zoom

next gen seq software

User's Manual

zillions of oligos mapped

Bioinformatics Solutions Inc.

BIOINFORMATICS SOLUTIONS INC

ZOOM User's Manual

---

470 Weber St. N. Suite 204
Waterloo, Ontario, Canada N2L 6J2
Phone 519-885-8288 • Fax 519-885-9075

Please contact BSI for questions
or suggestions for improvement.

Table of Contents

Chapter

1

Introduction

Introduction to ZOOM

ZOOM (Zillions Of Oligos Mapped) is designed to map millions of short reads, produced by next-generation sequencing technology, back to the reference genome, and carry out post-analysis. Based on a newly designed multiple spaced seeds theory, ZOOM guarantees great mapping accuracy with unparalleled speed. Both single and paired-end reads of various lengths from 12bp to 240bp can be handled. Any number of mismatches and one insertion/deletion of various lengths between the read and its target region on the reference sequence are allowed. Uniquely mapped results or best (top N) results for each read will be reported, according to the minimal mismatches and indel length between the read and its target positions. Based on the information from the mapped reads, ZOOM reconstructs a consensus sequence and outputs the coverage and heterozygote frequency of each position, which is useful for ChIP-Seq analysis or SNP identification.

ZOOM supports both Illumina/Solexa and ABI SOLiD instruments. For Illumina/Solexa data, quality scores generated by the sequencer for each of the short sequenced reads can be incorporated to reduce ambiguity of read mapping. For ABI SOLiD data, ZOOM directly aligns a color space read to a base space reference sequence. ZOOM is therefore able to differentiate a true polymorphism on the base space from the sequencing errors on the color space, and automatically corrects sequencing errors during the mapping process. Reads in color space will be decoded into base space, with both sequencing errors on color space and true polymorphisms to their target region on the reference genome marked, respectively.

Terminology and Abbreviations Glossary

ZOOM: Zillions Of Oligos Mapped, a next generation sequencing analysis tool

BSI (Bioinformatics Solutions Inc.): the maker of PEAKS, PatternHunter, RAPTOR, ZOOM and other fine bioinformatics software

Mismatch: A mismatch occurs when the nucleotide base from the read and the reference sequence are different, or when either of the sequences has an '*N*' at that position. If the sequencing qualities are also used, the mismatches occurring at low quality sites (determined by a quality threshold) will be ignored.

Indel: insertion and deletion mutations

Optimal spaced seed: a novel idea proposed first in PatternHunter to enhance both sensitivity and speed of filtering in the pairwise homology search process. Compared to a consecutive seed which requires the query sequence and the target sequence to share a sequence block of same nucleotides, optimal spaced seed requires only designated positions to be the same. The strategy was proven in PatternHunter to enhance sensitivity and speed greatly when compared to BLAST.

Multiple spaced seeds: Multiple spaced seeds, which further enhance the sensitivity, are several spaced seeds optimized simultaneously against a given level of similarity. PatternHunter II using multiple spaced seeds would approach the sensitivity of the Smith-Waterman algorithm while gaining Blastn speed.

SNP: Single Nucleotide Polymorphisms

Quality score: the quality or confidence score of each nucleotide sequenced

Oligos: oligonucleotides, short DNA or RNA sequences

100% sensitivity: the full capacity to find all target regions within user-defined mismatches on the reference sequence for each read

Hamming distance: the number of mismatches between a read and its target region on the reference sequence.

Edit distance: the summation of the number of mismatches and the length of indels

Target region: reference sequence segment where the read is mapped

Reference offset: the leftmost position where a read is mapped onto the reference sequence.

Uniquely mapped read: Each read might be mapped to multiple target regions in the reference sequence. The best mapping results of one read are the ones with smallest edit distance, or for equal edit distance, the shortest indel length (under the consideration that indels are less probable than mutations). If there is only one such best mapping result for the read, this is a uniquely mapped read. Otherwise, if there are multiple such mappings, the read will be considered ambiguously mapped.

Coverage: the number of reads that one segment/area of the reference sequence is sequenced. It also means the number of reads mapped back to one position or one area of the reference sequence.

Paired-end reads: two reads sequenced from both ends of the DNA fragment. The paired-end reads from the same region of the reference sequence are expected to be located on the same chain and separated by a known distance range. The orientation and distance limit help to locate unambiguous reads. They are also helpful in finding insertion/deletion and structural variations.

