Resource

BayesCall: A model-based base-calling algorithm for high-throughput short-read sequencing

Wei-Chun Kao,[1] Kristian Stevens,[2] and Yun S. Song[1,3,4]

[1]*Computer Science Division, University of California, Berkeley, Berkeley, California 94720, USA;* [2]*Department of Computer Science, University of California, Davis, Davis, California 95616, USA;* [3]*Department of Statistics, University of California, Berkeley, Berkeley, California 94720, USA*

Extracting sequence information from raw images of fluorescence is the foundation underlying several high-throughput sequencing platforms. Some of the main challenges associated with this technology include reducing the error rate, assigning accurate base-specific quality scores, and reducing the cost of sequencing by increasing the throughput per run. To demonstrate how computational advancement can help to meet these challenges, a novel model-based base-calling algorithm, BayesCall, is introduced for the Illumina sequencing platform. Being founded on the tools of statistical learning, BayesCall is flexible enough to incorporate various features of the sequencing process. In particular, it can easily incorporate time-dependent parameters and model residual effects. This new approach significantly improves the accuracy over Illumina's base-caller Bustard, particularly in the later cycles of a sequencing run. For 76-cycle data on a standard viral sample, phiX174, BayesCall improves Bustard's average per-base error rate by ~51%. The probability of observing each base can be readily computed in BayesCall, and this probability can be transformed into a useful base-specific quality score with a high discrimination ability. A detailed study of BayesCall's performance is presented here.

[Supplemental material is available online at http://www.genome.org. BayesCall is freely available at http://www.cs.berkeley.edu/~yss/bayescall/ under the GNU General Public License.]

Despite broad consensus in the scientific community regarding the value of whole-genome variation studies in large samples, obtaining such data was viewed as too costly. Now, thanks to recent advances in high-throughput sequencing technology (Metzker 2005; Bentley 2006), fast and cost-effective generation of immense amounts of raw, unassembled sequence data has become possible. Soon, obtaining whole-genome sequence information will become routine. This remarkable technological advancement is opening up vast new opportunities for biological sciences and related fields. The whole-genome description of DNA sequence variation will help us understand the biological mechanisms through which genetic polymorphisms affect phenotypes. However, to bring this promise of whole-genome sequencing to fruition, several immediate challenges must be overcome. In particular, we need to develop scalable, accurate computational tools for data acquisition, characterization, and analysis. In this paper, we address an important problem pertinent to data acquisition, namely, developing improved base-calling algorithms for high-throughput sequencing. The main challenges are to reduce the error rate, to assign more accurate base-specific quality scores, and to reduce the cost of sequencing by increasing the throughput per run.

Currently, a few large-scale resequencing projects are either underway or in near inception. In early 2008, an international research consortium announced the 1000 Genomes Project (http://1000genomes.com/), which will resequence the genomes of at least 1000 humans around the world. This effort will provide a comprehensive picture of human genomic variation. A project called 1001 Genomes (http://www.1001genomes.org/) will resequence a large number of *Arabidopsis* genomes, and the *Drosophila* Population Genomics Project (DPGP) (http://www.dpgp.org/) will resequence a similar quantity of *Drosophila melanogaster* genomes for population genomics analysis. The algorithm described in this paper will have direct and immediate impact on such resequencing projects, since accurate base calling has important implications for assembly, polymorphism detection (especially rare ones), and downstream analysis.

There now exist several competing ultra high-throughput sequencing platforms, including the ones from Illumina (formerly Solexa sequencing), 454 Life Sciences (Roche), and Applied Biosystems (SOLiD). While the discussion in this paper applies specifically to Illumina's Solexa technology (hereafter, Illumina), the high-level framework we propose—namely, using graphical models to devise model-based base-calling algorithms—should be applicable to other high-throughput sequencing platforms. Incidentally, the Illumina platform is the main sequencing technology adopted by the genome sequencing projects mentioned above.

Two main existing base-callers for the Illumina platform are Bustard (developed by Illumina themselves) and Alta-Cyclic (Erlich et al. 2008). We will elaborate on Bustard in the Methods section. Alta-Cyclic is a method based on the support vector machine (SVM) and requires "supervised learning using a rich DNA library with a known reference genome" (Erlich et al. 2008). Specifically, it requires using a control lane containing a sample with a known reference genome. It then relies on Bustard and alignment to create a rich training set for supervised learning. Note that using a control lane in every sequencing run to create a rich training library can be burdensome, since it incurs cost and takes up space on the flow cell (defined later) that could otherwise be used to sequence a sample of interest to the biologist. Further, Alta-Cyclic's software architecture is designed to run only in a clustered computing environment, thus limiting its utility.

In this paper, we introduce a novel model-based approach to base calling, founded on the tools of statistical learning. Our main goal is to model the sequencing process to the best of our knowledge, by taking stochasticity into account and by explicitly modeling how errors may arise. In contrast to Alta-Cyclic's SVM

[4]Corresponding author.
E-mail yss@cs.berkeley.edu; fax (510) 642-5775.
Article published online before print. Article and publication date are at http://www.genome.org/cgi/doi/10.1101/gr.095299.109.

19:1884–1895 © 2009 by Cold Spring Harbor Laboratory Press; ISSN 1088-9051/09; www.genome.org

approach, our method, called BayesCall, does not require supervised training and hence does not require using a control lane. Furthermore, BayesCall is flexible enough to incorporate various features of the sequencing process. To illustrate this point, we explicitly model residual effects and incorporate time-dependent parameters into our method. We show that our method significantly improves the accuracy of base calls, particularly in the later cycles of a sequencing run. For 76-cycle data on a standard viral sample, phiX174, BayesCall achieves ~51% improvement over Bustard in the average per-base error rate, and ~21% improvement over Alta-Cyclic. A detailed study of BayesCall's performance is presented in this paper.

In our approach, we obtain base calls by maximizing a posterior distribution of sequences given observed data (i.e., fluorescence intensities) and assuming a uniform prior on sequences. One of the advantages of this approach is that we can readily compute the probability of observing each base, and this probability can be transformed into a useful base-specific quality score. We show that BayesCall not only reduces the error rate substantially, but also produces quality scores with a high discrimination ability (Ewing and Green 1998) that consistently outperforms both Bustard's and Alta-Cyclic's.

We remark that the main novelty of our work is the explicit mathematical description of the sequencing process. Its flexibility should allow one to incorporate additional features that might later be found to be important. Moreover, being a fully parametric model, our approach provides information on the relative importance of various factors that contribute to the observed intensities; such information may become useful for designing an improved sequencing technology. Finally, we believe that the framework introduced here provides a basis for designing a more efficient, accurate algorithm in the future.

BayesCall, a C++ software implementation of the algorithms described in this paper, is available at http://www.cs.berkeley.edu/~yss/bayescall/ under the GNU General Public License. It can be run either on an ordinary PC or on a cluster, and it is fully compatible with the file formats used by Illumina's Genome Analyzer pipeline.

As mentioned above, our objective is to devise a parametric basecalling algorithm that accurately models the sequencing process. Therefore, to take various sources of error into account, it is important to understand how the sequencing platform of interest works. Below, we provide a brief overview of the data acquisition process in the Illumina sequencing platform, for which our base-calling algorithm is designed. We will refer to some of the key concepts mentioned below when we describe our method. Note that, in our present model, we do not try to capture all potential sources of error. In particular, we leave modeling overlapping clusters and amplification errors as an interesting future research problem. In what follows, we provide a brief overview of the Illumina sequencing platform.

Sequencing-by-synthesis

In sequencing-by-synthesis (SBS), a large number of random DNA fragments are clonally amplified and used as templates for sequencing in a highly parallel fashion. See Metzker (2005) and Bentley (2006) for an accessible introduction to currently available SBS platforms and whole-genome resequencing. The Illumina sequencing platform in particular works as follows.

1. A DNA sample from an individual is obtained. The sample contains many copies of the same genomic DNA at full length.
2. The DNA in the sample is randomly fragmented, and a DNA library is created according to a chosen fragment size.
3. Single-stranded DNA fragments from the library are placed on a glass surface (called the flow cell). Each single-stranded DNA fragment gets covalently attached to the surface and gets amplified in the neighborhood of the initial attachment, resulting in a cluster of about 1000 identical copies of the fragment. The flow cell has eight lanes, and the process just described can generate several million clusters of DNA fragments in each lane. (Note: Errors in the amplification step may be a significant source of error in base-calls, but quantifying this effect would require a detailed model of the amplification process. In our work, we assume for simplicity that the amplification step is error-free.)
4. Each single-stranded fragment in a given cluster serves as a template, and SBS is carried out by sequentially building up complementary bases as described below. The sequencing platform uses four distinct fluorescently labeled *terminating bases*, essentially A, C, G, T nucleotides with respective fluorophores and reversible terminators attached. The role of the terminator is to prevent continuation of the complementary strand synthesis.
   (a) A mixture of DNA polymerase and all four terminating base types are added to the flow cell. Ideally, the goal of this process is to add exactly a single complementary terminating base to each template, but, as we describe later, this process is not perfect and some templates may jump ahead (prephasing) or lag behind (phasing) in building up complementary strands. This is a major source of complication for base calling.
   (b) The clusters are then excited by lasers, and CCD images of fluorescence emission are taken four times in optimal wavelengths for the four fluorophores. Many nonoverlapping CCD images need to be taken to cover the entire flow cell. Each lane of the flow cell is divided into 330 (respectively, 100) nonoverlapping *tiles* in Genome Analyzer I (respectively, II).
   (c) Fluorophores and reversible terminators are then chemically removed, and the mixture of polymerase and terminating bases is then added to the flow cell to start the next cycle. Repeating this process for L cycles produces short-reads of length L.

Extracting sequence information

Obtaining actual DNA sequences from the Illumina platform involves two problems: image analysis and base calling. These steps are described below.

Image analysis

One of the primary functions of image analysis is to correct for the imperfect repositioning of the CCD camera between cycles and for chromatic aberration of its lens. Currently, this correction is done by aligning the images from subsequent cycles to a reference image from the initial cycle.

Another important function of image analysis is to identify clusters from their surrounding background, where each cluster contains identical copies of DNA templates (see step 3 in the above description of SBS). At present, this is done using thresholding and segmentation, which are standard image analysis techniques.

The signal for each cluster is characterized as time series data of fluorescence intensities and noise. The intensities for each

cluster are obtained by summing the pixel values within the cluster and subtracting out its neighborhood background signal.

