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

Wei-Chun Kao,1 Kristian Stevens,2 and Yun S. Song1,3,4

1Computer Science Division, University of California, Berkeley, Berkeley, California 94720, USA; 2Department of Computer Science, University of California, Davis, Davis, California 95616, USA; 3Department 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, 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
problem. In what follows, we provide a brief overview of the Illu-
clusters and amplification errors as an interesting future research
sources of error. In particular, we leave modeling overlapping
calling algorithm is designed. We will refer to some of the key
process in the Illumina sequencing platform, for which our base-
accurate algorithm in the future.

BayesCall not only reduces the error rate substantially, but also
transformed into a useful base-specific quality score. We show that
BayesCall, a C++ software implementation of the algorithms
in Genome Analyzer I (re-
www.genome.org

Sequencing-by-synthesis

In sequencing-by-synthesis (SBS), a large number of random DNA
fragments are clonally amplified and used as templates for se-
quencing 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 se-
quencing 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 ampli-
ﬁed 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 plat-
form uses four distinct ﬂuorescently labeled terminating bases,
especially 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 syn-
thesis.

(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 nonover-
lapping CCD images need to be taken to cover the entire ﬂow
cell. Each lane of the ﬂow cell is divided into 330 (respec-
tively, 100) nonoverlapping tiles in Genome Analyzer I (re-
spectively, 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 \( M \) is denoted by \( M^T \).
3. A one-dimensional normal distribution with mean \( \mu \) and variance \( \sigma^2 \) is denoted by \( N(\mu, \sigma^2) \).
4. A multivariate normal distribution is denoted by \( N(\mu, \Sigma) \), with \( \mu \) being the mean vector and \( \Sigma \) the covariance matrix.
5. In defining random variables, the usual symbol \( \sim \) is used to denote "distributed as."
6. The index \( i \) 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 \( \mathbf{I}_i = (I_{i1}, I_{i2}, I_{i3}, I_{i4}) \in \mathbb{R}^{4 \times 1} \) 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 \( \mathbf{Z}_{ik} = (\mathbf{Z}_{i1}^0, \mathbf{Z}_{i2}^0, \mathbf{Z}_{i3}^0, \mathbf{Z}_{i4}^0) \in \mathbb{R}^{4 \times 1} \), where \( \mathbf{Z}_{i1}^0 \) denotes the concentration (or number) of templates in cluster \( k \) with A-terminators at cycle \( t \); the variables \( \mathbf{Z}_{i2}^0, \mathbf{Z}_{i3}^0, \mathbf{Z}_{i4}^0 \) are similarly defined. The spectra of the four fluorophores usually overlap (Yin et al. 1996), and this effect can be modeled as

\[
\mathbf{I}_{ik} = \mathbf{X} \mathbf{Z}_{ik},
\]

where \( \mathbf{X} = (X_0) \in \mathbb{R}^{4 \times 4} \) is a 16-parameter matrix called the cross-talk matrix, with \( X_0 \) capturing the response in channel \( i \) due to fluorescence of a unit concentration of base \( j \). The Bustard base-caller assumes that \( \mathbf{X} \) 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 \( \mathbf{X} \) 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 \( \mathbf{X} \), and that method is employed in Bustard.

Upon estimating \( \mathbf{X} \), its inverse is used in Equation 1 to obtain an estimate of \( \mathbf{Z}_{ik} \).

Renormalization of concentrations
After observed intensities \( \mathbf{I}_i \) are transformed into concentrations \( \mathbf{Z}_{ik} \), Bustard tries to address the signal decay problem as follows. First, for each cycle \( t \), compute the average concentration \( \mathbf{Z}_t \) across a tile using

\[
\mathbf{Z}_t = \frac{1}{N} \sum_{k=1}^{N} (\mathbf{Z}_{ik}^0 + \mathbf{Z}_{ik}^1 + \mathbf{Z}_{ik}^2 + \mathbf{Z}_{ik}^3),
\]

where \( N \) denotes the total number of identified clusters in a given tile. Then, for every cycle \( t \) and cluster \( k \), renormalize the concentration \( \mathbf{Z}_{ik} \) by multiplying it with \( \mathbf{Z}_t \mathbf{Z}_{ik} \). 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 \( \mathbf{Z}_{ik} \) to denote renormalized concentrations. For a sequencing run with \( L \) cycles, we use \( \mathbf{Z}_t = (\mathbf{Z}_{1k}, \ldots, \mathbf{Z}_{Lk}) \in \mathbb{R}^{4 \times L} \) to denote the time series data of renormalized concentrations for cluster \( k \). The subsequent base-calling analysis is done on \( \mathbf{Z}_t \).

