

# SUPPORTING INFORMATION

## bcSeq : An R Package for Fast Sequence Mapping in High-Throughput shRNA and CRISPR Screens

Jiaxing Lin<sup>1</sup>, Jeremy Gresham<sup>2</sup>, Tongrong Wang<sup>1</sup>, So Young Kim<sup>3</sup>, James Alvarez<sup>4</sup>, Jeffrey S. Damrauer<sup>5</sup>, Scott Floyd<sup>2,6</sup>, Joshua Granek<sup>1,7</sup>, Andrew Allen<sup>1,2,7</sup>, Cliburn Chan<sup>1,2,7</sup>, Jichun Xie<sup>1,2,7</sup>, Kouros Owzar <sup>\*1,2,7</sup>

<sup>1</sup>Biostatistics and Bioinformatics, Duke University Medical Center, Durham, 27710, USA

<sup>2</sup>Duke Cancer Institute, Duke University Medical Center, Durham, 27710, USA

<sup>3</sup>Molecular Genetics and Microbiology, Duke University Medical Center, Durham, 27710, USA

<sup>4</sup>Pharmacology, Duke University Medical Center, Durham, 27710, USA

<sup>5</sup>Lineberger Comprehensive Cancer Center, University of North Carolina at Chapel Hill, Chapel Hill, 27599, USA

<sup>6</sup>Radiation Oncology, Duke University Medical Center, Durham, 27710, USA

<sup>7</sup>Duke Center For Statistical Genetics and Genomics, Duke University Medical Center, Durham, 27710, USA

## Contents

<b>1 Overview</b>	<b>2</b>
<b>2 Analysis Tools for High-Throughput Barcode Screens</b>	<b>2</b>
<b>3 Trie Data Structure</b>	<b>2</b>
<b>4 Alignment Probability and Read Classifier</b>	<b>4</b>
<b>5 Simulation Study</b>	<b>6</b>
<b>6 Performance Assessment</b>	<b>7</b>
<b>7 bcSeq Benchmark Analysis</b>	<b>9</b>
<b>8 Measurement Error Model for Statistical Inference</b>	<b>10</b>
<b>9 Data Availability</b>	<b>11</b>
<b>10 Code Availability</b>	<b>11</b>

---

\*Corresponding author

# 1 Overview

High-throughput sequencing screening assays using short hairpin RNA interference (shRNA) libraries and single-guide RNA CRISPR/Cas9 knockout (sgRNA) libraries are powerful tools that are used to study gene function in multiple diseases [1, 2, 3]. Mapping each *observed* read in a sequenced library back to its *originating* barcode from a reference library is a key step in the analysis. This mapping step poses computational and statistical challenges. One approach is to repurpose existing tools for mapping short reads to large genomes (*e.g.*, bowtie [4], bwa [5], and STAR [6]). This approach induces large overhead due to library indexing and determination of starting position for the alignment of each read. Other mapping tools designed for mapping short reads to short barcodes (*e.g.*, shALIGN [7], barcas [8], edgeR [9], and MAGeCK [10]) lack the capability to handle ambiguous mappings.

We have developed `bcSeq`, an open-source R [11] extension package for fast mapping of reads from high-throughput shRNA and sgRNA sequencing assays with multi-threading support. The package uses a Trie [12] data structure implemented in C++ and resolves ambiguous mappings on the basis of a statistical model.

This supporting information (SI) document provides additional technical details and results for “An R Package for Fast Sequence Mapping in High-Throughput shRNA and CRISPR Screens” by Lin *et al.*. This document provides an overview and comparison to other packages and tools available for mapping for barcode screens, and provides information about code and data availability for reproducing the simulation study and for replicating the benchmarking study referenced in the paper.

## 2 Analysis Tools for High-Throughput Barcode Screens

In Table S1, we provide a list of published tools and software packages for read mapping in barcode sequencing screens. For each tool, we provide a citation for the paper and the theoretical computational cost of the alignment. We also provide details about the programming language environment and whether the package supports multithreading. It is noted that among the tools considered, `bcSeq` is the only tool that tolerates errors, resolves ambiguities, has a cost of  $O(n)$  and supports multithreading.