Base calling

For each cluster isolated by image analysis, the base-calling step converts the fluorescence signals into actual sequence data with quality scores. A primary challenge in base calling for this sequencing technology is that the data between cycles are not independent. As explained in step 4a of the description above, some templates in each cluster may undergo imperfect synthesis and jump ahead or fall behind each cycle. These effects become more pronounced in later cycles, thus putting a serious limitation on the read length. There are other stochastic effects in the sequencing process that we need to model to obtain accurate sequence data.

Methods

In this section, we first review how Illumina's base-calling algorithm Bustard works. Then, we describe our new base-calling algorithm, BayesCall. Throughout, we adopt the following notational convention:

1. Multidimensional variables are written in boldface, while scalar variables are written in normal face.
2. The transpose of a matrix $\boldsymbol{M}$ is denoted by $\boldsymbol{M}'$.
3. A one-dimensional normal distribution with mean μ and variance σ^2 is denoted by $\mathcal{N}(\mu, \sigma^2)$.
4. A multivariate normal distribution is denoted by $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with $\boldsymbol{\mu}$ being the mean vector and $\boldsymbol{\Sigma}$ the covariance matrix.
5. In defining random variables, the usual symbol "$\sim$" is used to denote "distributed as."
6. The index t is used to refer to a particular cycle.
7. The index k is used to refer to a particular cluster in a tile.
8. The total number of cycles in a run is denoted by L.

Bustard: Illumina's base-caller

The objective of base calling is to transform observed intensities into sequences. Bustard achieves this goal by performing the three steps described below.

From intensities to concentrations

We use $\boldsymbol{I}_{t,k} = (I^A_{t,k}, I^C_{t,k}, I^G_{t,k}, I^T_{t,k})' \in \mathbb{R}^{4\times 1}$ to denote the fluorescence intensities of the A, C, G, T channels at cycle t in cluster k, after subtracting out the background signals. Define $\boldsymbol{\mathcal{Z}}_{t,k} = (\mathcal{Z}^A_{t,k}, \mathcal{Z}^C_{t,k}, \mathcal{Z}^G_{t,k}, \mathcal{Z}^T_{t,k})' \in \mathbb{R}^{4\times 1}$, where $\boldsymbol{\mathcal{Z}}^A_{t,k}$ denotes the concentration (or number) of templates in cluster k with A-terminators at cycle t; the variables $\mathcal{Z}^C_{t,k}, \mathcal{Z}^G_{t,k}, \mathcal{Z}^T_{t,k}$ are similarly defined. The spectra of the four fluorophores usually overlap (Yin et al. 1996), and this effect can be modeled as

$$\boldsymbol{I}_{t,k} = \boldsymbol{X}\boldsymbol{\mathcal{Z}}_{t,k}, \tag{1}$$

where $\boldsymbol{X} = (X_{ij}) \in \mathbb{R}^{4\times 4}$ is a 16-parameter matrix called the *crosstalk* matrix, with X_{ij} capturing the response in channel i due to fluorescence of a unit concentration of base j. The Bustard base-caller assumes that $\boldsymbol{X}$ is the same for all clusters within a given tile. While some physical properties (such as overlapping dye spectra) modeled by this matrix should remain constant over time, others are not. For instance, the relative intensities of dyes to concentration are affected by variations in focus, temperature, and fluorescent illumination. Because the crosstalk matrix $\boldsymbol{X}$ is the unobserved basis for the intensity space, the problem of estimating the crosstalk matrix in multivariate statistics can be generalized as a factor analysis. Li and Speed (1999) suggested a specialized iterative method for estimating $\boldsymbol{X}$, and that method is employed in Bustard. Upon estimating $\boldsymbol{X}$, its inverse is used in Equation 1 to obtain an estimate of $\boldsymbol{\mathcal{Z}}_{t,k}$.

Renormalization of concentrations

After observed intensities $\boldsymbol{I}_{t,k}$ are transformed into concentrations $\boldsymbol{\mathcal{Z}}_{t,k}$, Bustard tries to address the signal decay problem as follows. First, for each cycle t, compute the average concentration $\overline{\mathcal{Z}}_t$ across a tile using

$$\overline{\mathcal{Z}}_t := \frac{1}{N}\sum_{k=1}^{N}(\mathcal{Z}^A_{t,k} + \mathcal{Z}^C_{t,k} + \mathcal{Z}^G_{t,k} + \mathcal{Z}^T_{t,k}),$$

where N denotes the total number of identified clusters in a given tile. Then, for every cycle t and cluster k, renormalize the concentration $\boldsymbol{\mathcal{Z}}_{t,k}$ by multiplying it with $\overline{\mathcal{Z}}_1/\overline{\mathcal{Z}}_t$. Note that this renormalization leads to the same tile-wide average concentration for all cycles. To avoid introducing more notation, in what follows we use the same symbol $\boldsymbol{\mathcal{Z}}_{t,k}$ to denote renormalized concentrations. For a sequencing run with L cycles, we use $\boldsymbol{\mathcal{Z}}_k = (\boldsymbol{\mathcal{Z}}_{1,k}, \ldots, \boldsymbol{\mathcal{Z}}_{L,k}) \in \mathbb{R}^{4\times L}$ to denote the time series data of renormalized concentrations for cluster k. The subsequent base-calling analysis is done on $\boldsymbol{\mathcal{Z}}_k$.

From renormalized concentrations to sequences

The DNA synthesis process is modeled by the following Markov model:

> Phasing: With probability p, no new base is synthesized.
> Prephasing: With probability q, two bases are synthesized.
> Normal incorporation: With probability $1 - p - q$, exactly one new base is synthesized.

At cycle 0, all templates start at position 0; i.e., no nucleotide has been synthesized. In each subsequent cycle, the position of the terminator in a template changes from i to j according to the following $(L+1)$-by-$(L+1)$ transition matrix $\boldsymbol{P} = (P_{ij})$, where $0 \le i,j \le L$:

$$P_{ij} = \begin{cases} p, & \text{if } j = i, \\ 1-p-q, & \text{if } j = i+1, \\ q, & \text{if } j = i+2, \\ 0, & \textit{otherwise}. \end{cases}$$

The estimation of p and q is done for each lane of the flow cell, and the same p and q are used for all cycles. Note that the (i,j) entry of the matrix $\boldsymbol{P}^t$ corresponds to the probability that a terminator at position i moves to position j after t cycles.

Now, let $\boldsymbol{\mathcal{Z}}^\circ_k = (\boldsymbol{\mathcal{Z}}^\circ_{1,k}, \ldots, \boldsymbol{\mathcal{Z}}^\circ_{L,k}) \in \mathbb{R}^{4\times L}$ denote the concentrations in the ideal case in which there is no phasing or prephasing. Then, the observed concentrations $\boldsymbol{\mathcal{Z}}_k = (\boldsymbol{\mathcal{Z}}_{1,k}, \ldots, \boldsymbol{\mathcal{Z}}_{L,k})$ in the presence of phasing and prephasing are assumed to be related to $\boldsymbol{\mathcal{Z}}^\circ_k$ by $\boldsymbol{\mathcal{Z}}_k = \boldsymbol{\mathcal{Z}}^\circ_k\boldsymbol{Q}$, where $\boldsymbol{Q} = (Q_{jt})$ is an L-by-L matrix with $Q_{jt} = [\boldsymbol{P}^t]_{o,j}$, the probability that a template terminates at position j after t cycles. More explicitly, $\boldsymbol{\mathcal{Z}}_{t,k} = \sum_{j=1}^{L} \boldsymbol{\mathcal{Z}}^\circ_{j,k}[\boldsymbol{P}^t]_{o,j}$. Hence, to invert the phasing and prephasing effects and to infer $\boldsymbol{\mathcal{Z}}^\circ_k$, Bustard computes $\boldsymbol{\mathcal{Z}}_k\boldsymbol{Q}^{-1}$ for each cluster k. Finally, base calling for that cluster k is done as follows: For $j = 1, \ldots, L$, the jth base of the templates is chosen to be the row index of the largest entry in column j of the 4-by-L matrix $\boldsymbol{\mathcal{Z}}_k\boldsymbol{Q}^{-1}$. (Recall that the row indices 1, 2, 3, 4 correspond to A, C, G, T, respectively.)

BayesCall: Our new method

Let $s_{1,k}, s_{2,k}, \ldots, s_{L,k}$ denote the length-L prefix of the true complementary DNA sequence in cluster k. Ideally, at cycle t we want each template in the cluster to be synthesizing the base $s_{t,k}$. However, because of the stochastic effects described earlier, at cycle t some templates might be synthesizing the base pair at position $t' < t$, while others might be synthesizing at position $t' > t$. The extent of this "getting out of phase" phenomenon increases with t.

The underlying graphical model

Here, we introduce a flexible base-calling algorithm based on statistical learning. This framework allows us to handle stochasticity in a more sophisticated way than does Bustard. For example, as we elaborate later, it is possible to incorporate cycle-dependent parameters into our model. For ease of notation, we present our algorithm for the case in which parameters are cycle independent.

The basic model

Let $\boldsymbol{e}_i$ denote a four-component column unit vector with a 1 in the ith entry and zeroes elsewhere. As before, we use the basis with A, C, G, T corresponding to indices 1, 2, 3, 4, respectively. Hence, $s_{t,k} = A$ corresponds to the column vector $\boldsymbol{S}_{t,k} = \boldsymbol{e}_A$, etc. Whenever we modify the character $s_{t,k}$, we modify the corresponding vector $\boldsymbol{S}_{t,k}$ accordingly, and vice versa. We use $\boldsymbol{S}_k = (\boldsymbol{S}_{1,k}, \ldots, \boldsymbol{S}_{L,k})$ to denote the 4-by-L binary *sequence matrix* corresponding to the complementary DNA sequence $s_{1,k}, s_{2,k}, \ldots, s_{L,k}$ in cluster k. The main goal of base calling is to infer $\boldsymbol{S}_k$, and hence the sequence $s_{1,k}, s_{2,k}, \ldots, s_{L,k}$.

Let $\boldsymbol{Q}_t$ denote the column t of the L-*by*-L matrix $\boldsymbol{Q}$ defined above. Recall that the jth component of $\boldsymbol{Q}_t$ *is*$[\boldsymbol{P}^t]_{o,j}$, where the matrix $\boldsymbol{P}^t$ gives the probability distribution of the position of the terminator at cycle t in a given template. Therefore, the product $\boldsymbol{S}_k\boldsymbol{Q}_t \in [0,1]^{4\times 1}$ gives the probability distribution for a given template in cluster k to end with an A-, C-, G-, or T-terminator.

