achieve comparable accuracy while processing data at a much faster rate

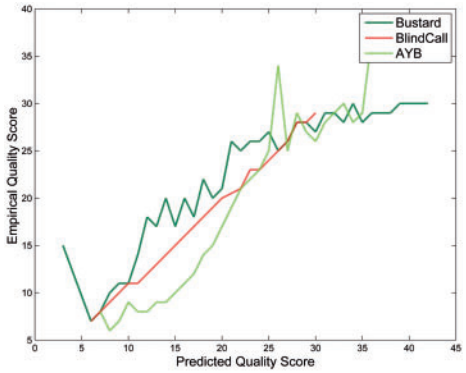

Fig. 4. BlindCall produces accurate calibrated quality scores. We plot observed error rates (on the PHRED scale) for Bustard, AYB and BlindCall as predicted by quality scores and observed high correlation for all base callers

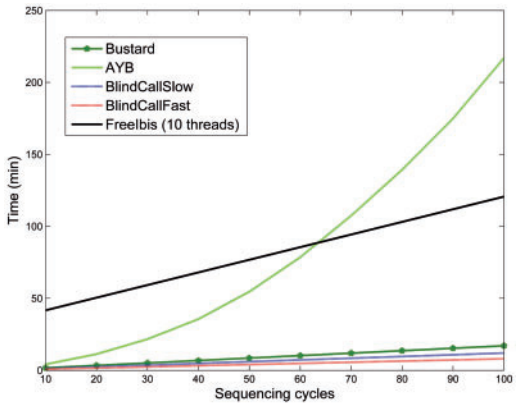

Fig. 5. Base-calling by blind deconvolution is scalable to long read lengths. We compare the computational time of BlindCall with a state-of-the-art probabilistic base caller AYB, the state-of-the-art supervised learning method freeIbis and Illumina's Bustard on the PhiX174 dataset reported in Table 1 as a function of the number of sequencing cycles. Since most model-based base callers resort to a dynamic programming solution, running time is quadratic with respect to the read length. In contrast, BlindCall scales linearly with read length. Base callers based on the blind deconvolution framework will be able to scale as sequencers produce longer reads. freeIbis also scales linearly but is much slower than BlindCall

Table 2. Accuracy comparison

		Ibis Test		*B.pertussis*		PhiX174 (AYB)	
		Perfect reads	Error rate (%)	Perfect reads	Error rate (%)	Perfect reads	Error rate (%)
Bustard		99 834	1.45	1 557 963	2.01	24 478	0.49
AYB		133 537	0.73	2 304 005	1.26	26 878	0.38
BlindCall slow		110 951	1.12	1 902 621	1.61	25 144	0.45
BlindCall fast		105 312	1.26	1 856 286	1.66	24 740	0.47
Time	Slow	0.08/0.3/1		0.11/6/10		0.15/14/22	
	Fast	0.08/0.1/1		0.11/3/8		0.15/7/16	

Accuracy for Bustard, AYB and BlindCall on various datasets. BlindCall was able to produce comparable accuracy to state-of-the-art base callers at significantly faster computational time. All methods improve on Bustard base calls. Run times for BlindCall are reported as (training time/processing time/total time in minutes) where the total time includes reading intensity data from disk and writing base-calls to disk.

accuracy to probabilistic base-calling methods while producing base-calls at rates more than ten times faster.

Almost all probabilistic methods solve the base-calling problem in a 'forward' way, i.e. by setting a set of basis functions and searching for an optimal path, which often leads to dynamic programming solutions. Fitting these statistical methods is computationally expensive, and will not scale as the increase in sequencing throughput continues. Also, a stationarity assumption must be made in order to estimate parameters in these probabilistic methods through a Markov process. In contrast, BlindCall models base-calling as an 'inverse' problem of blind deconvolution, which requires no probabilistic assumptions of the sequencing process.