Single-end reads: reads that were sequenced separately

Base space: reads represented in the alphabet of nucleotides {*A, C, G, T, N*}, such as *ACGTAAA*

Color space: also called di-base alphabet. This is the data format produced by the ABI SOLiD sequencer. Reads are represented as colors, in the way that two adjacent nucleotides are encoded by one color letter, represented as {*0, 1, 2, 3*}. The convert from base space to color space uses the following table:

	A	C	G	T
A	0	1	2	3
C	1	0	3	2
G	2	3	0	1
T	3	2	1	0

Chapter

2

Getting started with ZOOM

What we will need

Package contents

The ZOOM package should contain:

- This manual
- ZOOM binary distribution

System requirements

ZOOM will run on most platforms with the following requirements:

- Equivalent or superior processing power to a Pentium III at 800 MHz
- Unix/Linux operation system
- The memory requirement is proportional to input data size; 8 GB RAM is recommended for aligning 25 million reads once.

Instrumentation

ZOOM will work with both single-end reads and paired-end reads of length ranging from 12bp to 240bp from the following next generation sequencing instruments:

Illumina/Solexa: *_seq.txt and *_prb.txt

Applied Biosystems SOLiD: *.csfasta

Installation

ZOOM is provided as a binary executable file which is usually packed. Use the following Unix/Linux command to unpack it:

```
tar –zxvf zoom.tar.gz
```

Chapter

3

How to Use ZOOM

This chapter of the manual will walk you through most of the basic functionality of ZOOM. This section includes all the parameters used to configure ZOOM to your requirements. First, the workflows of ZOOM for Illumina/Solexa data and ABI SOLiD data will be illustrated. Then, a quick start section will give you a first glimpse of what ZOOM can do. Finally, the format of input and output files of ZOOM will be explained in detail. Usage of command line parameters is shown in this chapter.

Understanding ZOOM

The key strategy adopted by ZOOM is the utilization of multiple spaced seeds to map short reads to the reference sequence with a number of mismatches and still obtain 100% sensitivity. ZOOM indexes the reads using these spaced seeds and scans the reference genome for the potential matches. Therefore, the memory usage is related to the number of reads being processed, while the efficiency is associated with the number of times the reference genome is scanned. Thus, it is recommended that reads from multiple small files (for example, reads from different lanes) are put into a single file to be input into ZOOM. For 8G RAM, 25 million reads are recommended in one run. However, if the file is too large to fit in memory, please divide the data set into several parts and run ZOOM several times, then merge the mapping results. In later release, ZOOM will automatically partition the files (when the GUI version is used, the user can specify a directory of read files to be processed) into several jobs to obtain optimal efficiency according to the RAM limitations.