We explicitly model the decay in intensities as follows. We use $\Lambda_{t,k}$ to denote the random variable corresponding to the per-cluster density of templates that are "active" (i.e., able to synthesize further) at cycle t in cluster k. Our model for stochastic changes in $\Lambda_{t,k}$ is

$$\Lambda_{t,k} = (1-d)\Lambda_{t-1,k} + (1-d)\Lambda_{t-1,k^\varepsilon}, \tag{2}$$

where ε is a one-dimensional Gaussian noise with zero mean and variance σ^2. Recall that, in Bustard, $\mathcal{Z}^A_{t,k}, \mathcal{Z}^C_{t,k}, \mathcal{Z}^G_{t,k}$, and $\mathcal{Z}^T_{t,k}$ denote the concentration of templates in cluster k at cycle t with A-, C-, G-, and T-terminators, respectively. In our framework, $\Lambda_{t,k}\boldsymbol{S}_k\boldsymbol{Q}_t$ is analogous to Bustard's $\boldsymbol{\mathcal{Z}}_{t,k} = (\mathcal{Z}^A_{t,k}, \mathcal{Z}^C_{t,k}, \mathcal{Z}^G_{t,k}, \mathcal{Z}^T_{t,k})'$, and we use $\boldsymbol{Z}_{t,k} = (Z^A_{t,k}, Z^C_{t,k}, Z^G_{t,k}, Z^T_{t,k})'$ to denote $\Lambda_{t,k}\boldsymbol{S}_k\boldsymbol{Q}_t$.

We also explicitly model the stochasticity associated with observing intensities as multivariate Gaussian noise. Our basic model is given by the following formulation:

$$\boldsymbol{I}_{t,k} = \boldsymbol{X}\boldsymbol{Z}_{t,k} + \sum_{b\in\{A,C,G,T\}} Z^b_{t,k}\boldsymbol{\eta}^b, \tag{3}$$

where $\boldsymbol{X}$ is the previously described 4-by-4 crosstalk matrix, and $\boldsymbol{\eta}^A, \boldsymbol{\eta}^C, \boldsymbol{\eta}^G, \boldsymbol{\eta}^T \in \mathbb{R}^{4\times 1}$ are independent and identically distributed four-dimensional Gaussian noises each with zero mean and covariance matrix Σ. In what follows, we simplify this basic model to speed up the computation and enrich it to make it more accurate.

Simplification

In practice, the L-dimensional column vector $\boldsymbol{Q}_t$ will have only a few dominant components, with the rest being very small. More precisely, the dominant components will be concentrated about the tth entry. Therefore, for each given t, we can simplify the computation by considering only l positions before t and r positions after t. We refer to this simplification as looking at "a window of size $l + r + 1$ about t" and use $\boldsymbol{Q}_t^\omega$ to denote the L-dimensional column vector obtained from $\boldsymbol{Q}_t$ by setting the entries outside the window to zero. Then, instead of $\boldsymbol{Z}_{t,k}$, we use

$$\boldsymbol{Z}^\omega_{t,k} := \Lambda_{t,k}\boldsymbol{S}_k\boldsymbol{Q}^\omega_t, \tag{4}$$

which is the concentration of templates with A-, C-, G-, T-terminators at cycle t, including the contributions of phased and prephased templates in the window ω about position t. In our implementation, l and r are free parameters that the user may specify. We used $l = 5$ and $r = 5$ in our experiment.

Enrichment

In addition to phasing and prephasing effects, we have observed other residual effects that propagate from one cycle to the next. We speculate that this is perhaps due to incomplete washing of chemicals or an increase in nonspecific illumination. Importantly, we found that modeling such extra residual effects improves the base-call accuracy. In our model, we introduce a parameter α and assume that the observed intensity $\boldsymbol{I}_{t,k}$ at cycle t contains the residual contribution $\alpha(1-d)\boldsymbol{I}_{t-1,k}$ from the previous cycle.

Summary

To recapitulate, our model is

$$\boldsymbol{S}_{t,k} \sim \mathrm{Unif}(\{\boldsymbol{e}_A, \boldsymbol{e}_C, \boldsymbol{e}_G, \boldsymbol{e}_T\}), \tag{5}$$

$$\Lambda_{t,k}|\Lambda_{t-1,k} \sim \mathcal{N}((1-d)\Lambda_{t-1,k}, (1-d)^2\Lambda^2_{t-1,k}\sigma^2), \tag{6}$$

$$\boldsymbol{I}_{t,k}\,|\,\boldsymbol{I}_{t-1,k}, \boldsymbol{S}_k, \Lambda_{t,k} \sim \mathcal{N}(\boldsymbol{\mu}_{t,k}, \|\boldsymbol{Z}^\omega_{t,k}\|^2_2\boldsymbol{\Sigma}), \tag{7}$$

where $\|\bullet\|_2$ denotes the 2-norm and

$$\boldsymbol{\mu}_{t,k} = \begin{cases} \boldsymbol{X}\boldsymbol{Z}^\omega_{t,k}, & \text{if} \quad t = 1, \\ \boldsymbol{X}\boldsymbol{Z}^\omega_{t,k} + \alpha(1-d)\boldsymbol{I}_{t-1,k}, & \text{if} \quad t > 1. \end{cases} \tag{8}$$

We put a uniform (improper) prior on $\Lambda_{1,k}$. A graphical representation of the underlying model is illustrated in Figure 1.

Base calling

Let $\Theta = \{p, q, d, \alpha, \sigma^2, \boldsymbol{X}, \boldsymbol{\Sigma}\}$ denote the set of parameters in our model. We first estimate these parameters for a given tile using the method described presently. Then, our base-call for cluster k is given by $\boldsymbol{S}_k$ in the maximum a posteriori (MAP) estimate of the pair $(\boldsymbol{S}_k, \boldsymbol{\Lambda}_k)$.

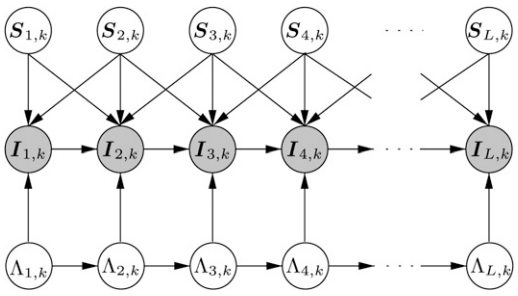

Figure 1. The graphical model corresponding to our base-calling algorithm for cluster k. The observed random variables are the intensities $\boldsymbol{I}_{t,k}$. Base-calling is done by finding the MAP estimates of $\boldsymbol{S}_{t,k}$. In this example, the window size is 3, with $l = r = 1$. See Methods for a detailed description.

In our implementation, we adopt simulated annealing to achieve a faster convergence rate and more accurate MAP estimate. For a given cluster k, the unobserved variables we sample are $\boldsymbol{S}_{t,k}$ and $\Lambda_{t,k}$ for $t \in \{1, \ldots, L\}$. This procedure is detailed below.

Initialization

For each cycle $t \in \{1,\ldots, L\}$ and cluster k, the initialization $\mathring{\boldsymbol{S}}_{t,k}$ of $\boldsymbol{S}_{t,k}$ is obtained by inferring its associated base using

$$s_{t,k} = \underset{b\in\{A,C,G,T\}}{\arg\max} \left(\boldsymbol{X}^{-1}\boldsymbol{I}_{t,k}\right)_b, \tag{9}$$

where $(\boldsymbol{X}^{-1}\boldsymbol{I}_{t,k})_b$ denotes the bth component of the matrix product $\boldsymbol{X}^{-1}\boldsymbol{I}_{t,k}$. The initialization $\mathring{\Lambda}_{t,k}$ of $\Lambda_{t,k}$ is given by

$$\mathring{\Lambda}_{t,k} = \underset{\lambda}{\arg\min}\, (\lambda\boldsymbol{X}\mathring{\boldsymbol{S}}_{t,k} - \boldsymbol{I}_{t,k})'\Sigma^{-1}(\lambda\boldsymbol{X}\mathring{\boldsymbol{S}}_{t,k} - \boldsymbol{I}_{t,k}). \tag{10}$$

This is the maximum likelihood estimate of $\Lambda_{t,k}$ if we assume $\boldsymbol{I}_{t,k} \sim \mathcal{N}(\Lambda_{t,k}\boldsymbol{X}\mathring{\boldsymbol{S}}_{t,k}, \Sigma)$, which is an approximation of our full model.

Subsequent iterations

For iteration i of the Metropolis–Hastings algorithm, let $\boldsymbol{\lambda}_k = (\lambda_{1,k},\ldots,\lambda_{L,k})$ denote the value of $\boldsymbol{\Lambda}_k = (\Lambda_{1,k},\ldots,\Lambda_{L,k})$ from iteration $i-1$. We first sample a position $x \sim \text{Unif}\{1, 2,\ldots, L\}$ to modify and update $\boldsymbol{S}_{x,k}$ according to

$$\mathbb{P}(\boldsymbol{S}_{x,k} = e_b) \propto \left(\boldsymbol{X}^{-1}\boldsymbol{I}_{x,k}\right)_b. \tag{11}$$

Then, we update $\Lambda_{x,k}$ according to a proposal distribution that is an approximation to the conditional distribution $\mathbb{P}(\Lambda_{x,k}|\boldsymbol{I}_k, \boldsymbol{S}_k, \mathcal{D}_x(\lambda_k))$, where $\mathcal{D}_x(\boldsymbol{\lambda}_k)$ denotes $\boldsymbol{\lambda}_k$ with the xth component $\lambda_{x,k}$ deleted. More exactly, we update $\Lambda_{x,k}$ according to the proposal distribution

$$\Lambda_{x,k}\Big|\boldsymbol{S}_k, \lambda_{x-1,k}, \lambda_{x,k}, \lambda_{x+1,k}, \boldsymbol{I}_{x,k}, \boldsymbol{I}_{x-1,k} \sim \mathcal{N}(m_{x,k}, v^2_{x,k}), \tag{12}$$

where the mean $m_{x,k}$ and variance $v^2_{x,k}$ of the normal distribution are given by

$$v^2_{x,k} = \left[\frac{1}{(1-d)^2\lambda^2_{x-1,k}\sigma^2} + \frac{(1-d)^2}{\lambda^2_{x+1,k}\sigma^2} + \frac{\left(\boldsymbol{X}\boldsymbol{S}_k\boldsymbol{Q}^{\omega}_x\right)'\Sigma^{-1}\left(\boldsymbol{X}\boldsymbol{S}_k\boldsymbol{Q}^{\omega}_x\right)}{\left\|z^{\omega}_{x,k}\right\|^2_2}\right]^{-1},$$

$$m_{x,k} = v^2_{x,k}\left[\frac{1}{(1-d)\lambda_{x-1,k}\sigma^2} + \frac{1-d}{\lambda_{x+1,k}\sigma^2} + \frac{\left(\boldsymbol{X}\boldsymbol{S}_k\boldsymbol{Q}^{\omega}_x\right)'\Sigma^{-1}\left(\boldsymbol{I}_{x,k} - r_{x-1,k}\right)}{\left\|z^{\omega}_{x,k}\right\|^2_2}\right],$$

with

$$z^{\omega}_{x,k} = \lambda_{x,k}\boldsymbol{S}_k\boldsymbol{Q}^{\omega}_k,$$

and

$$\boldsymbol{r}_{x-1,k} := \begin{cases} 0, & \text{if} \quad x = 1, \\ \alpha(1-d)\boldsymbol{I}_{x-1,k}, & \text{if} \quad x > 1. \end{cases}$$

A detailed derivation of the above distribution is provided in the Supplemental material. In computing $v^2_{x,k}$ and $m_{x,k}$, all terms involving $x-1$ are set to zero if $x \le 1$, and all terms involving $x+1$ are also set to zero if $x \ge L$.