## 3 Trie Data Structure

A Trie is a tree-like data structure commonly adapted in string searching algorithms. They feature fixed-time searching complexity with respect to the size of the library of strings represented in the Trie [12], and combine common prefixes to reduce memory usage. A Trie data structure is suitable for performing short DNA or shRNAs sequence mapping [17]. A schematic of a Trie data structure representing a library of DNA barcodes is shown in Figure S1. Pseudocode for adding a new barcode to the Trie data structure is provided by Algorithm 1. Mapping a read to the barcodes based on Trie data structure is a “down Trie” comparing process and pseudocode is provided by Algorithm 2. The comparison is usually implemented in a recursive way [8]. The algorithm for exact mapping is the same as Algorithm 1 without adding the sequence. A schematic

Tool	Error Tol	Resolve Ambiguity	Cost	Environment	MT
bcSeq	Yes	Yes	$O(n)$	R/C++	Yes
barcas[8]	Yes	No	$O(n)$	Java	Yes
MAGeCK[10]	No	No	$O(n)$	python	No
shALIGN[13]	Yes	No	$O(nL)^*$	perl	Yes
BiNGS!LS-seq[14]	Yes	No	$O(nL)^*$	mixed	Yes
PinAPL-Py[15]	Yes	No	$O(nL)^*$	web service	N/A**
edgeR[9]	Yes	No	$O(nKL)^{***}$	R	No

Table S1: This table provides details for a list of published tools and software packages for analysis of barcode sequencing screens. Error Tol: Indicates if the tool allows errors in the alignment; Resolve Ambiguity: Indicates if the tool resolves ambiguous mappings, or provides methods to detect the barcode that a read should be mapped to from candidate barcodes (Please refer to formal definition of ambiguous mappings in Section 4); Cost: Indicates the tool’s theoretical computational complexity for alignment; Environment: Indicates the programming and development environment for the tool; MT: Indicates if the tool supports multithreading for the mapping \*Theoretical computational cost for the BWT [16] algorithm based on the number of reads  $n$  and the average read length  $L$ . \*\*PinAPL-Py is a server-based service that appears to use Bowtie2 internally, but there is no mechanism given for a user to control multithreading in the mapping. \*\*\*edgeR compares each read to the library sequences one by one and stops at the first matched barcode. The computational cost is proportion to the number of reads  $n$ , average read length  $L$ , and the number of barcodes in the library  $K$ . For the worst case, the computational cost can be written as  $O(nKL)$  for edgeR.

of mapping a read to a barcode library is also shown in Figure S1. bcSeq tolerates mismatches as well as internal insertion and deletion.

---

**Algorithm 1** Pseudocode for adding a sequence to a Trie data structure.

---

```

1: Node Current = Root
2: for i=1 to seq.length() do
3:   Node Child = Current.findChild(seq[i])
4:   if Current != NULL then
5:     Current = Child
6:   else
7:     Node tmp = new Node(seq[i])
8:     Current.addChild(tmp)
9:     Current = tmp
10:  end if
11: end for
12: Current.setLeaf()

```

---



$j \in \{1, \dots, L\}$ . The sequencing reads consist of  $n$  reads and are assumed to have been randomly sampled from the barcodes in this library. For a given read,  $\tilde{R}$  denotes the sequence of the originating barcode and  $R$  denotes the sequence of the observed read.  $Z \in \{1, \dots, K\}$  denotes the index of the originating barcode  $\tilde{R}$ . In other words,  $Z = k$  if and only if  $\tilde{R} = \tilde{\mathbf{r}}^k$  where  $k \in \{1, \dots, K\}$ .

The sequence of read  $i \in \{1, \dots, n\}$  is denoted by  $R_i = [R_{i,1}, \dots, R_{i,L}]$  and its originating sequence is denoted by  $\tilde{R}_i = [\tilde{R}_{i,1}, \dots, \tilde{R}_{i,L}]$ , where  $\tilde{R}_{i,j}, R_{i,j} \in \{A, T, G, C\}$ . The Phred scores for read  $i$  are denoted by  $Q_i = [Q_{i,1}, \dots, Q_{i,L}]$ . The index corresponding to the originating read  $\tilde{R}_i$  is denoted by  $Z_i$ .

For a given sequencing read, let  $d_k \in \{0, 1, 2, \dots\}$  denote the distance between the read and barcode  $k$ . Furthermore, for each  $d \in \{0, 1, 2, \dots\}$ , let  $N[d] = \{k \in \{1, \dots, K\} : d_k = d\}$ . When allowing up to  $n$  mismatches we consider the mapping of a read to be *ambiguous* if  $m = \min\{d_1, \dots, d_K\}$ ,  $0 < m \leq n$ , and  $|N[m]| > 1$  (*i.e.*, there is no perfect match and there are at least two barcodes that are within the minimum distance  $\min\{d_1, \dots, d_K\}$  of the read).

Let  $\mathbb{P}(\tilde{\mathbf{r}}^k | \mathbf{r}) := \mathbb{P}(\tilde{R} = \tilde{\mathbf{r}}^k | R = \mathbf{r})$  denote the probability that a read originated from barcode  $k$ , with the corresponding sequence  $\tilde{\mathbf{r}}^k$ , given that the sequence of the observed read is  $\mathbf{r} = [r_1, \dots, r_L]$ . The mapped barcode index on the basis of the observed read will be obtained, on the basis of a Bayes' classifier as  $\hat{Z} = \arg \max \mathbb{P}(\tilde{\mathbf{r}}^k | \mathbf{r})$  over  $k \in \{1, \dots, K\}$ .