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 \( \mathbf{P} \) (where \( 0 \leq i,j \leq L \)):

\[
\mathbf{P} = \begin{pmatrix}
    p & 0 & 0 & \cdots & 0 \\
    1-p-q & q & 0 & \cdots & 0 \\
    0 & 0 & 0 & \cdots & q \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & 0 & \cdots & 0
\end{pmatrix}
\]

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 \( \mathbf{P}^t \) corresponds to the probability that a terminator at position \( i \) moves to position \( j \) after \( t \) cycles.

Now, let \( \mathbf{Z}_t = (\mathbf{Z}_{1k}, \ldots, \mathbf{Z}_{Lk}) \in \mathbb{R}^{4 \times L} \) denote the concentrations in the ideal case in which there is no phasing or prephasing. Then, the observed concentrations \( \mathbf{Z}_t = (\mathbf{Z}_{1k}, \ldots, \mathbf{Z}_{Lk}) \) in the presence of phasing and prephasing are assumed to be related to \( \mathbf{Z}_t \) by \( \mathbf{Z}_t = \mathbf{Z}_t \mathbf{Q} \), where \( \mathbf{Q} \) is an \( (L+1) \)-by-\((L+1)\) matrix with \( Q_{ij} = \mathbf{P}_{ij} \), the probability that a template terminates at position \( j \) after \( t \) cycles. More explicitly, \( \mathbf{Z}_t = \mathbf{Z}_t \mathbf{Q} \) such that each cluster \( k \) is modeled as follows: For \( j = 1, \ldots, L \), the \( j \)th base of the templates is chosen to be the row index of the largest entry in column \( j \) of the 4-by-\((L+1)\) matrix \( \mathbf{Z}_t \mathbf{Q} \). (Recall that the row indices 1, 2, 3, 4 correspond to A, C, G, T, respectively.)
BayesCall: A model-based base-calling algorithm

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 $e_t$ denote a four-component column unit vector with a 1 in the $t$th 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 $S_{t,k} = e_t$, etc. Whenever we modify the character $s_{t,k}$, we modify the corresponding vector $S_{t,k}$ accordingly, and vice versa. We use $S_t = (S_{t,1}, \ldots, S_{t,L})$ 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 $S_t$ and hence the sequence $s_{1,k}, s_{2,k}, \ldots, s_{L,k}$.

Let $Q_t$ denote the column $t$ of the $L$-by-$L$ matrix $Q$ defined above. Recall that the $j$th component of $Q_t$ is $[Q]_{t,j} = q_{t,j}$, where the matrix $Q$ gives the probability distribution of the position of the terminator at cycle $t$ in a given template. Therefore, the product $S_t Q_t \in [0,1]^{4 \times L}$ 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 $A_{t,k}$ 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 $A_{t,k}$ is

$$A_{t,k} = (1 - d)A_{t-1,k} + (1 - d)A_{t-1,k'},$$

where $d$ is a one-dimensional Gaussian noise with zero mean and variance $\sigma^2$. Recall that, in Bustard, $Z^{i}_{t,k}$, $Z^{o}_{t,k}$, and $Z^{2}_{t,k}$ denote the concentration of templates in cluster $k$ at cycle $t$ with $A$, $C$, $G$, and $T$-terminators, respectively. In our framework, $A_{t,k} S_{t,k} Q_t$ is analogous to Bustard's $Z_{t,k}$ is $(Z^{i}_{t,k}, Z^{o}_{t,k}, Z^{2}_{t,k})$, and we use $Z_{t,k} = (Z^{i}_{t,k}, Z^{o}_{t,k}, Z^{2}_{t,k})'$ to denote $A_{t,k} S_{t,k} Q_t$.