Simulated annealing

Our target distribution is

$$\mathbb{P}(\boldsymbol{S}_k, \boldsymbol{\Lambda}_k | \boldsymbol{I}_k, \Theta),$$

where $\boldsymbol{\Lambda}_k = (\Lambda_{1,k},\ldots,\Lambda_{L,k})$. To obtain the MAP estimate of $(\boldsymbol{S}_k, \boldsymbol{\Lambda}_k)$ efficiently, we adopt simulated annealing. In an execution with n total iterations, the temperature in the ith iteration is taken as $(n - i + 1)/n$. The main advantage of simulated annealing over the straight Metropolis–Hastings algorithm is its enhanced ability to converge to the maximum in fewer iterations, leading to a more accurate MAP estimate.

Parameter estimation

As above, let Θ denote the set of parameters in our model. The parameters are assumed to be shared across all clusters within a given tile, so Θ will be estimated only once per tile. Our estimation procedure uses the expectation–maximization (EM) algorithm, viewing $\boldsymbol{S}_k$ and $\boldsymbol{\Lambda}_k$ as missing data. Let K denote the number of clusters used in parameter estimation. (In the empirical study discussed in the Results, we used $K = 250$.) The log joint probability over the K clusters can be written as

$$\begin{aligned}
\log\prod_{k=1}^{K}\mathbb{P}(\boldsymbol{I}_k, \boldsymbol{\Lambda}_k, \boldsymbol{S}_k \mid \Theta) &= \log\left\{\prod_{k=1}^{k}\left[\mathbb{P}(\boldsymbol{S}_k|\Theta)\mathbb{P}(\Lambda_{1,k})\mathbb{P}(\boldsymbol{I}_{1,k} \mid \Lambda_{1,k}, \boldsymbol{S}_k, \Theta)\right.\right. \\
&\quad \left.\left.\times\prod_{t=2}^{L}\mathbb{P}(\boldsymbol{I}_{t,k} \mid \boldsymbol{I}_{t-1,k}, \Lambda_{t-1,k}, \Lambda_{t,k}, \boldsymbol{S}_k, \Theta)\mathbb{P}(\Lambda_{t,k} \mid \Lambda_{t-1,k}, \Theta)\right]\right\} \\
&= -\frac{1}{2}\sum_{k=1}^{K}\sum_{t=1}^{L}\left[\log\det(\|\boldsymbol{Z}^{w}_{t,k}\|^2_2\boldsymbol{\Sigma}) + \frac{(\boldsymbol{I}_{t,k} - \boldsymbol{\mu}_{t,k})'\,\boldsymbol{\Sigma}^{-1}(\boldsymbol{I}_{t,k} - \boldsymbol{\mu}_{t,k})}{\|\boldsymbol{Z}^{w}_{t,k}\|^2_2}\right] \\
&\quad -\frac{1}{2}\sum_{k=1}^{K}\sum_{t=1}^{L}\left[2\log((1-d)\Lambda_{t-1,k}\sigma) + \frac{(\Lambda_{t,k} - (1-d)\Lambda_{t-1,k})^2}{((1-d)\Lambda_{t-1,k}\sigma)^2}\right] \\
&\quad + \text{constant},
\end{aligned} \tag{13}$$

where $\boldsymbol{\mu}_{t,k}$ is defined in Equation 8, $\det(\|\boldsymbol{Z}^{\omega}_{t,k}\|^2_2\boldsymbol{\Sigma})$ denotes the determinant of $\|\boldsymbol{Z}^{\omega}_{t,k}\|^2_2\boldsymbol{\Sigma}$, and the constant term in the last line depends only on the total number L of cycles and the number K of clusters. Let Θ_i denote the set of parameters in the ith EM iteration. The estimate of Θ_i is given by

$$\Theta_i = \underset{\Theta}{\arg\max}\, \mathbb{E}_{\Theta_{i-1}}\left[\log\prod_{k=1}^{K}\mathbb{P}(\boldsymbol{I}_k, \boldsymbol{\Lambda}_k, \boldsymbol{S}_k \mid \Theta)\right], \tag{14}$$

where the expectation is taken with respect to $\mathbb{P}(\boldsymbol{\Lambda}_k, \boldsymbol{S}_k \mid \boldsymbol{I}_k, \Theta_{i-1})$ and is computed using Monte Carlo integration with the Metropolis–Hastings algorithm.

In the maximization step, we optimize the expected log-likelihood with respect to one parameter at a time, and iterate through all parameters. One can obtain analytic solutions for updating α, σ^2, $\boldsymbol{X}$, $\boldsymbol{\Sigma}$ in the maximization step, but there is no analytic solution for updating d, p, and q. We therefore employ an interior point method to maximize the expected log-likelihood function with respect to p and q directly and approximate d with a simpler formulation. Details can be found in the Supplemental material.

Cycle–dependent parameters

Both Alta-Cyclic (Erlich et al. 2008) and Bustard assume that parameters are cycle-independent. One novelty of our model is the ability to incorporate cycle-dependent parameters. We will see in the Results that the use of cycle-dependent parameters not only increases the accuracy significantly but also helps us to understand the underlying noise structure better. This may provide information on what can be improved in the Illumina technology.

Although the algorithm described above is for cycle-independent parameters, it is straightforward to incorporate cycle dependency into our probabilistic framework. For concreteness, consider the decay parameter d in Equation 2. Although, in general,

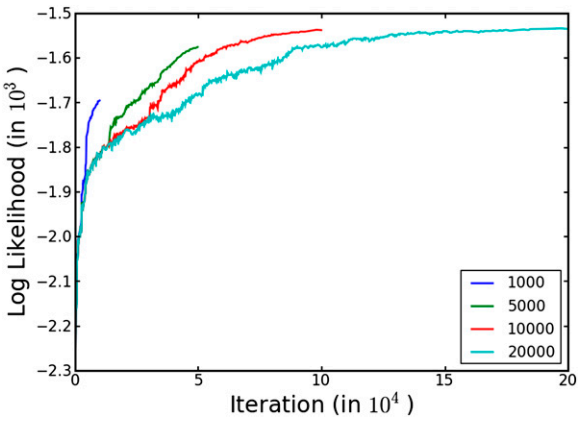

Figure 2. Convergence of simulated annealing with 1000, 5000, 10,000, and 20,000 total iterations. The temperature parameter in the *i*th iteration of simulated annealing is taken as $(n - i + 1)/n$, with n being the total number of iterations. Although using a larger value of n maximizes the likelihood slightly better, the inferred MAP estimate of $\boldsymbol{S}_k$ does not change so much.

the total observed intensity tends to decay with cycles, it neither decays at a constant rate nor is a monotonically decreasing function of time; various stochastic effects (e.g., temperature fluctuation) can lead to fluctuations in the total intensity as time passes. Hence, to model the fluctuation in intensities more accurately, we modify Equation 2 as

$$\Lambda_{t,k} = (1 - d_t)\Lambda_{t-1,k} + (1 - d_t)\Lambda_{t-1,k}\,\varepsilon_t, \tag{15}$$

where the rate d_t now depends on the cycle t, and the one-dimensional Gaussian noise ε_t with zero mean now has cycle-dependent variance σ_t^2. We remark that d_t may take on a negative value for some cycle t. Hence, unlike in Equation 2, it no longer admits interpretation as a pure decay parameter.

More generally, we use the subscript t to denote cycle dependency. Our current implementation can handle the following cycle-dependent parameters: $d_t, \alpha_t, \sigma_t^2, \boldsymbol{X}_t, \boldsymbol{\Sigma}_t$. To avoid over-fitting and to reduce the number of clusters required to estimate parameters, we divide the read length L into nonoverlapping windows of size 5 and assume that the parameters remain constant within each window. Cycle-dependent parameters can be estimated using the EM algorithm. To reduce the fluctuation of parameters between windows, we devised an estimation method that uses information from adjacent windows. See the Supplemental material for details.

We observed that d_t is highly correlated with the average intensity and that the contribution of residual effects tends to grow with t. Such observations may help us to characterize the intrinsic properties of the Illumina platform.