The current implementation of `bcSeq` models the joint probability distribution of the originating reads conditional on the corresponding observed reads,  $\mathbb{P}(\tilde{\mathbf{r}}^k | \mathbf{r})$ , under the assumption of conditional independence, as

$$\begin{aligned} \mathbb{P}(\tilde{\mathbf{r}}^k | \mathbf{r}) &= \mathbb{P}(\tilde{R} = \tilde{\mathbf{r}}^k | R = \mathbf{r}) \\ &= \mathbb{P}(\tilde{R} = [\tilde{r}_1, \dots, \tilde{r}_L] | R = [r_1, \dots, r_L]) \\ &= \prod_{i=1}^L \mathbb{P}(\tilde{R}_i = \tilde{r}_i | R = [r_1, \dots, r_L]) \\ &= \prod_{i=1}^L \mathbb{P}(\tilde{R}_i = \tilde{r}_i | R_i = r_i). \end{aligned}$$

The marginal conditional probability is modeled as

$$\mathbb{P}(\tilde{R}_j = \tilde{r}_j | R_j = r_j) = \begin{cases} 1 - \epsilon_j, & \tilde{r}_j = r_j \\ \frac{\epsilon_j}{3}, & \tilde{r}_j \neq r_j, \end{cases}$$

where  $\epsilon_j = 10^{-q_j/10}$  is the base-calling error probability corresponding to the observed Phred score  $q_j$ .

This model is similar to that proposed by Li et al. [18] (Supplementary PDF Section 1.1) in that it assumes conditional independence and that conditional on the observed base  $R_j = r_j$ , the probability that the true base  $\tilde{R}_j$  is a nucleotide other than  $r_j$  is  $\frac{\epsilon_j}{3}$ . The difference is that our model conditions on the observed read whereas the model used by Heng *et al.* conditions on the

originating barcode. The two approaches are of course identical when  $\frac{\mathbb{P}(\tilde{R}_j = \tilde{r}_j)}{\mathbb{P}(R_j = r_j)} = 1$  as

$$\mathbb{P}(\tilde{R}_j = \tilde{r}_j | R_j = r_j) = \frac{1}{\mathbb{P}(R_j = r_j)} \mathbb{P}(\tilde{R}_j = \tilde{r}_j, R_j = r_j) = \frac{\mathbb{P}(\tilde{R}_j = \tilde{r}_j)}{\mathbb{P}(R_j = r_j)} \mathbb{P}(R_j = r_j | \tilde{R}_j = \tilde{r}_j).$$

The package is designed to facilitate the specification of user defined models for  $\mathbb{P}(\tilde{\mathbf{r}}^k | \mathbf{r})$ . To this end, instructions are provided in the package documentation.

## 5 Simulation Study

For the simulation study, the reference barcode library is assumed to consist of  $K$  unique sequences, each of length  $L$ . For barcode  $k \in \{1, \dots, K\}$ , the sequence is denoted as  $\tilde{\mathbf{r}}^k = [\tilde{r}_1^k, \dots, \tilde{r}_L^k]$  where  $\tilde{r}_j^k \in \{A, T, G, C\}$  for  $j \in \{1, \dots, L\}$ . For each simulation replicate,  $n$  sequencing reads are randomly sampled from the barcodes in this library. We denote these by  $\tilde{R}_1, \dots, \tilde{R}_n$  where  $\tilde{R}_i = [\tilde{R}_{i,1}, \dots, \tilde{R}_{i,L}]$ , for  $i \in \{1, \dots, n\}$ , and  $\tilde{R}_{i,j} \in \{A, T, G, C\}$  for  $j \in \{1, \dots, L\}$ .  $Z_1, \dots, Z_n$  denote the corresponding barcode indices. In other words,  $Z_i = k$  if and only if  $\tilde{R}_i = [\tilde{r}_1^k, \dots, \tilde{r}_L^k]$ .

For barcode  $k$  its prevalence  $\pi_k$  is generated from a sorted modified uniform distribution similar to Claessen et al. [19]. Let  $E = \frac{C}{K}$ . First,  $\tilde{P}_1, \dots, \tilde{P}_K$  are drawn from a uniform distribution  $\mathbb{U}(m, M)$ , where  $m = \frac{1}{K} - E$  and  $M = \frac{1}{K} + E$ .  $C$  is an arbitrary user specified scalar in the interval  $(0, 1)$ .  $\tilde{P}_1, \dots, \tilde{P}_K$  are then modified subject to the constraint  $\sum_{k=1}^K \tilde{P}_k = 1$  as follows. We set

$$\begin{aligned} P_1 &= \tilde{P}_1 \\ E_1 &= P_1 - \frac{1}{K}, \end{aligned}$$

and for  $i \in \{2, \dots, K-1\}$

$$P_i = \begin{cases} \tilde{P}_i & |\tilde{P}_i - \frac{1}{K} + \sum_{j=0}^{i-1} E_j| < E \\ \frac{1}{K} - (\tilde{P}_i - \frac{1}{K}) & \text{o.w.}, \end{cases} \quad (1)$$

and

$$E_i = P_i - \frac{1}{K},$$

and finally

$$P_K = 1 - \sum_{j=1}^{K-1} P_j$$

Finally, we set  $\pi_k = P_{(k)}$ , where  $P_{(k)}$  denotes the  $k$ -th order statistic among  $P_1, \dots, P_K$ .