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

$$I_{t,k} = XZ_{t,k} + \sum_{b \in \{A,C,G,T\}} Z_{t,k}^{b} \eta^{b},$$

where $X$ is the previously described 4-by-4 cross-talk matrix, and $\eta^{1}, \eta^{C}, \eta^{G}, \eta^{T} \in \mathbb{R}^{4 \times 4}$ are independent and identically distributed four-dimensional Gaussian noises each with zero mean and covariance matrix $\Sigma$. 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 $Q_t$ will have only a few dominant components, with the rest being very small. More precisely, the dominant components will be concentrated about the $t$th 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 $Q_{t}^{w}$ to denote the $L$-dimensional column vector obtained from $Q_t$ by setting the entries outside the window to zero. Then, instead of $Z_{t,k}$, we use

$$Z_{t,k}^{w} = A_{t,k} S_{t,k} Q_{t}^{w},$$

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 $\omega$ 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 $\alpha$ and assume that the observed intensity $I_{t,k}$ at cycle $t$ contains the residual contribution $\alpha(1 - d)I_{t-1,k}$ from the previous cycle.

Summary

To recapitulate, our model is

$$S_{t,k} \sim \text{Unif}\{(e_1, e_2, e_3, e_4)\},$$

$$A_{t,k} | A_{t-1,k} \sim N((1 - d)A_{t-1,k}, (1 - d)^2 A_{t-1,k}^2 \sigma^2),$$

$$I_{t,k} | I_{t-1,k}, S_{t,k}, A_{t,k} \sim N(\mu_{t,k}, \| Z_{t,k}^{w} \|_2^2 \Sigma),$$

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

$$\mu_{t,k} = \| XZ_{t,k}^{w} \| + \alpha (1 - d)I_{t-1,k},$$

if $t = 1$.

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

Base calling

Let $\Theta = \{p, q, d, \alpha, \sigma, X, \Sigma\}$ 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 $\hat{S}_{k}$ in the maximum a posteriori (MAP) estimate of the pair $(S_{k}, A_{k})$.

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

For each cluster \( k \) and \( t \in \{1, \ldots, L\} \), the initialization \( S_{t,k} \) of \( S_{t,k} \) is obtained by inferring its associated base using

\[
s_{t,k} = \text{arg max}_{b \in \{A,C,G,T\}} \left( X^{-1} I_{b,k} \right)_t,
\]

where \( X^{-1} I_{b,k} \) denotes the \( b \)th component of the matrix product \( X^{-1} I_{k} \). The initialization \( F_{t,k} \) of \( F_{t,k} \) is given by

\[
F_{t,k} = \text{arg min}_{k} \left( \lambda X S_{t,k} - I_{t,k} \right) \Sigma^{-1} \left( \lambda X S_{t,k} - I_{t,k} \right).
\]

This is the maximum likelihood estimate of \( F_{t,k} \) if we assume \( I_{t,k} \sim N(\Lambda_{t,k} X S_{t,k}, \Sigma) \), which is an approximation of our full model.

Subsequent iterations

For iteration \( i \) of the Metropolis–Hastings algorithm, let \( \Lambda_{i,k} = (\Lambda_{1,k}, \ldots, \Lambda_{K,k}) \) denote the value of \( \Lambda \) from \( \Lambda_{i-1,k} \) to \( \Lambda_{i,k} \) for iteration \( i = 1 \). We first sample a position \( x = \text{Unif}(1, 2, \ldots, L) \) and modify and update \( S_{t,k} \) according to

\[
F(S_{t,k} = x) \times \left( X^{-1} I_{t,x,k} \right)_s.
\]