Quality scores

An explicit error model is usually provided by a sequencing platform to assign a measure of uncertainty to each base-call. A natural way to model uncertainty in base calling is to treat the base at position t as a random variable B_t taking values in $\{A, C, G, T\}$. If b denotes the called base, $\mathbb{P}(B_t = b)$ then denotes the probability of that base-call being correct. When defined correctly, a base-calling error model provides a measure of certainty when making inferences, as well as a measure of the information content. In the context of Illumina's technology, such an error model has been used to provide a better sequence alignment (Li et al. 2008), whereas retaining high-quality sub-strings from low-quality reads has been shown to improve coverage under a global alignment model (Rougemont et al. 2008). Recently, Brockman et al. (2008) have suggested a general method to define more accurate quality scores in sequencing-by-synthesis systems and applied it to data from the 454 Life Sciences (Roche) platform. Their method requires first finding a good set of error predictors. In our work, the probability of observing each base can be readily computed and be transformed into a useful base-specific quality score with a high discrimination ability. Below, we provide a detailed description on how quality score is computed in Bustard and BayesCall.

phred *quality scores*

The most common representation of uncertainty in base calling is quality score. Quality scores have been an important feature of modern sequencing platforms since the early days of the human genome project. The classic definition of a quality score was laid out in the implementation of the widely used traditional sequencing base-caller *phred* (Ewing and Green 1998). It defines a quality score $Q_{phred}(b)$ for each called base b as a scaled log of the error probability:

$$Q_{phred}(b) = -10 \log_{10} \mathbb{P}(B_t \neq b).$$

If a base has a quality score of 20, then it has a 1% estimated chance of being wrong, while a score of 30 corresponds to a 0.1% estimated chance of error, and so on. This quality score gained wide

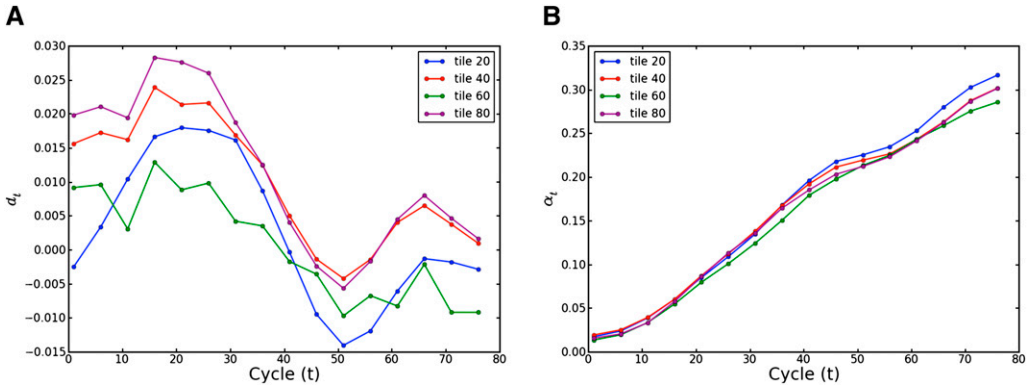

Figure 3. Estimated cycle-dependent parameters for four different tiles of the 76-cycle phiX174 data set. These plots illustrate that parameters change over time and that the residual effect associated with α_t tends grow with time. (*A*) d_t, (*B*) α_t.

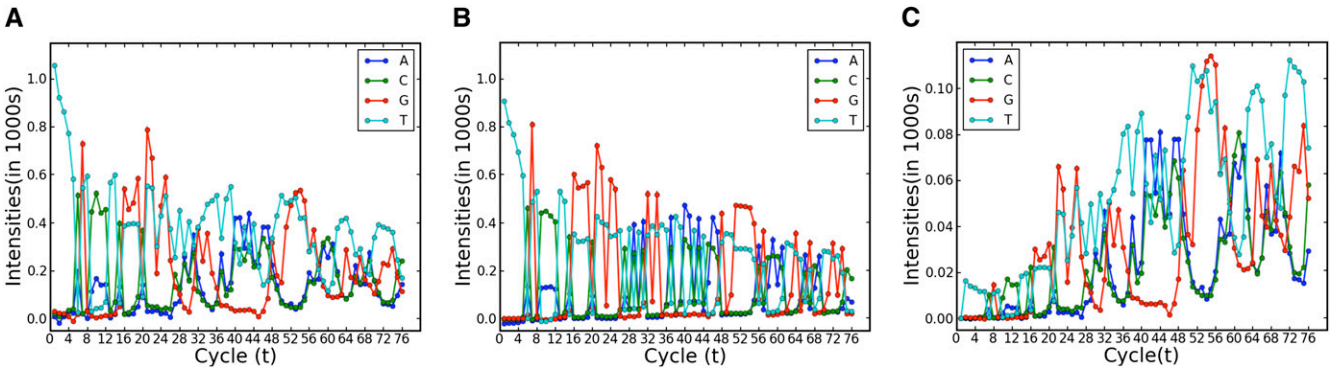

Figure 4. Observed intensities and the decomposition of $\boldsymbol{\mu}_{t,k}$ for a particular cluster k in the 76-cycle phiX174 data. (*A*) Observed intensities $\boldsymbol{I}_{t,k}$. (*B*) The contribution of $\Lambda_{t,k}\boldsymbol{X}_t\boldsymbol{S}_k\boldsymbol{Q}_t^{\omega}$ to $\boldsymbol{\mu}_{t,k}$. (*C*) The contribution of the residual effect $\alpha_t\,(1-d_t)\,\boldsymbol{I}_{t-1,k}$.

popularity with *phred*, and consequently it is commonly referred to as the *phred* quality score.

Base-specific quality scores in Bustard

By definition, the error probability $\mathbb{P}(B_t \neq b)$ of a base-call is the sum of the probabilities of the three alternate bases being correct, i.e.,

$$\mathbb{P}(B_t \neq A) = \mathbb{P}(B_t = C) + \mathbb{P}(B_t = G) + \mathbb{P}(B_t = T),$$

and so on. If we wish to compute $\mathbb{P}(B_t = x)$ for all $x \in \{A,C,G,T\}$ but only a *phred* quality score for the called base is provided, then one would typically assume that the three alternate hypotheses are equally probable. A richer description would try to distinguish between the four possible outcomes. The Illumina platform provides such a full representation of uncertainty, producing a quality value for each of the four possible nucleotides. Briefly, they are defined as follows.

For a given cluster k and a given position t of the template sequence, let b denote the row index of the largest entry in column t of $\boldsymbol{Z}_k\boldsymbol{Q}^{-1}$ and let $(y^A_{t,k}, y^C_{t,k}, y^G_{t,k}, y^T_{t,k})'$ denote that column. Note that b corresponds to the base called by Bustard. To assign base-specific quality scores, Bustard considers the random vector $\boldsymbol{Y}_{t,k} = (Y^A_{t,k}, Y^C_{t,k}, Y^G_{t,k}, Y^T_{t,k})'$ distributed as

$$\boldsymbol{Y}_{t,k} \sim \mathcal{N}(y^b_{t,k}\boldsymbol{e}_b, \boldsymbol{M}_{t,k}),$$

where the covariance matrix $\boldsymbol{M}_{t,k}$ is defined as $\boldsymbol{M}_{t,k} = (\boldsymbol{X}^{-1}\varepsilon_{t,k})(\boldsymbol{X}^{-1}\varepsilon_{t,k})'$, with $\boldsymbol{X}$ being the crosstalk matrix and $\varepsilon_{t,k} = (\varepsilon^A_{t,k}, \varepsilon^C_{t,k}, \varepsilon^G_{t,k}, \varepsilon^T_{t,k})$ a noise vector at cycle t in cluster k estimated by Illumina's image analysis software *Firecrest*. Note that the mean vector of $\boldsymbol{Y}_{tk}$ is non-zero only in the bth component.

For a particular alternate base $a \neq b$, the difference $Y^b_{t,k} - Y^a_{t,k}$ is then distributed as

$$Y^b_{t,k} - Y^a_{t,k} \sim \mathcal{N}(y^b_{t,k}, \boldsymbol{c}'\boldsymbol{M}_{t,k}\boldsymbol{c}),$$

```
True sequence    : TTTTTCGTCCCCTTCGGGGCGGTGGTCTATAGTGTTATTAATATCAAGTTGGGGGAGCACATTGTAGCATTGTGCC
Bustard result   : TTTTTCGTCCCCTTCGGGGCGGTGGTTTTTAGTTTTTTTAATATTAAGTTTGGGGGGCACATTTTTTTATTTTTTC
Errors                                    * *   * *     *    *   *     * ***   * **

True sequence    : TTTTTCGTCCCCTTCGGGGCGGTGGTCTATAGTGTTATTAATATCAAGTTGGGGGAGCACATTGTAGCATTGTGCC
Alta-Cyclic result: TTTTTCGTCCCCTTCGGGGCGGTGGTCTATAGTTTTATTAATATTAAGTTGGGGGAGCACATTGTAGCATTTTGCC
Errors                                    *       *                     *
```

Figure 5. Base-calling results for the particular cluster discussed in Figure 4. Our method BayesCall called all 76 bases correctly for this particular cluster. In contrast, Bustard made 14 base-calling errors, with 13 of them incorrectly called as T. Alta-Cyclic made three errors, with all of them incorrectly called as T. (*) Base-calling errors. In general, both Bustard and Alta-Cyclic tend to suffer more from the "anomalous T" effect than does our method. (See text for details.)

where $\boldsymbol{c} = (c_A, c_C, c_G, c_T)'$ is a four-dimensional column vector with $c_b = 1$, $c_a = -1$, and 0 for the other components. Finally, define

$$\mathbb{P}(a) = \mathbb{P}(Y^b_{t,k} - Y^a_{t,k} \leq y^a_{t,k}), \tag{16}$$

and let

$$\mathbb{P}(b) = 1 - \sum_{a \neq b} \mathbb{P}(a). \tag{17}$$

In Bustard, the quality score assigned to base $b \in \{A,C,G,T\}$ is

$$Q_{\text{Bustard}}(b) = 10\log_{10}\left[\frac{\mathbb{P}(b)}{1-\mathbb{P}(b)}\right],$$

with $\mathbb{P}(b)$ given by Equations 16 and 17. According to this definition, the quality score $Q_{\text{Bustard}}(b)$ of the called base b may actually be lower than the quality score $Q_{\text{Bustard}}(a)$ of some alternate base $a \neq b$. Note that $Q_{\text{Phred}}(b)$ and $Q_{\text{Bustard}}(b)$ are related as follows:

$$Q_{phred}(b) = 10\log_{10}\left[1 + 10^{Q_{\text{Bustard}}(b)/10}\right].$$

Base-specific quality scores in our method

Since we have constructed a probabilistic model based on statistical learning, it would be natural to use the posterior probability of observing a certain base to define its quality score. However, to estimate the posterior probability accurately, it is necessary to take a large number of samples in the Metropolis–Hastings algorithm, which is computationally expensive. Furthermore, since we adopt the simulated annealing method to obtain the MAP estimate, the same set of samples cannot be reused to estimate the posterior probability. To avoid these problems, we define our quality score with the conditional observation likelihood of the MAP estimate.