For a simulated read  $\tilde{R}_i$ ,  $i \in \{1, \dots, n\}$ , the corresponding  $L$  Phred scores,  $Q_{1,i}, \dots, Q_{L,i}$ , are drawn from a discrete uniform distribution on the set  $\{a, \dots, b\}$ , where  $a < b$  are positive integers.

Let  $F_{r,\epsilon}$  denote a discrete distribution over the set  $\{A, T, G, C\}$  that assigns probability  $1 - \epsilon$  to  $r \in \{A, T, G, C\}$  and probability  $\frac{\epsilon}{3}$  to each element of  $\{A, T, G, C\} - \{r\}$ . Conditional on the Phred scores  $[Q_{1,i} = q_{1,i}, \dots, Q_{L,i} = q_{L,i}]$ , the observed read  $R_i = [R_{1,i}, \dots, R_{L,i}]$  corresponding

to the originating read  $\tilde{R}_i = [\tilde{R}_{1,i}, \dots, \tilde{R}_{L,i}]$  is obtained by drawing  $R_{j,i}$  from  $F_{\tilde{R}_{j,i}, \epsilon_{j,i}}$  where  $\epsilon_{i,j} = 10^{-q_{i,j}/10}$ .

For the simulation study presented in the paper we consider a reference barcode library consisting of  $K = 500$  barcodes each of length  $L = 18$ . The minimum pairwise edit distance among these 500 barcodes is 2. The relative prevalences for the barcodes are determined based on Eq. 1 with  $C = 0.75$ . For simulation replicate  $b \in \{1, \dots, B = 1,000\}$ , we draw  $n = 10^6$  reads from this library according to these relative prevalences. The simulated Phred scores are sampled randomly from a real sequencing library generated by Koike-Yusa et al. [20] (see Data Availability section for obtaining the file). For each simulated read, a sequence of Phred scores is drawn at random from the records in the FASTQ file. For each record in this file, the mean of the the Phred scores was used to quantify the quality of the record. The records were binned based on their quality, *i.e.*, read average Phred scores. In the simulation, the Phred scores were sampled from records in each of these bins for different read qualities.

## 6 Performance Assessment

This section provides additional details for the calculations and results for specificity, total number of reads mapped, and proportion of reads mapped in the simulation study considered in the manuscript.

For simulation replicate  $b \in \{1, \dots, B\}$ , let  $P_{k,b}$  denote the number of barcodes, among  $n$ , *originating* from barcode  $k \in \{1, \dots, K\}$  (the positives), and its estimation as  $\hat{P}_{k,b}$ . Correspondingly, let  $N_{k,b}$  denote the number of reads *not originating* from barcode  $k \in \{1, \dots, K\}$ , and its estimation as  $\hat{N}_{k,b}$ . We express the positives in terms of true-positives and false-negatives as  $P_{k,b} = \text{TP}_{k,b} + \text{FN}_{k,b}$ , and the negatives in terms of the true-negatives and false-positives as  $N_{k,b} = \text{TN}_{k,b} + \text{FP}_{k,b}$ , where

$$\text{TP}_{k,b} = \sum_{i=1}^n \mathbb{I}[Z_{i,b} = k, \hat{Z}_{i,b} = k],$$

$$\text{FP}_{k,b} = \sum_{i=1}^n \mathbb{I}[Z_{i,b} \neq k, \hat{Z}_{i,b} = k].$$

$$\text{FN}_{k,b} = \sum_{i=1}^n \mathbb{I}[Z_{i,b} = k, \hat{Z}_{i,b} \neq k],$$

and

$$\text{TN}_{k,b} = \sum_{i=1}^n \mathbb{I}[Z_{i,b} \neq k, \hat{Z}_{i,b} \neq k],$$

Let  $B_k = \sum_{b=1}^B \mathbb{I}[\hat{N}_{k,b} > 0]$ . We define the empirical specificity for barcode  $k$ , averaged over the  $B$  simulation replicates as

$$\eta_k = \frac{1}{B_k} \sum_{b=1}^B \frac{\text{TN}_{k,b}}{\text{TN}_{k,b} + \text{FP}_{k,b}},$$

and the total number of mapped reads for simulation replicate  $b$  is

$$\hat{N}_b = \sum_{k=1}^K \hat{P}_{k,b}$$

We also defined the proportion of reads mapped for simulation replicate  $b$  as

$$\hat{R}_b = \frac{\sum_{k=1}^K \hat{P}_{k,b}}{\sum_{k=1}^K P_{k,b}}$$

The proportion of reads mapped and the specificity (excluding the unmapped reads) are reported on the Figure 1 of the manuscript. The total number of mapped reads  $\hat{N}_b$  are shown in Figure S2. We observe that our approach consistently maps more reads to barcodes in all Phred score scenarios. Furthermore, we observe that the number of mapped reads for our method improves as the quality of the reads decreases compared with perfect match.