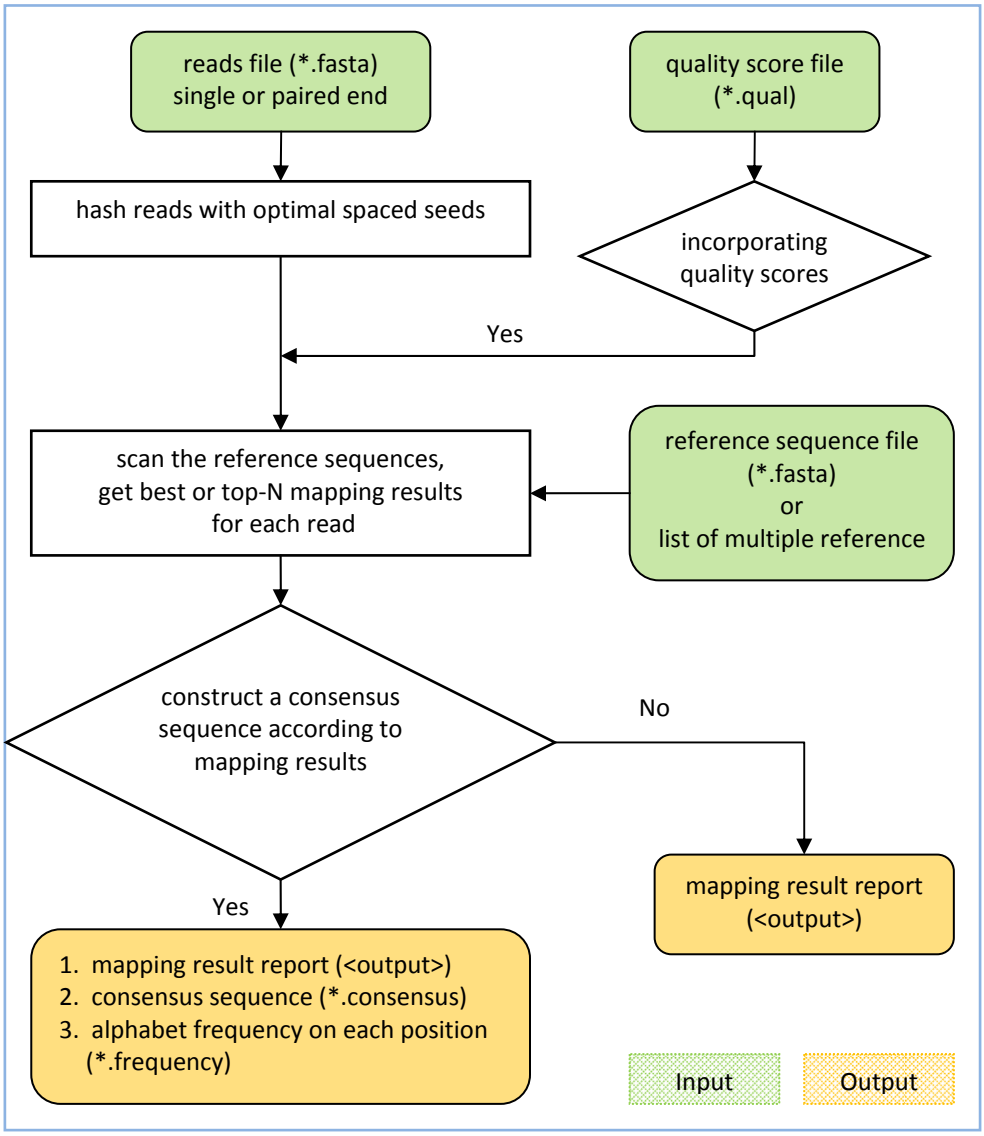
We recommend users to concatenate reads data from many lanes into one read file, since ZOOM indexes the reads and scans the reference genome once.

ZOOM designs a framework to use a multiple spaced seeds strategy, which is specially designed for short reads mapping to achieve 100% sensitivity for a large range of read length *m* and mismatch number *k*. ZOOM compiles different built-in modules for practically used *(m,k)* cases. When the required mismatch number is beyond *k*, the sensitivity will decrease slightly. ZOOM supports various read lengths in one input file. The reads are categorized according to their length, and different ZOOM modules are selected automatically to map them to the reference sequence.