For a given cluster k, let $\boldsymbol{S}^*_k = \left(\boldsymbol{S}^*_{1,k}, \ldots, \boldsymbol{S}^*_{L,k}\right)$ and $\boldsymbol{\Lambda}^*_k = (\Lambda^*_{1,k}, \ldots, \Lambda^*_{L,k})$ denote the MAP estimates of $\boldsymbol{S}_k$ and $\boldsymbol{\Lambda}_k$, respectively. Further, let $\mathcal{D}_t(\boldsymbol{S}^*_k)$ be the same as $\boldsymbol{S}^*_k$ with the tth component deleted, and let $\mathcal{R}^b_t(\boldsymbol{S}^*_k)$ be the same as $\boldsymbol{S}^*_k$ with tth component replaced with $\boldsymbol{e}_b$. Then, for position $t \in \{1,\ldots,L\}$ and $b \in \{A,G,C,T\}$, we define

$$\mathbb{P}(b) := \mathbb{P}\left[\boldsymbol{S}_{t,k} = \boldsymbol{e}_b, \boldsymbol{I}_{t,k} \mid \boldsymbol{I}_{t-1,k}, \mathcal{D}_t(\boldsymbol{S}^*_k), \Lambda^*_{t,k}, \Theta\right], \tag{18}$$

which is equal to

$$\frac{1}{4}\mathbb{P}\left[\boldsymbol{I}_{t,k} \mid \boldsymbol{S}_{t,k} = \boldsymbol{e}_b, \boldsymbol{I}_{t-1,k}, \mathcal{D}_t(\boldsymbol{S}^*_k), \Lambda^*_{t,k}, \Theta\right],$$

Table 1. Comparison of error rates on phiX174 data

Method	36-cycle data		76-cycle data	
	By base[a]	By read[b]	By base[a]	By read[b]
(1) Bustard	0.01313	0.29806	0.01557	0.39484
(2) Alta-Cyclic[c]	NA	NA	0.00969	0.31150
(3) BayesCall	0.00805	0.17757	0.00762	0.23188
Improvement of (3) over (1)	39%	40%	51%	41%
Improvement of (3) over (2)	NA	NA	21%	26%

[a]The "by-base" error rate refers to the ratio of the number of miscalled bases to the total number of base-calls made.
[b]The "by-read" error rate refers to the ratio of the number of reads each with at least one miscalled base to the total number of reads considered.
[c]Alta-Cyclic was run only on the 76-cycle data.
NA, not available.

for all $b \in \{A, C, G, T\}$, since we assume a uniform prior over $\boldsymbol{S}_{t,k}$. It then follows that

$$\mathbb{P}(b) = \frac{\mathbb{P}\left[\boldsymbol{I}_{t,k}|\boldsymbol{S}_{t,k}=\boldsymbol{e}_b, \boldsymbol{I}_{t-1,k}, \mathcal{D}_t(\boldsymbol{S}_k^*), \Lambda_{t,k}^*, \Theta\right]}{\sum_{x\in\{A,C,G,T\}} \mathbb{P}\left[\boldsymbol{I}_{t,k}|\boldsymbol{S}_{t,k}=\boldsymbol{e}_x, \boldsymbol{I}_{t-1,k}, \mathcal{D}_t(\boldsymbol{S}_k^*), \Lambda_{t,k}^*, \Theta\right]}$$

$$= \frac{\mathbb{P}\left[\boldsymbol{I}_{t,k}|\boldsymbol{I}_{t-1,k}, \mathcal{R}_t^b(\boldsymbol{S}_k^*), \Lambda_{t,k}^*, \Theta\right]}{\sum_{x\in\{A,C,G,T\}} \mathbb{P}\left[\boldsymbol{I}_{t,k}|\boldsymbol{I}_{t-1,k}, \mathcal{R}_t^x(\boldsymbol{S}_k^*), \Lambda_{t,k}^*, \Theta\right]},$$

where $\mathbb{P}[\boldsymbol{I}_{t,k}\,|\,\boldsymbol{I}_{t-1,k}, \mathcal{R}_t^x(\boldsymbol{S}_k^*), \Lambda_{t,k}^*, \Theta]$ is the observation likelihood of our model defined in Equation 7.

In our method, the quality score assigned to base $b \in \{A,C,G,T\}$ is

$$Q_{\text{BayesCall}}(b) = 10\log_{10}\left[\frac{\mathbb{P}(b)}{1-\mathbb{P}(b)}\right],$$

with $\mathbb{P}(b)$ given by Equation 18.

Results

In this section, we summarize the performance of our model-based method BayesCall. Since Illumina's base-caller Bustard is currently the most widely used method, we largely focus on benchmarking our method in comparison to it. The performance of Alta-Cyclic (Erlich et al. 2008) is also discussed, though not as extensively as the other two methods.

Short-read data and experiment setup

To test the performance of our method, we used sequencing data on a standard viral sample, phiX174, provided to us by Illumina and the DPGP Sequencing Lab at UC Davis. The data from Illumina were obtained from a 36-cycle run on the Genome Analyzer I (GA-I) platform, while the data from UC Davis were obtained from a 76-cycle run on the Genome Analyzer II (GA-II) platform. phiX174 has a known assembled genome, which we assumed to be correct. The viral sample has two outstanding features. First, in contrast to a typical diploid genomic sample, there is only one unique underlying parent DNA molecule. Secondly, the parent DNA sample is relatively short and complex, and thus has a well-characterized sequence alignment.

To create a classification data set, sequences determined by the Bustard base-caller were aligned against the assembled reference sequence. Illumina's Eland alignment algorithm allows only mismatches and is constrained to at most two of them in the first 32 bp, effectively removing some reads with very high error rates from the test. We further discarded reads that could not be aligned to the reference genome with at least 70% identity (out of the entire read length). In the remaining set of reads, we then assumed that all mismatches in alignment are base-calling errors. The same set of reads was used to test the accuracy of Alta-Cyclic and BayesCall, thus favoring Bustard a bit. For each data set, we used four tiles for base calling, resulting in 19,063 reads and 686,268 bases for the 36-cycle data, and 268,997 reads and 20,443,772 bases for the 76-cycle data.

Recall that Alta-Cyclic requires supervised learning using a rich training set. We ran Alta-Cyclic only on the 76-cycle data, since we did not have a sufficiently large training set available to us

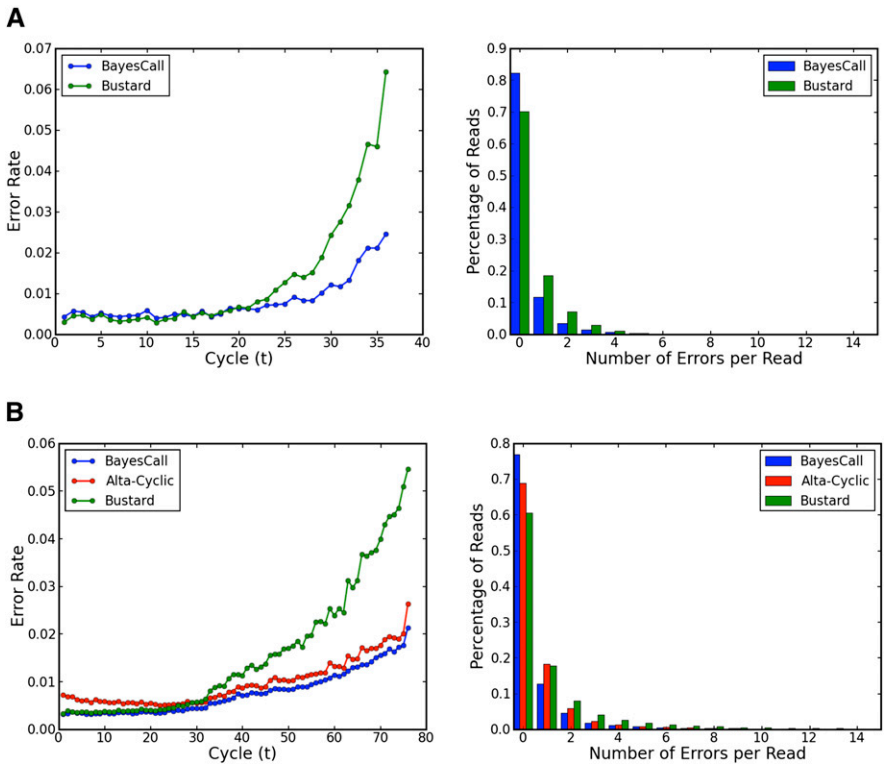

Figure 6. Average per-cycle error rates and histograms for the number of errors per read. BayesCall had substantially lower error rates in later cycles compared with Bustard, and the difference tended to increase with cycles. Further, BayesCall produced substantially more perfect reads than did Bustard. Alta-Cyclic was run only on the 76-cycle phiX174 data set. BayesCall had a lower average error rate than that of Alta-Cyclic's for all cycles. Note that although Alta-Cyclic is more accurate than Bustard in later cycles, the opposite is true for earlier cycles. (*A*) Results for the 36-cycle data set. (*B*) Results for the 76-cycle data set.

Table 2. Confusion matrices for the 76-cycle phiX174 data on GA-II

Base called by Bustard	A	C	G	T
True A	0.98896	0.00337	0.00296	0.00470
True C	0.00877	0.97716	0.00336	0.01071
True G	0.00485	0.00252	0.98617	0.00646
True T	0.00289	0.00517	0.00665	0.98529
Alta-Cyclic				
True A	0.99404	0.00235	0.00141	0.00220
True C	0.00586	0.98678	0.00144	0.00591
True G	0.00336	0.00091	0.99288	0.00285
True T	0.00104	0.00273	0.00338	0.99285
BayesCall				
True A	0.99548	0.00179	0.00129	0.00144
True C	0.00500	0.98908	0.00132	0.00460
True G	0.00349	0.00090	0.99384	0.00176
True T	0.00158	0.00339	0.00333	0.99170

The (x,y) entry in the table corresponds to the percentage of times that base x is called as base y. These tables illustrate that BayesCall is in general less "confused;" that is, the diagonal entries for BayesCall are higher than the corresponding entries for the other methods, except for the (T,T) entry in Alta-Cyclic. Further, BayesCall suffers less from the "anomalous T" effect than does Bustard or Alta-Cyclic; i.e., the (A,T), (C,T), and (G,T) entries for BayesCall are less than the corresponding entries for Bustard or Alta-Cyclic.

for the 36-cycle data. The 76-cycle data from GA-II consisted of 100 tiles in total. In constructing the training set for Alta-Cyclic, we omitted the four test tiles mentioned above and used the remaining 96 tiles. Alta-Cyclic used Bustard and alignment to create a training set containing about 100,000 reads.

Convergence of simulated annealing

Figure 2 shows the convergence of simulated annealing with 1000, 5000, 10,000, and 20,000 total iterations. Recall that the temperature parameter in the ith iteration of simulated annealing is taken as $(n - i + 1)/n$, where n denotes the total number of iterations. We remark that the MAP estimate for $n = 1000$ has a lower likelihood. However, although using a larger value of n maximizes the likelihood better, the inferred MAP estimate of $\boldsymbol{S}_k$ does not change so much, if at all. We conclude that the total number n of annealing iterations can be chosen to be less than 10,000.