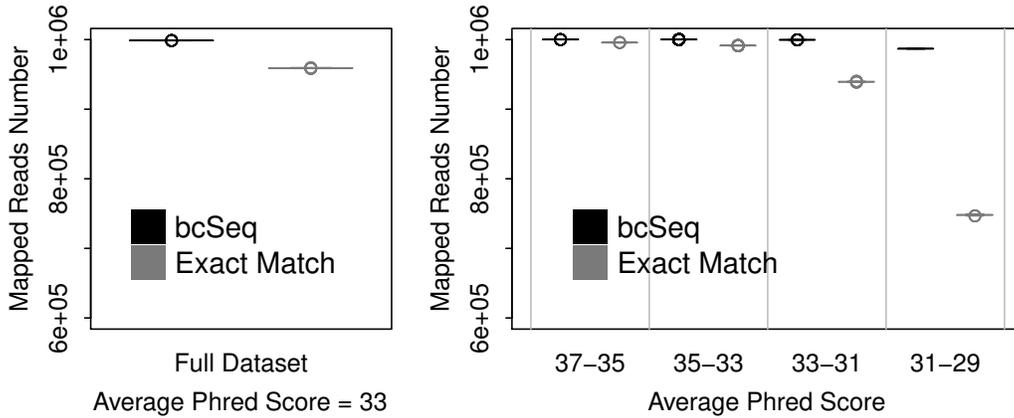


Figure S2: These figures illustrate comparisons of total number of mapped reads to library barcodes. Each example is based on 500 barcodes and  $10^6$  reads with 1000 simulation replicates. The left panel shows that **bcSeq** has a higher number of mapped reads than perfect match when run on barcode reads simulated using Phred scores sampled from real sequencing data. The right panel elucidates the role of sequencing error in **bcSeq**'s performance. Each read in the real sequencing data was binned based on its average Phred score, and **bcSeq** and perfect match were run on reads simulated using Phred scores sampled from each of these bins. perfect match's number of mapped reads degrades much more rapidly than **bcSeq**'s as average Phred score decreases.

Next, we compare the performance of **bcSeq** to three published packages: **barcas** [8] (version 1.0), **edgeR** [9] (version 3.5) and **MAGeCK** [10] (version 0.5.6). **bcSeq** provides the user with the option to report the barcodes to which each read in the sequencing library was mapped, in addition to reporting the number of reads mapped to each barcode. As the other three packages seemingly

only offer the latter option, one cannot directly calculate sensitivity, specificity and MCC as in the case of the previous simulation study based on our package. For this present comparison, we evaluate the performance of each of the four methods on the basis of  $L^1$ -norm:

$$D(\hat{\boldsymbol{\pi}}|\boldsymbol{\pi}) = \sum_{k=1}^{K+1} |\hat{\pi}_k - \pi_k| = |\hat{\pi}_{K+1} - \pi_{K+1}| + \sum_{k=1}^K |\hat{\pi}_k - \pi_k| = \hat{\pi}_{K+1} + \sum_{k=1}^K |\hat{\pi}_k - \pi_k|, \quad (2)$$

where  $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K, \pi_{K+1})^T$  is the vector of barcode prevalences in the reference library and  $\hat{\boldsymbol{\pi}} = (\hat{\pi}_1, \dots, \hat{\pi}_K, \hat{\pi}_{K+1})^T$  is the vector of observed proportion of reads mapped to each barcode. Here  $\hat{\pi}_{K+1}$  denotes the proportion of unmapped reads. Excluding unmapped reads, that is  $\hat{\pi}_{K+1} = 0$ , Eq. 2 becomes:

$$D(\hat{\boldsymbol{\pi}}|\boldsymbol{\pi}) = \sum_{k=1}^K |\hat{\pi}_k - \pi_k| \quad (3)$$

For this comparison study, we employ the design used in the simulation study. The distance between each read and a candidate barcode is quantified using the Hamming distance. The performance is assessed based on tolerating zero, one, two or three mismatches without considering unmapped reads (Eq. 3). As **MAGeCK** only accommodates perfect matching, the evaluation of its performance is limited to the case with no mismatches. The **processAmplicons** function from **edgeR** is used in this study.