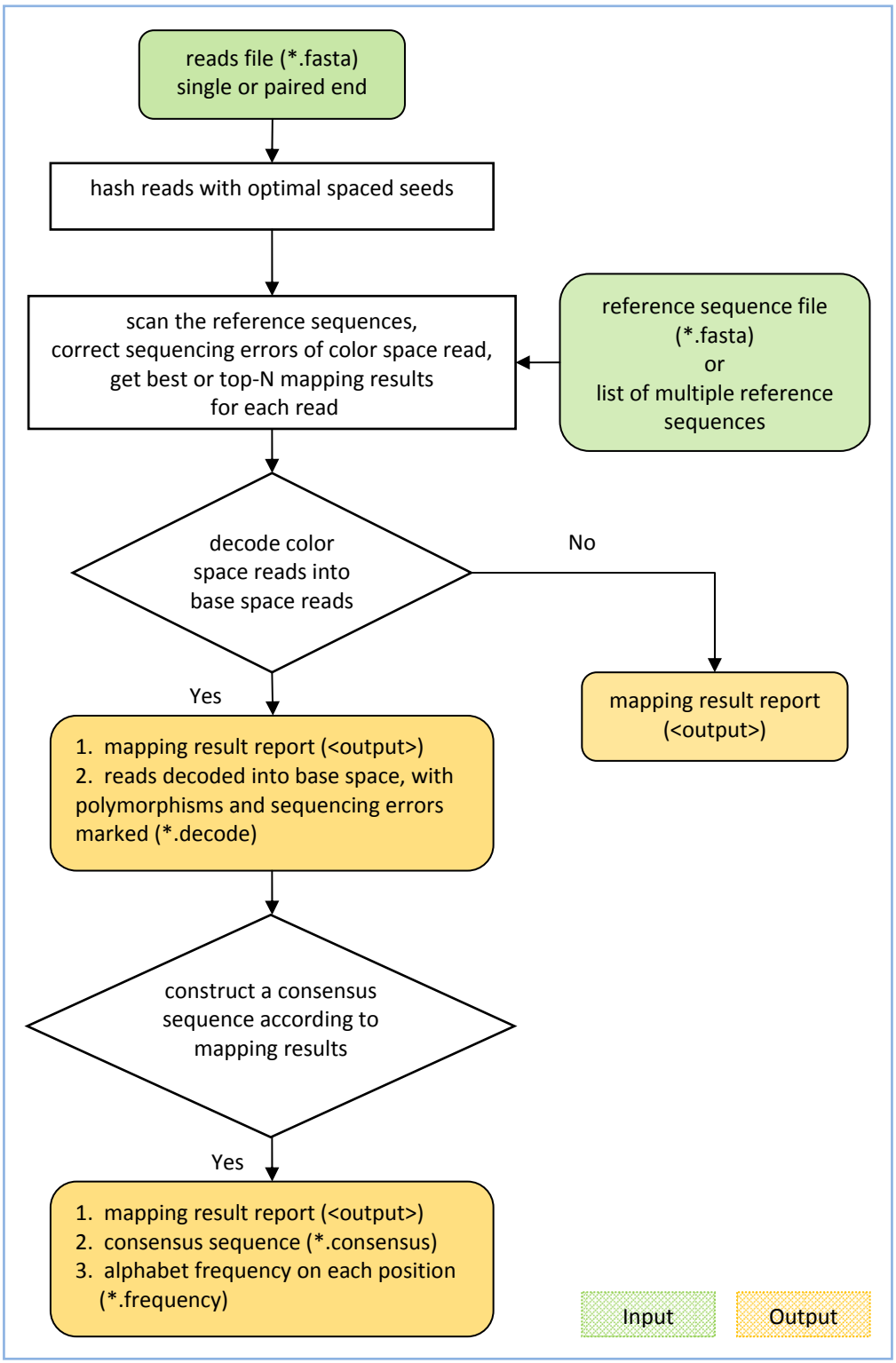
Since ABI SOLiD reads data is in color space, rather than in normal FASTA format of Illumina/Solexa data, ZOOM adopts different strategies to deal with these two kinds of data, which are shown in the following workflows.

ZOOM workflow

Workflow for Illumina/Solexa Data



Workflow for ABI/SOLiD Data



Quick Start

Quick start for Illumina/Solexa data

1. mapping reads to one reference sequence with two mismatches

```
ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.m2 –mm 2
```

ZOOM maps reads in *BAC.fasta* to *chr6.fa*, allowing two mismatches between each read and its target region on *chr6.fa*. The mapping results are recorded in the output file *BAC.vs.chr6.m2*. When n mismatches are required, use "–mm n".

2. mapping reads to multiple reference sequences

There are three ways:

a) The first way is to list all files containing different reference sequence after "*-g*" parameter.

```
ZOOM –i BAC.fasta –g chr1.fa chr2.fa chr3.fa \
–o BAC.vs.three_genome.m2 –mm 2
```

ZOOM maps reads in *BAC.fasta* to three reference sequences --- chromosome 1 (*chr1.fa*), chromosome 2 (*chr2.fa*) and chromosome 3 (*chr3.fa*).

b) The second way is to record the file names in one list file and put the list file name after "*-g*" parameter as input:

```
ZOOM –i BAC.fasta –g all.lst –o BAC.vs.all.m2 –mm 2
```

Reads are mapped to all human chromosomes in FASTA format. The names of each chromosome file are listed in the file *all.lst*.

c) The third way is to input a file contain multiple reference sequences in FASTA format.

```
ZOOM –i BAC.fasta –g referenece.fa –o BAC.vs.ref.m2 –mm 2
```

reference.fa may contains three chromosomes.

3. mapping reads to reference sequence with insertions/deletions

In this release, ZOOM can only deal with one gap of various lengths.

There are two ways:

a) designate the edit distance allowed between reads and the reference sequence:

```
ZOOM –i BAC.fasta –g all.lst –o BAC.vs.all –ed 2
```

ZOOM maps reads in *BAC.fasta* to all chromosomes listed in *all.lst*, allowing at most two edit distances, which means the summation of mismatches and the length of insertion/deletion.

b) designate the mismatches and length of insertion/deletion allowed between reads and the reference sequence respectively:

```
ZOOM –i BAC.fasta –g all.lst –o BAC.vs.all –mm 2 –id 3
```

ZOOM maps reads in *BAC.fasta* to all chromosomes listed in *all.lst*, allowing two mismatches and one additional gap whose length is up to three.