Parameters in BayesCall

In contrast to Alta-Cyclic (Erlich et al. 2008), our method does not require supervised learning. In BayesCall, it is possible to estimate the parameters of the underlying model using a small number of randomly chosen clusters. To estimate the cycle-dependent parameters d_t, α_t, σ_t^2, $\boldsymbol{X}_t$, $\boldsymbol{\Sigma}_t$ for a given tile, we randomly chose a total of 250 clusters from the tile and used the algorithm described in the Methods to perform parameter estimation. (We also tried using 150 and 500 clusters in the training set. The accuracy of base-calls changed very little, while the running time of parameter estimation scaled roughly linearly with the number of clusters. See supplemental material for details.) Here, we highlight the cycle dependency of the parameters in our model.

Recall that Bustard tries to address the signal decay problem by renormalizing concentrations, whereas in our method, we explicitly model how the per-cluster density of active templates evolves according to Equation15. Figure 3A illustrates why the rate d_t should be modeled as a cycle-dependent parameter. It shows that d_t varies significantly over cycles. The parameter α_t captures extra residual effects in addition to phasing and prephasing that propagate from one cycle to the next. Our estimates of α_t for the 76-cycle data are shown in Figure 3B. This plot illustrates that the extra residual effects captured by α_t grow with cycle t. We found that it is important to model extra residual effects in later cycles to improve the base-call accuracy.

A detailed example

For a particular cluster k in the 76-cycle phiX174 data, Figure 4A shows the observed intensity $\boldsymbol{I}_{t,k}$ for $t = 1,\ldots, 76$. We can see that in later cycles, the observed intensity of base T increases abnormally; we refer to this as the "anomalous T" effect. This was also noted as an anomaly in Erlich et al. (2008). Because of this anomaly, Bustard produced many base-calling errors—14 errors, to be exact, with 13 of them incorrectly called as T. Alta-Cyclic made three base-calling errors on this cluster, with all three being incorrectly called as T. (See Figure 5 for details; errors are indicated by asterisks).

In contrast, our method BayesCall was able to call all 76 bases correctly. We can explain this as follows. The inferred mean $\boldsymbol{\mu}_{t,k}$ of the distribution for $\boldsymbol{I}_{t,k}$ can be decomposed into $\Lambda_{t,k}\boldsymbol{X}_t\boldsymbol{S}_k\boldsymbol{Q}_t^{\omega}$ and $\alpha_t\,(1-d_t)\,\boldsymbol{I}_{t-1,k}$, the latter capturing extra residual effects. This decomposition is illustrated in Figure 4, B and C. From this figure, we see that the aforementioned anomaly can be attributed to growing residual effects (illustrated in Fig. 4C) in later cycles. Because of its ability to decouple such residual effects from other stochastic effects, BayesCall was able to call all bases correctly for this cluster. Clearly the presence of artifacts of this kind supports the value of modeling residual effects.

Summary of base-call accuracy

Shown in Table 1 are the overall error rates of the three base-calling methods, averaged over four tiles for each data set. In the table, "by-base" error rate refers to the ratio of the number of miscalled bases to the total number of base-calls made, while "by-read" error rate refers to the ratio of the number of reads each with at least one

Table 3. Joint errors in Bustard and BayesCall for the 76-cycle phiX174 data on GA-II

No. of Bustard errors	No. of BayesCall errors								
	0	1	2	3	4	5	6	7	8
0	0.5979	0.0069	0.0005	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
1	0.1061	0.0672	0.0030	0.0005	0.0001	0.0000	0.0000	0.0000	0.0000
2	0.0354	0.0245	0.0181	0.0015	0.0003	0.0001	0.0000	0.0000	0.0000
3	0.0148	0.0123	0.0085	0.0038	0.0011	0.0002	0.0001	0.0000	0.0000
4	0.0073	0.0065	0.0052	0.0033	0.0018	0.0006	0.0002	0.0000	0.0000
5	0.0037	0.0037	0.0033	0.0027	0.0019	0.0011	0.0004	0.0002	0.0001
6	0.0019	0.0022	0.0021	0.0019	0.0016	0.0011	0.0007	0.0003	0.0002
7	0.0012	0.0013	0.0015	0.0013	0.0011	0.0010	0.0007	0.0005	0.0002
8	0.0007	0.0008	0.0011	0.0009	0.0009	0.0008	0.0007	0.0005	0.0004

The (x,y) entry corresponds to the percentage of reads with x errors in Bustard and y errors in BayesCall. For $x < y$, the upper diagonal entry (x,y) is generally much smaller than the corresponding the lower diagonal entry (y,x), thus, indicating that BayesCall generally produces sequence reads with substantially fewer errors than does Bustard. See Figure 7 for a heat plot of joint error a larger range of x and y.

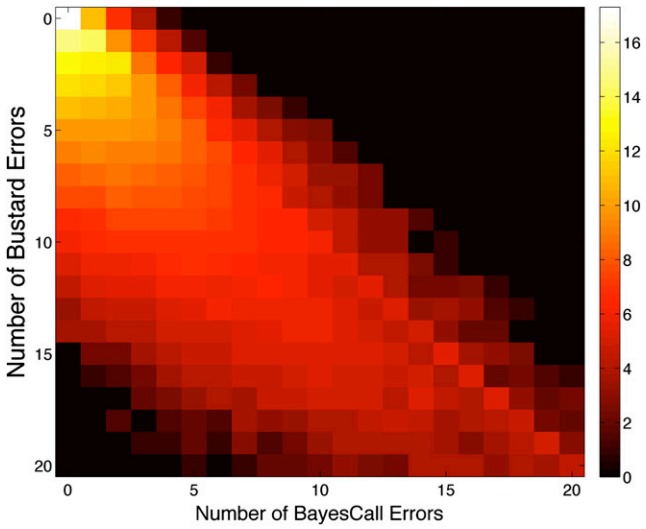

Figure 7. Heat plot of joint errors in Bustard and BayesCall for the 76-cycle phiX174 data. This plot depicts the joint error matrix shown in Table 3. The (x,y) entry in the plot corresponds to $\log_2$ of the number of reads with x errors in Bustard and y errors in BayesCall. This plot clearly illustrates that BayesCall generally produces sequence reads with substantially fewer errors than that produced by Bustard.

miscalled base to the total number of reads considered. For both data sets we considered, our new method BayesCall produced significantly more accurate base-calls than did Bustard. For the 76-cycle phiX174 data from GA-II, BayesCall achieved an improvement of ~51% over Bustard in the by-base error rate, and ~41% improvement in the by-read error rate. For the same data set, BayesCall outperformed Alta-Cyclic as well, achieving an improvement of ~21% in the by-base error rate, and ~26% improvement in the by-read error rate.

For a finer comparison of the methods, we examined the per-cycle error rates, illustrated in the plots in the left column of Figure 6. Bustard and BayesCall had comparable error rates in the first 25 or so cycles. However, BayesCall had substantially lower error rates in later cycles compared with Bustard, and the difference tended to increase with cycles. This observation suggests that it is feasible to run the sequencing machine for longer cycles and obtain useful sequence information for longer reads by using an improved base-calling algorithm such as ours. For the 76-cycle data, BayesCall had a lower average error rate than that of Alta-Cyclic's for every cycle. Although Alta-Cyclic was considerably more accurate than Illumina's base-caller Bustard in later cycles, note that the opposite was true for earlier cycles.

Shown in the right column of Figure 6 are histograms for the number n_e of errors per read. As mentioned before, BayesCall produced substantially more perfect reads (i.e., with $n_e = 0$) than did either Bustard or Alta-Cyclic. Furthermore, for $n_e > 1$, the number of reads with n_e errors was smaller in BayesCall than in either Bustard or Alta-Cyclic.

A statistic that is of interest is the percentage $c(x,y)$ of times that base x is called as base y by a base-calling algorithm. This information can be summarized in what we call a *confusion matrix* $\boldsymbol{C}$, in which the (x,y) entry corresponds with $c(x,y)$. For the 76-cycle data on GA-II, the confusion matrices $\boldsymbol{C}_{\text{Bustard}}$, $\boldsymbol{C}_{\text{Alta-Cyclic}}$, and $\boldsymbol{C}_{\text{BayesCall}}$ of Bustard, Alta-Cyclic, and BayesCall, respectively, are shown in Table 2. For this data set, every diagonal entry (x,x) (i.e., the percentage of correct calls) in $\boldsymbol{C}_{\text{BayesCall}}$ is larger than that in $\boldsymbol{C}_{\text{Bustard}}$, while every off-diagonal entry (x,y) (i.e., the percentage of erroneous calls) in $\boldsymbol{C}_{\text{BayesCall}}$ is smaller than that in $\boldsymbol{C}_{\text{Bustard}}$. Also, BayesCall is less "confused" than Alta-Cyclic, in that the diagonal entries in $\boldsymbol{C}_{\text{BayesCall}}$ are higher than the corresponding entries in $\boldsymbol{C}_{\text{Alta-Cyclic}}$, except for the (T,T) entry.

Recall that the "anomalous T" effect refers to calling abnormally high percentages of other bases erroneously as T. In Table 2, the T column of $\boldsymbol{C}_{\text{Bustard}}$ is abnormally high, suggesting that Bustard suffers from the "anomalous T" effect. BayesCall suffers much less from this effect than does Bustard or Alta-Cyclic; i.e., the (A,T), (C,T), and (G,T) entries of $\boldsymbol{C}_{\text{BayesCall}}$ are significantly smaller than the corresponding entries in $\boldsymbol{C}_{\text{Bustard}}$ or $\boldsymbol{C}_{\text{Alta-Cyclic}}$.

To compare further the performance of Bustard and BayesCall, we examined the percentage of reads with x errors in Bustard and y errors in BayesCall. For $x,y \leq 8$, these numbers are summarized in the joint error matrix shown in Table 3. Figure 7 illustrates the overall pattern for a larger range of x and y. What is abundantly clear is that the upper diagonal entries (x,y), for $x < y$, are much smaller than the corresponding lower diagonal entries (y,x). This indicates that BayesCall generally produces sequence reads with substantially fewer errors than does Bustard.

Gain in accuracy from modeling extra residual effects

In BayesCall, extra residual effects are captured by the parameter α_t. To examine the advantage of modeling residual effects, we considered the following three cases of BayesCall: (1) No residual effect (i.e., impose $\alpha_t = 0$ for all t). (2) Constant residual effect (i.e., assume α_t is equal to some constant α for all t). (3) Cycle-dependent residual effect (i.e., α_t allowed to change over cycles). For this study, we considered a single tile from the 76-cycle data set; the resulting test set contained 68,272 reads and 5,188,672 bases. Parameter estimation was performed for each case allowing d_t, σ_t^2, $\boldsymbol{X}_t$, and $\boldsymbol{\Sigma}_t$ to be cycle-dependent.