The results for this comparison study are shown in Table S2. We note for the scenarios considered here, **bcSeq** performs uniformly better compared to the other three packages. For low quality read data, **bcSeq** performance can be improved by allowing more mismatches. A similar trend is not observed for **edgeR** and **barcas** for which allowing more mismatches may reduce performance. **edgeR** performs mapping by comparing each read to the library barcodes one by one and returns the first matched barcode. This, when tolerating mismatches, is not exhaustive and can potentially miss the correct mapping. **barcas** discards ambiguous mappings. If a read originating from barcode  $k$  is discarded due to ambiguity, the norm  $D$  is increased due to utilize less data. This will also lead to potentially biased estimate of  $\hat{\pi}_k$ . The scripts for reproducing the results in Table S2 are included as Supplementary Material and detailed in the Code Availability section.

## 7 bcSeq Benchmark Analysis

To assess the computational performance of **bcSeq**, we apply our package to map a sequenced library consisting of approximately  $n = 10$  million reads to a reference library consisting of  $K = 87,897$  barcodes. The sequencing library is obtained from the tutorial website of **MAGeCK** and the barcode library data is provided by Koike-Yusa et al. [20]. The length of each sequencing read and reference barcode is  $L = 19$ . A tabulation of the pairwise Hamming distances for the reference barcodes is provided in Table S3. Details about how to obtain these files are provided in the Data Availability section of this document.

We assess the performance of our package based on  $m = 1, 2, 4, 6$  and 8 cores. For each core count, we run the mapping process in triplicate. The median completion times are 60, 31, 16, 12, and 9 minutes based on  $m = 1, 2, 4, 6$  and 8 cores respectively. The individual completion times

Phred Score	MM	bcSeq	edgeR	barcas	MAGeCK
high	0	0.0014	0.0014	0.0014	0.0014
	1	0.00041	0.00068	0.00060	N/A
	2	0.00037	0.00069	0.0032	N/A
	3	0.00037	0.00069	0.0071	N/A
low	0	0.0071	0.0071	0.0071	0.0071
	1	0.0038	0.0096	0.0089	N/A
	2	0.0027	0.012	0.045	N/A
	3	0.0021	0.013	0.12	N/A

Table S2: Comparison of the mapping performance based on an  $L^1$ -norm Eq. 3 (excluding unmapped reads) for four packages: **bcSeq**, **MAGeCK**, **edgeR** and **barcas**. For each example, the number of barcodes is  $K = 500$  and the number of reads is  $n = 10^6$ . MM denotes the maximum number of allowed mismatches. As the **MAGeCK** package only supports exact matching, its performance is not evaluated for  $MM > 0$ .

are shown in Figure S3. This benchmarking study was conducted using version 1.0.0 of **bcSeq** under R version 3.4.1 and Bioconductor release version 3.5 on an AMD FX 8350 desktop CPU running the Debian Jessie (8.9) AMD64 GNU/Linux operating system.

## 8 Measurement Error Model for Statistical Inference

In the context of statistical inference, the goal is to establish if the event that a read originates from a given barcode is associated with an outcome or experimental condition, which we will label  $Y$ . For a given read, let  $Z$  denote the index of the originating barcode and let  $\hat{Z}$  denote the index of the barcode to which it was mapped. Due to sequencing error,  $Z$  is not observable. The goal is to establish if  $Y$  is associated with the latent event  $Z_k = \mathbb{I}[Z = k]$ ; it is not the goal to establish a relationship between  $Y$  and the observed event  $\hat{Z}_k = \mathbb{I}[\hat{Z} = k]$ .

In the presence of sequencing error, what is observed for experimental unit  $i$  is  $(Y_i, R_{i1}, \dots, R_{in})$  and not  $(Y_i, \tilde{R}_{i1}, \dots, \tilde{R}_{in})$ . For notational simplicity, we assume that  $Y_i$  follows a discrete distribution. The probability mass function of  $Y_i$  given the observed read,  $R_{ij} = r_{ij}$ , is

$$\begin{aligned}
\mathbb{P}(Y_i = y_i | R_{ij} = r_{ij}) &= \sum_{\tilde{r} \in \mathcal{R}} \mathbb{P}(Y_i = y_i, \tilde{R}_{ij} = \tilde{r} | R_{ij} = r_{ij}) \\
&= \sum_{\tilde{r} \in \mathcal{R}} \mathbb{P}(Y_i = y_i | \tilde{R}_{ij} = \tilde{r}) \mathbb{P}(\tilde{R}_{ij} = \tilde{r} | R_{ij} = r_{ij}) \\
&= \sum_{\tilde{r} \in \mathcal{R}} \mathbb{P}(Y_i = y_i | \tilde{R}_{ij} = \tilde{r}) \omega_{ij}(\tilde{r}),
\end{aligned}$$

for  $y_i$  in the support of the distribution of  $Y_i$ . Here,  $\mathcal{R}$  denotes the set of all unique reference barcode sequences.