Then, we update \( \Lambda_{i,k} \) according to a proposal distribution that is an approximation to the conditional distribution \( F(\Lambda_{i,k} | I_{t,k}, S_{t,k}, D_{t}(\Lambda_{i,k})) \), where \( D_{t}(\Lambda_{i,k}) \) denotes \( \Lambda_{i,k} \) with the \( t \)th component \( \lambda_{i,t} \) deleted. More exactly, we update \( \Lambda_{i,k} \) according to the proposal distribution

\[
\Lambda_{i+1,k} = \text{arg min}_{k} \left( \lambda X S_{t,k} - I_{t,k} \right) \Sigma^{-1} \left( \lambda X S_{t,k} - I_{t,k} \right).
\]

where the mean \( m_{t,k} \) and variance \( \nu^2_{t,k} \) of the normal distribution are given by

\[
\nu^2_{t,k} = \frac{1}{(1-d)^2} \lambda_{t,k}^2 \sigma^2 + (1-d)^2 \frac{(X S_{t,k}, Q_{t,k}) \Sigma^{-1} (X S_{t,k}, Q_{t,k})}{\nu^2_{t,k}},
\]

\[
m_{t,k} = \nu^2_{t,k} \left[ \frac{1}{(1-d)\lambda_{t,k} \sigma^2} + \frac{(X S_{t,k}, Q_{t,k}) \Sigma^{-1} (X S_{t,k}, Q_{t,k})}{\nu^2_{t,k}} \right]^{-1}.
\]

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 \( \alpha, \sigma^2, \lambda, \Sigma \) in the maximization step, but there is no analytic solution for updating \( d, p, q \).

A detailed derivation of the above distribution is provided in the Supplemental material. In computing \( \nu^2_{t,k} \) and \( m_{t,k} \), all terms involving \( x \) - 1 are set to zero if \( x = 1 \), and all terms involving \( x + 1 \) are also set to zero if \( x \geq L \).

Simulated annealing

Our target distribution is

\[
F(S_{t,k} | I_{t,k}, \Theta),
\]

where \( \Theta = (\Lambda_{1,k}, \ldots, \Lambda_{K,k}) \). To obtain the MAP estimate of \( (S_{t,k}, \Theta) \) efficiently, we adopt simulated annealing. In an execution with \( n \) total iterations, the temperature in the \( i \)th 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 \( \Theta \) denote the set of parameters in our model. The parameters are assumed to be shared across all clusters within a given tile, so \( \Theta \) will be estimated only once per tile. Our estimation procedure uses the expectation–maximization (EM) algorithm, viewing \( S_{t,k} \) and \( A_{t,k} \) 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

\[
\log \prod_{k=1}^{K} \mathbb{P}(I_{t,k}, A_{t,k}, S_{t,k} | \Theta) = \log \left( \prod_{k=1}^{K} \mathbb{P}(S_{t,k} | \Theta) \mathbb{P}(A_{t,k} | S_{t,k} \Theta) \mathbb{P}(I_{t,k} | A_{t,k}, S_{t,k}, \Theta) \right)
\]

\[
= - \frac{1}{2} \sum_{t=1}^{L} \sum_{k=1}^{K} \left[ \log \det(\Sigma) + \frac{1}{2} \left( I_{t,k} - m_{t,k} \right)^2 \Sigma^{-1} \left( I_{t,k} - m_{t,k} \right) \right] + \frac{1}{2} \sum_{t=1}^{L} \sum_{k=1}^{K} \left( 2 \log(1 - C_{0}) + \frac{(\Lambda_{t,k} - (1 - d) \Lambda_{t-1,k})^2}{((1 - d) \Lambda_{t-1,k})^2} \right) + \text{constant},
\]

where \( m_{t,k} \) is defined in Equation 8, \( \det(\Sigma) \) denotes the determinant of \( \Sigma^{-1} \), and the constant term in the last line depends only on the total number \( L \) of cycles and the number \( K \) of clusters. Let \( \Theta \) denote the set of parameters in the \( k \)th EM iteration. The estimate of \( \Theta \) is given by

\[
\Theta_{k} = \text{arg max}_{\Theta} \log \prod_{t=1}^{K} \mathbb{P}(I_{t,k}, A_{t,k}, S_{t,k} | \Theta).
\]

where the expectation is taken with respect to \( \mathbb{P}(S_{t,k} | I_{t,k}, \Theta_{k-1}) \) and is computed using Monte Carlo integration with the Metropolis–Hastings algorithm.