Results from this study are shown in Table 4. Even without modeling extra residual effects (i.e., with $\alpha_t = 0$ for all t), BayesCall achieved an improvement of 38% (respectively, 29%) over Bustard in the by-base (respectively, by-read) error rate. This gain in accuracy illustrates the utility of using cycle-dependent parameters d_t, σ_t^2, $\boldsymbol{X}_t$, and $\boldsymbol{\Sigma}_t$. BayesCall's improvement over Bustard increased slightly when a model with a constant residual effect was used. When the full model with cycle-dependent α_t was used, the improvement over Bustard further increased to 55% in the by-base error rate and 45% in the by-read error rate.

Discrimination ability of quality scores

To compare the utility of quality scores, we define the discrimination ability $D(\varepsilon)$ at error tolerance ε as follows: Sort the bases

Table 4. Gain in accuracy from different levels of modeling extra residual effects

Method	By-base error rate	By-read error rate
Bustard	0.01737	0.44431
BayesCall ($a_t = 0$ for all t)	0.01076	0.31328
BayesCall (a_t = some a for all t)	0.00965	0.28748
BayesCall (cycle-dependent a_t)	0.00783	0.24398

In BayesCall, extra residual effects are captured by the parameter α_t. We used cycle-dependent parameters d_t, σ_t^2, $\boldsymbol{X}_t$, and $\boldsymbol{\Sigma}_t$ in all three cases of BayesCall. Even without modeling extra residual effects (i.e., with $\alpha_t = 0$ for all t), BayesCall achieved an improvement of 38% over Bustard in the by-base error rate. When the full model with cycle-dependent α_t was used, the improvement over Bustard increased to 55%.

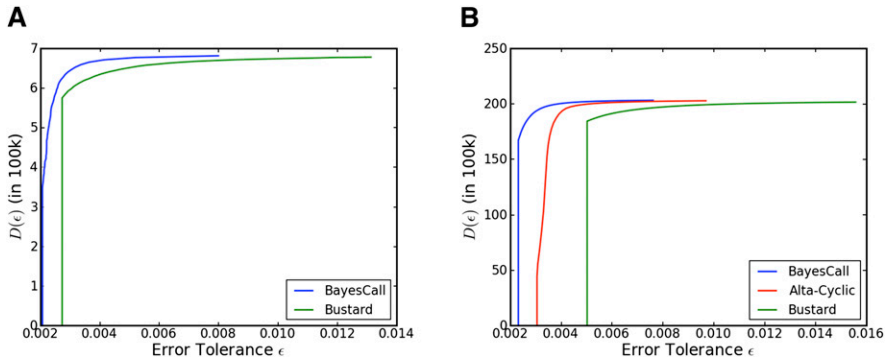

Figure 8. Discrimination ability ***D***(ε) of quality scores at error tolerance ε. We define ***D***(ε) as the number of correctly called bases at error tolerance ε. BayesCall maintains a high discrimination ability which outperforms both Bustard's and Alta-Cyclic's. (*A*) Results for the 36-cycle phiX174 data set. (*B*) Results for the 76-cycle phiX174 data set.

according to their quality scores, from the highest to the lowest. Then, go down that sorted list until the error rate surpasses ε. The number of correctly called bases up to that point corresponds to $D(\varepsilon)$. This notion is essentially the same as the discrimination ability defined in Ewing and Green (1998). Shown in Figure 8 are the plots of $D(\varepsilon)$ for the data sets considered in this paper. We see that BayesCall not only reduces the error rate, but also produces base-specific quality scores with a high discrimination ability that consistently outperform both Bustard's and Alta-Cyclic's.

An error tolerance ε implies a corresponding quality score cutoff for each base-caller; bases with quality scores below the cutoff may be considered unreliable and get thrown out. This means that for all reasonable error tolerances, BayesCall produces a much larger number of bases and consequently a lower cost per base. This is a much stronger result than simply having a lower error rate at a single cutoff.

Difference between GA-I and GA-II

The known upgrades between the GA-I and GA-II platforms include an imaging system with a wider field of view, a larger flow cell, and a larger CCD. Additional Peltier devices were employed to regulate the flow cell temperature better throughout the sequencing process. Further, a slight modification to the flow cell design also reduced the number of poorly imaged tiles caused by displaced fluid from the oil immersion microscope.

The sequencing chemistry was improved, resulting in a noticeable decrease in the phasing rate p. This change is illustrated in Table 5, which shows our estimates of the phasing and prephasing rates for the phiX174 data on GA-I and GA-II. As a consequence of the chemistry improvement, the rate d_t also decreased in magnitude and in the amount of fluctuation. However, we still observed a marked residual effect (parametrized by α_t), as illustrated in Figure 3B. Residual effects contribute significantly to the cycle-to-cycle decrease in the platform's signal-to-noise ratio. To the extent we can better model it, we can obtain longer reads for a given error tolerance.

Discussion

In this paper, we provided a detailed description of Illumina's sequencing platform and its current base-calling algorithm Bustard. Beyond its direct relevance to our method, the description should be of interest to the many users of the platform. We then introduced a novel model-based approach to base calling, which we believe is the first such detailed approach in the literature. In contrast to an SVM-based approach, or other black box machine-learning approaches, a model-based method provides quantitative and testable insight into the underlying chemical process. It also provides a systematic way to estimate various interdependent parameters. Further, the framework based on graphical models can easily incorporate cycle-dependent parameters; we showed here that this extension significantly improves the accuracy of basecalls.

In addition to transparency, the utility of our model-based approach was validated by showing substantially reduced error rates relative to both Bustard and Alta-Cyclic (Erlich et al. 2008). We observed that one of the primary issues with Bustard is its way of handling the phasing and prephasing phenomenon, specifically, its method of creating a large matrix to model the process and then subsequently inverting it to *deconvolve* the process. We observed that the values in the inverted matrix are often orders of magnitude larger than the original matrix. This leads to large amplifications of the additive instrument noise. We also observed significant residual effects and explicitly incorporated them into our model. Our approach resulted in significant improvement in accuracy in later cycles, as the phasing and prephasing effects, as well as residual effects, became more pronounced.

We note that parameter estimation in our method is done without supervision. This has obvious advantages over supervised learning methods, such as Alta-Cyclic, which require preclassified training data. More generally, our method has advantages over the methods that rely on a control lane with a reference library to estimate parameters. In addition to being able to handle variability across lanes, our method can save the cost associated with using a control lane.

Our parameter estimation procedure produces very good results even when using a training set consisting of only a few hundred clusters randomly chosen from the tens to hundreds of thousands of clusters on a given tile. In general, the ability to estimate local parameters using a small fraction of the data allows one to model the significant differences between different tiles and lanes. To speed up the estimation procedure, one could adopt the following strategy: First, try to obtain lane-wide parameter estimates by training on a set of clusters evenly distributed throughout a given lane. Then, for each tile of the lane, randomly select a training set of clusters from the tile and obtain tile-specific parameter estimates by performing a few iterations of the EM

Table 5. Average estimates of phasing and prephasing rates for the phiX174 data on GA-I and GA-II

Platform	Phasing rate p	Prephasing rate q
GA-I	6.8×10^{-4}	3.4×10^{-3}
GA-II	3.0×10^{-8}	3.3×10^{-3}

Because of improvement in the sequencing chemistry, the phasing rate p for GA-II is substantially lower than that for GA-I. However, prephasing and residual effects seem to persist in GA-II.

algorithm, initialized with the lane-wide estimates from the first step.

Our current implementation uses standard Monte Carlo techniques in both estimating parameters and base calling. One drawback of this approach is that it is computationally intensive (see Supplemental material for running time statistics). In BayesCall, the estimation of cycle-dependent parameters can be performed progressively as the sequencing machine runs, so we believe that the bottleneck is in base calling. We are currently investigating methods to achieve faster parameter estimation and base calling. We believe that the running time of BayesCall can be sped up considerably while maintaining high accuracy.

The primary goal of our work was to introduce a better model. We hope that our detailed and descriptive approach to modeling the underlying sequencing process will spark additional work in this area. In the development of our method, we observed that a large number of unalignable sequences were due to crosstalk between clusters. In addition to the aforementioned advantages, since we explicitly model the number of active templates, we note that it is possible to build a model on top of our current model to take inter-cluster crosstalk effects into account.

Acknowledgments

We thank Charles H. Langley, Junming Yin, and Kurt Tadayuki Miller for useful discussions, and we thank Illumina for providing us with their data. This research is supported in part by NIH grants R01-HG002942 (K.S.) and R00-GM080099 (W.C.K.,Y.S.S.), NSF CAREER grant DBI-0846015 (W.C.K., Y.S.S.), an Alfred P. Sloan Research Fellowship (Y.S.S.), and a Packard Fellowship for Science and Engineering (Y.S.S.).

References

Bentley DR. 2006. Whole-genome re-sequencing. *Curr Opin Genet Dev* **16:** 545–552.

Brockman W, Alvarez P, Young S, Garber M, Giannoukos G, Lee WL, Russ C, Lander ES, Nusbaum C, Jaffe DB, et al. 2008. Quality scores and SNP detection in sequencing-by-synthesis systems. *Genome Res* **18:** 763–770.

Erlich Y, Mitra P, Delabastide M, McCombie W, Hannon G. 2008. Alta-Cyclic: A self-optimizing base caller for next-generation sequencing. *Nat Methods* **5:** 679–682.

Ewing B, Green P. 1998. Base-calling of automated sequencer traces using *phred*. II. Error probabilities. *Genome Res* **8:** 186–194.

Li L, Speed T. 1999. An estimate of the crosstalk matrix in four-dye fluorescence-based DNA sequencing. *Electrophoresis* **20:** 1433–1442.

Li H, Ruan J, Durbin R. 2008. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res* **18:** 1851–1858.

Metzker ML. 2005. Emerging technologies in DNA sequencing. *Genome Res* **15:** 1767–1776.

Rougemont J, Amzallag A, Iseli C, Farinelli L, Xenarios I, Naef F. 2008. Probabilistic base calling of Solexa sequencing data. *BMC Bioinformatics* **9:** 431. doi: 10.1186/1471-2105-9-431.

Yin Z, Severin J, Giddings MC, Huang WA, Westphall MS, Smith LM. 1996. Automatic matrix determination in four dye fluorescence-based DNA sequencing. *Electrophoresis* **17:** 1143–1150.

Received April 21, 2009; accepted in revised form August 4, 2009.

BayesCall: A model-based base-calling algorithm for high-throughput short-read sequencing

Wei-Chun Kao, Kristian Stevens and Yun S. Song

Genome Res. 2009 19: 1884-1895 originally published online August 6, 2009
Access the most recent version at doi:10.1101/gr.095299.109