Hamming distance	count number
0	460
1	266
2	403
3	1243
4	9067
5	65817
6	407492
7	2063702
8	8614029
9	29721309
10	84854044
11	200092303
12	387796184
13	611808873
14	773786111
15	766469587
16	573283565
17	304701088
18	102711235
19	16510578

Table S3: A tabulation of the pairwise Hamming distances for the the  $K = 87,897$  reference barcodes provided by Koike-Yusa et al. [20].

## 9 Data Availability

The sequencing library used in the benchmarking analysis can be downloaded from the Sequence Read Archive <sup>1</sup>. The reference barcode library can be downloaded from the Supplementary Information section of Koike-Yusa et al. [20] <sup>2</sup>. The md5sum hash keys for these two files are:

```
md5sum ERR376998.fastq.gz
7b552f8efd6fe90fc6bc34879c617040 ERR376998.fastq.gz
md5sum nbt.2800-S7.xlsx
545d1ed872003a39e4abdb78ac1c89d8 nbt.2800-S7.xlsx
```

See Code Availability section for information to pre-process these files.

## 10 Code Availability

The source code for `bcSeq` is written in C++ and ported to R by Rcpp [21] and released under a GPL-v3 license. The source code and binary versions of the package are disseminated through the Bioconductor project [22] webpage (<http://bioconductor.org/packages/bcSeq/>).

<sup>1</sup><ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR376/ERR376998/ERR376998.fastq.gz>

<sup>2</sup><https://www.nature.com/nbt/journal/v32/n3/extref/nbt.2800-S7.xlsx>

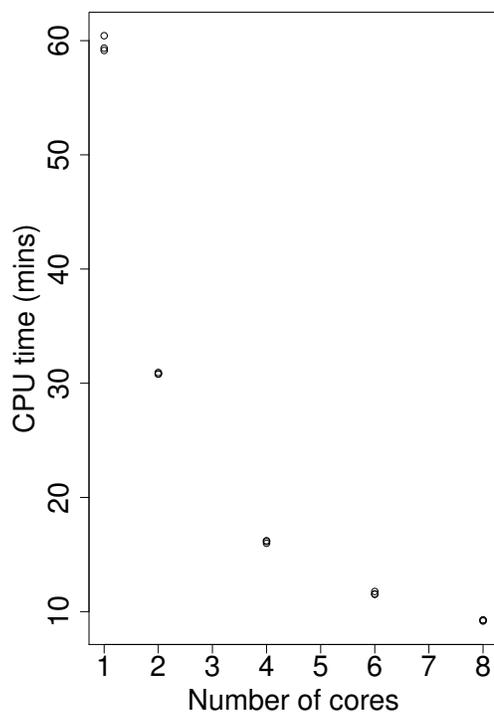


Figure S3: Plot of computational time (in minutes) used by `bcSeq` to perform mapping for  $n = 10$  million reads from MAGeCK website and a reference library consisting of  $K = 87,897$  barcodes library from Koike-Yusa et al. [20] using  $m = 1, 2, 4, 6$  and 8 cores. The timings for each core count are run in triplicate.

As supplementary material, we provide a bash script (`bcSeq-benchmark.sh`) for downloading, and for preprocessing and transforming these two files from the benchmarking study to required `.fastq` and `.fasta` formats.

The R scripts for executing the simulation study (`bcSeq-simulation.R`), benchmarking study (`bcSeq-benchmark.R`) and package performance comparison (`bcSeq-pkgCompare.R`) along with a Makefile are also provided as supplementary material.

The benchmarking study is used as an example in the package vignette. The benchmarking, simulation, and package performance comparison analyses can be executed from a bash shell as follows:

```
make benchmark
make simulation
make pkgCompare
```

These analyses are tested using R (version 3.4.1) gcc (version 6.3.0) and java (version 1.8.0). The package comparison analysis section provides information for `bcSeq` and `edgeR` package versions. See the Data Availability section in this document for obtaining the requisite data source files for the benchmarking study.