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,
where the rate \(\lambda_{t,k}\) now depends on the cycle \(t\), and the one-dimensional Gaussian noise \(\epsilon_t\) with zero mean now has cycle-dependent variance \(\sigma_t^2\). 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: \(d_t, \alpha_t, \sigma_t^2, \mathbf{x}_t, \mathbf{z}_t\). 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, \(P(B_t = b)\) 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 (Rougemonet 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 their 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 \(Q_{\text{phred}}(b)\) for each called base \(b\) as a scaled log of the error probability:

\[
Q_{\text{phred}}(b) = -10 \log_{10} P(B_t \neq b).
\]

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
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 $b$ of $ZQ^T$ and let $(y_{tk}, y_{tk}', y_{tk}^{\prime\prime}, y_{tk}^{\prime\prime\prime})$ denote that column. Note that $b$ corresponds to the base called by Bustard. To assign base-specific quality scores, Bustard considers the random vector $Y_{tk} = (y_{tk}, y_{tk}', y_{tk}^{\prime\prime}, y_{tk}^{\prime\prime\prime})$ distributed as

$$Y_{tk} \sim N(y_{tk}^b \epsilon_0, M_{b,k}),$$

where the covariance matrix $M_{b,k}$ is defined as $M_{b,k} = (X^{-1}e_b, X^{-1}e_b, X^{-1}e_b, X^{-1}e_b)$ with $X$ being the crosstalk matrix and $e_b = (e_{tk}^b, e_{tk}^{\prime\prime}, e_{tk}^{\prime\prime\prime}, e_{tk}^{\prime\prime\prime})$ a noise vector at cycle $t$ in cluster $k$ estimated by Illumina’s image analysis software Firecrest. Note that the mean vector of $Y_{tk}$ is non-zero only in the $b$th component.

For a particular alternate base $a \neq b$, the difference $y_{tk}^a - y_{tk}^b$ is then distributed as

$$y_{tk}^a - y_{tk}^b \sim N(y_{tk}^a - y_{tk}^b, \epsilon M_{b,k} \epsilon),$$

**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.)
for all \( b \in \{A, C, G, T\} \), since we assume a uniform prior over \( S_{i,k} \). It then follows that

\[
P(b) = \frac{\mathbb{P}(I_{1,k} | S_{i,k} = e, I_{1-1,k}, D_t(S_{i,k}), \Lambda_{t,k}, \Theta)}{\sum_{x \in \{A,C,G,T\}} \mathbb{P}(I_{1,k} | S_{i,k} = x, I_{1-1,k}, D_t(S_{i,k}), \Lambda_{t,k}, \Theta)}
\]

\[
= \frac{\mathbb{P}(I_{1,k} | I_{1-1,k}, R_t^x(S_{i,k}), \Lambda_{t,k}, \Theta)}{\sum_{x \in \{A,C,G,T\}} \mathbb{P}(I_{1,k} | I_{1-1,k}, R_t^x(S_{i,k}), \Lambda_{t,k}, \Theta)}
\]

where \( \mathbb{P}(I_{1,k} | I_{1-1,k}, R_t^x(S_{i,k}), \Lambda_{t,k}, \Theta) \) 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

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

with \( P(b) \) 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 II (GA-II) platform, while the data from UC Davis were obtained from a 76-cycle run on the Genome Analyzer I (GA-I) 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.
much, if at all. We conclude that the total number
in Alta-Cyclic. Further, BayesCall suffers less from the “anomalous T” effect
the corresponding entries for the other methods, except for the (T,T) entry
less “confused;” that is, the diagonal entries for BayesCall are higher than

BayesCall

True A 0.99548 0.00179 0.99384 0.00176
True C 0.00500 0.98908 0.00132 0.00460
True G 0.00349 0.00090 0.99288 0.00285
True T 0.00115 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 Σ 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 $d_t$, $\alpha_t$, $\sigma_t^2$, $X_t$, $\Sigma_t$ 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 Equation 15. 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 $\alpha_t$ captures extra residual effects in addition to phasing and prephasing that propagate from one cycle to the next. Our estimates of $\alpha_t$ for the 76-cycle data are shown in Figure 3B. This plot illustrates that the extra residual effects captured by $\alpha_t$ 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 $I_{t,k}$ 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 $\mu_{t,k}$ of the distribution for $I_{t,k}$ can be decomposed into $\Lambda_t X_t S_t Q_t^{T}$ and $\alpha_t (1-d_t) I_{t-1,k}$, 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
BayesCall: A model-based base-calling algorithm