As steady progress has been made to improve the accuracy of probabilistic methods, we expect that similar progress will be made on non-probabilistic methods based on the blind deconvolution methods described in this article. Furthermore, these methods will be better suited to cope with increased throughput and read lengths of new sequencing technologies.

ACKNOWLEDGEMENTS

We thank Najib El-Sayed and the University of Maryland IBBR Sequencing core for their assistance with test data, James A. Yorke and his research group in University of Maryland for insightful discussions and Gabriel Renaud at Max Planck Institute for assistance with freeIbis.

Funding: National Institute of Health [R01HG005220, in part] and [R01HG006102, in part].

Conflict of Interest: none declared

REFERENCES

Aird,D. *et al.* (2011) Analyzing and minimizing PCR amplification bias in Illumina sequencing libraries. *Genome Biol.*, **12**, R18.

Alkan,C. *et al.* (2011) Limitations of next-generation genome sequence assembly. *Nat. Methods*, **8**, 61–65.

Belkin,M. and Niyogi,P. (2001) Laplacian eigenmaps and spectral techniques for embedding and clustering. *Adv. Neural Inf. Process. Syst.*, **14**, 585–591.

Bentley,D.R. *et al.* (2008) Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, **456**, 53–59.

Bravo,H.C. and Irizarry,R.A. (2010) Model-based quality assessment and base-calling for second-generation sequencing data. *Biometrics*, **66**, 665–674.

Dohm,J.C. *et al.* (2008) Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. *Nucleic Acids Res.*, **36**, e105.

Erlich,Y. *et al.* (2008) Alta-Cyclic: a self-optimizing base caller for next-generation sequencing. *Nat. Methods*, **5**, 679–682.

Illumina (2013) HiSeq Systems Comparison. http://www.illumina.com/systems/hiseq_2500_1500/performance_specifications.ilmn.

Kao,W.-C. and Song,Y.S. (2011) naiveBayesCall: an efficient model-based base-calling algorithm for high-throughput sequencing. *J. Comput. Biol. A J. Comput. Mol. Cell Biol.*, **18**, 365–377.

Kao,W.-C. *et al.* (2009) BayesCall: A model-based base-calling algorithm for high-throughput short-read sequencing. *Genome Res.*, **19**, 1884–1895.

Kircher,M. *et al.* (2009) Improved base calling for the Illumina Genome Analyzer using machine learning strategies. *Genome Biol.*, **10**, R83.

Langmead,B. and Salzberg,S.L. (2012) Fast gapped-read alignment with Bowtie 2. *Nat. Methods*, **9**, 357–359.

Levin,A. *et al.* (2011) Understanding blind deconvolution algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, **33**, 2354–2367.

Mallat,S.G. (2009) *A Wavelet Tour of Signal Processing : the Sparse Way*. Elsevier/Academic Press, Amsterdam/Boston.

Massingham,T. and Goldman,N. (2012) All Your Base: a fast and accurate probabilistic approach to base calling. *Genome Biol.*, **13**, R13.

Page,L. *et al.* (1999) The PageRank citation ranking: bringing order to the web. Stanford InfoLab. Technical Report. Stanford InfoLab.

Renaud,G. *et al.* (2013) freeIbis: an efficient basecaller with calibrated quality scores for Illumina sequencers. *Bioinformatics*, **29**, 1208–1209.

Shi,J.B. and Malik,J. (2000) Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell*, **22**, 888–905.

Wang,Y. and Yin,W. (2010) Sparse signal reconstruction via iterative support detection. *SIAM J. Imaging Sci.*, **3**, 462–491.

Wang,Y.L. *et al.* (2008) A new alternating minimization algorithm for total variation image reconstruction. *SIAM J. Imaging Sci.*, **1**, 248–272.

Xu,L. *et al.* (2013) Unnatural L 0 sparse representation for natural image deblurring. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, 1107–1114.

Ye,C. *et al.* (2012) Exploiting sparseness in de novo genome assembly. *BMC Bioinform.*, **13 (Suppl 6)**, S1.