## References

- [1] P. D. Hsu, E. S. Lander, and F. Zhang. Development and applications of CRISPR-Cas9 for genome engineering. *Cell*, 157:1262, 2014.
- [2] A. S. L. Wong, G. C. G. Choi, C. H. Cui, G. Pregonig, P. Milani, M. Adam, S. D. Perli, S. W. Kazer, A. Gaillard, M. Hermann, A. K. Shalek, E. Fraenkel, and T. K. Lu. Multiplexed barcoded CRISPR-Cas9 screening enabled by CombiGEM. *Proc. Natl. Acad. Sci. U.S.A.*, 113:2544, 2016.
- [3] H. C. Bhang, D. A. Ruddy, K. R. Viveksagar, C. X. Justina, R. Zhao, M. M. Hims, A. P. Singh, I. Kao, D. Rakiec, P. Shaw, M. Balak, A. Raza, E. Ackley, N. Keen, M. R. Schlabach, M. Palmer, R. J. Leary, D. Y. Chiang, W. R. Sellers, F. Michor, V. G. Cooke, J. M. Korn, and F. Stegmeier. Studying clonal dynamics in response to cancer therapy using high-complexity barcoding. *Nat. Med.*, 21:440, 2015.
- [4] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.*, 10:R25, 2009.
- [5] H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25:1754, 2009.
- [6] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras. STAR: Ultrafast universal RNA-seq aligner. *Bioinformatics*, 29:15, 2013.
- [7] D. Sims, A. M. Mendes-Pereira, J. Frankum, D. Burgess, M. Cerone, C. Lombardelli, C. Mitsopoulos, J. Hakas, N. Murugaesu, C. M. Isacke, K. Fenwick, I. Assiotis, I. Kozarewa, M. Zvelebil, A. Ashworth, and C. J. Lord. High-throughput RNA interference screening using pooled shRNA libraries and next generation sequencing. *Genome Biol.*, 12:R104, 2011.
- [8] J. Mun, D. Kim, K. Hoe, and S. Kim. Genome-wide functional analysis using the barcode sequence alignment and statistical analysis (Barcas) tool. *BMC Bioinformatics*, 17:475, 2016.
- [9] Z. Dai, J. M. Sheridan, L. J. Gearing, D. L. Moore, S. Su, S. Wormald, S. Wilcox, L. O'Connor, R. A. Dickins, M. E. Blewitt, and M. E. Ritchie. edgeR: a versatile tool for the analysis of shRNA-seq and CRISPR-Cas9 genetic screens. *F1000Research*, 3:95, 2014.
- [10] W. Li, H. Xu, T. Xiao, L. Cong, M. I. Love, F. Zhang, R. A. Irizarry, J. S. Liu, M. Brown, and X. S. Liu. MAGeCK enables robust identification of essential genes from genome-scale CRISPR-Cas9 knockout screens. *Genome Biol.*, 15:554, 2014.
- [11] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.
- [12] P. Bieganski, J. Riedl, J.V. Cartis, and E. F. Retzel. Generalized suffix trees for biological sequence data: Applications and implementation. In *1994 Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences*, volume 5, page 35, 1994.

- [13] D. Sims, A. M. Mendes-Pereira, J. Frankum, D. Burgess, M. Cerone, C. Lombardelli, C. Mitsopoulos, J. Hakas, C. M. Murugaesu, N. and Isacke, K. Fenwick, I. Assiotis, I. Kozarewa, M. Zvelebil, A. Ashworth, and C. J. Lord. High-throughput RNA interference screening using pooled shRNA libraries and next generation sequencing. *Genome Biol.*, 12:R104, 2011.
- [14] J. Kim and A. C. Tan. *BiNGS!SL-seq: A Bioinformatics Pipeline for the Analysis and Interpretation of Deep Sequencing Genome-Wide Synthetic Lethal Screen*, page 389. Humana Press, 2012.
- [15] Spahn P. N., Bath T., Weiss R. J., Kim J., Esko J. D., N. E. Lewis, and O. Harismendy. PinAPL-Py: a web-service for the analysis of CRISPR-Cas9 screens. *bioRxiv*, 2017.
- [16] H. Li and R. Durbin. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, 25:1754, 2009.
- [17] H. Li and N. Homer. A survey of sequence alignment algorithms for next-generation sequencing. *Briefings in Bioinformatics*, 11:473, 2010.
- [18] H. Li, J. Ruan, and R. Durbin. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res.*, 18:1851, 2008.
- [19] K. Claessen, J. Duregrard, and M. Pałka. *Generating Constrained Random Data with Uniform Distribution*, page 18. Springer International Publishing, 2014.
- [20] H. Koike-Yusa, Y. Li, E. Tan, M. Velasco-Herrera, and K. Yusa. Genome-wide recessive genetic screening in mammalian cells with a lentiviral CRISPR-guide RNA library. *Nat. Biotech.*, 32:267273, 2014.
- [21] D. Eddelbuettel and R. François. Rcpp: Seamless R and C++ integration. *J. Stat. Softw.*, 40:1, 2011.
- [22] W. Huber, V. J. Carey, R. Gentleman, S. Anders, M. Carlson, B. S. Carvalho, H. C. Bravo, S. Davis, L. Gatto, T. Girke, R. Gottardo, F. Hahne, K. D. Hansen, R. A. Irizarry, M. Lawrence, M. I. Love, J. MacDonald, V. Obenchain, A. K. Oles, H. Pagès, A. Reyes, P. Shannon, G. K. Smyth, D. Tenenbaum, L. Waldron, and M. Morgan. Orchestrating high-throughput genomic analysis with Bioconductor. *Nat. Meth.*, 12:115, 2015.