While every off-diagonal entry \((x,y)\) (i.e., the percentage of erroneous calls) in \(C_{\text{BayesCall}}\) is smaller than that in \(C_{\text{Bustard}}\). Also, BayesCall is less "confused" than Alta-Cyclic, in that the diagonal entries in \(C_{\text{BayesCall}}\) are higher than the corresponding entries in \(C_{\text{Alta-Cyclic}}\) 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 \(C_{\text{Bustard}}\) 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 \(C_{\text{BayesCall}}\) are significantly smaller than the corresponding entries in \(C_{\text{Bustard}}\) or \(C_{\text{Alta-Cyclic}}\).

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 \(a_t\). To examine the advantage of modeling residual effects, we considered the following three cases of BayesCall: (1) No residual effect (i.e., impose \(a_t = 0\) for all \(t\)). (2) Constant residual effect (i.e., assume \(a_t\) is equal to some constant \(a\) for all \(t\)). (3) Cycle-dependent residual effect (i.e., \(a_t\) 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 \(d_t, a_t^2, X_t, a_t,\) and \(\Sigma_t\) to be cycle-dependent.

Results from this study are shown in Table 4. Even without modeling extra residual effects (i.e., with \(a_t = 0\) 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 \(d_t, a_t^2, X_t, a_t,\) and \(\Sigma_t\). BayesCall’s improvement over Bustard increased slightly when a model with a constant residual effect was used. When the full model with cycle-dependent \(a_t\) 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 \(D(\epsilon)\) at error tolerance \(\epsilon\) as follows: Sort the bases

![Figure 7](image-url)  
**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.

<table>
<thead>
<tr>
<th>Method</th>
<th>By-base error rate</th>
<th>By-read error rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bustard</td>
<td>0.01737</td>
<td>0.44431</td>
</tr>
<tr>
<td>BayesCall ((a_t = 0) for all (t)))</td>
<td>0.01076</td>
<td>0.31328</td>
</tr>
<tr>
<td>BayesCall ((a_t = \text{some } a) for all (t)))</td>
<td>0.00965</td>
<td>0.28748</td>
</tr>
<tr>
<td>BayesCall (cycle-dependent (a_t))</td>
<td>0.00783</td>
<td>0.24398</td>
</tr>
</tbody>
</table>

In BayesCall, extra residual effects are captured by the parameter \(a_t\). We used cycle-dependent parameters \(d_t, a_t^2, X_t,\) and \(\Sigma_t\) in all three cases of BayesCall. Even without modeling extra residual effects (i.e., \(a_t = 0\) for all \(t\)), BayesCall achieved an improvement of 38% over Bustard in the by-base error rate. When the full model with cycle-dependent \(a_t\) was used, the improvement over Bustard increased to 55%.
According to their quality scores, from the highest to the lowest. Then, go down that sorted list until the error rate surpasses $\varepsilon$. 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 $\varepsilon$ 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 $d_\alpha$ also decreased in magnitude and in the amount of fluctuation. However, we still observed a marked residual effect (parametrized by $\alpha_\delta$), 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 base-calls.

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

<table>
<thead>
<tr>
<th>Platform</th>
<th>Phasing rate $p$</th>
<th>Prephasing rate $q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA-I</td>
<td>$6.8 \times 10^{-4}$</td>
<td>$3.4 \times 10^{-3}$</td>
</tr>
<tr>
<td>GA-II</td>
<td>$3.0 \times 10^{-8}$</td>
<td>$3.3 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

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

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

Supplemental Material
http://genome.cshlp.org/content/suppl/2009/09/02/gr.095299.109.DC1

References
This article cites 9 articles, 3 of which can be accessed free at:
http://genome.cshlp.org/content/19/10/1884.full.html#ref-list-1

License

Email Alerting Service
Receive free email alerts when new articles cite this article - sign up in the box at the top right corner of the article or click here.

To subscribe to Genome Research go to:
http://genome.cshlp.org/subscriptions

Copyright © 2009 by Cold Spring Harbor Laboratory Press