4. dealing with multiple hits for each read

ZOOM maps reads to all candidate target positions on a reference sequence within an assigned difference threshold. Users can choose to output the unique best mapping result or the top N best mapping results for each read.

```
ZOOM –i sRNA.fasta –g chr6.fa –o RNA.vs.chr6 –mm 2 –mk 100
```

ZOOM maps reads in *sRNA.fasta* to the reference sequence *chr6.fa*, allowing two mismatches and outputs the best 100 mapping results of each mapped read. The uniquely mapped reads will be recorded in the file *RNA.vs.chr6*, while the top 100 results for each read is reported in the file *RNA.vs.chr6.all*.

5. mapping reads taking the advantage of quality scores

```
ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.rh4.m2 –mm 2 –rh 4
```

ZOOM maps reads in *BAC.fasta* to *chr6.fa*, allowing two mismatches at the positions whose base quality score is no less than four. The uniquely mapped reads are reported in *BAC.vs.chr6.rh4.m2*.

6. dealing with paired-end reads

The two ends of a pair of Illumina/Solexa paired-end data should come from forward strand and reverse strand respectively.

Paired-end (mate-pair) reads can be inputted in two ways:

a) Two input files recording reads from two ends separately:

```
ZOOM –if BAC_forward.fa –ir BAC_reverse.fa –pe –pemin 1495 \
–pemax 2027 –g all.lst –mm 2 -o BAC_paired.vs.all.m2
```

ZOOM will automatically mate up the reads from *BAC_forward.fa* and *BAC_reverse.fa* according to their name (please make sure the names of two ends are in format *_F3 and *_R3 respectively). Then ZOOM maps the reads in pairs. Each read in a pair allows two mismatches to the reference sequence. Only those read pairs whose two ends are mapped to forward strand and reverse strand of the reference sequence respectively within the distance between 1495bp and 2027bp are reported.

b) One FASTA input file with two reads of a pair appearing in odd record and even record respectively. (The first read and the second read form a pair; the third and the fourth form a pair…)

```
ZOOM –i paired_reads.fa –pe –pemin 1495 –pemax 2027 –g all.lst \
–ed 2 -o paired_reads.vs.all.ed2
```

ZOOM will map the read pairs in *paired_reads.fa*. Each read of one pair is mapped to the reference genome within two edit distances. When reads number is unacceptable for RAM, the best way is to mate up reads into one file and divide the file into several parts, then apply paired-end mapping as in b) to the several parts separately, and finally merge the mapping results and do the assembly. In later release of ZOOM, this procedure will be handled automatically.

7. assembly according to mapping results

Assembly can be done by giving read files or mapping results:

a) assembly with reads:

```
ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.ed2 –ed 2 –asbl –mc 3
```

ZOOM maps reads in *BAC.fasta* to *chr6.fa*, allowing two edit distances and reconstructs a consensus sequencing using the uniquely mapped reads information. The uniquely mapped reads information is reported in *BAC.vs.chr6.ed2*. The coverage of each position of the consensus sequence is in the file *BAC.vs.chr6.ed2.frequency*. The reconstructed consensus assembly in FASTA format is in *BAC.vs.chr6.ed2.consensus*, where the bases whose coverage is less than three will be marked by lowercase letters.

b) assembly with mapping results:

```
ZOOM –i BAC.fasta –g chr6.fa –mii my.rst –o BAC.vs.chr6.m2 –asbl
```

ZOOM loads in mapping results from the file *my.rst*, carries out assembly according to the mapping results and outputs the consensus sequence in the file *BAC.vs.chr6.m2.consensus*. The coverage and frequency information is in the file *BAC.vs.chr6.m2.frequency*. This option is useful especially when the data set is divided into several parts and run separately. Users can merge the mapping results into one file and do assembly using this option.

8. getting the coverage information and the frequency of {A,C,G,T} on each position

```
ZOOM –i ChIP_Seq.fasta –g ref.fa –o ChIP.vs.ref.m2 \
–ed 2 –asbl –mc 100
```

ZOOM maps reads in *ChIP_Seq.fasta* to *ref.fa*, allowing two edit distances and reconstructs a consensus sequence using the uniquely mapped reads information. The uniquely mapped reads information is reported in *ChIP.vs.ref.m2*. The coverage and nucleotide frequency in each position of the consensus sequence is in the file *ChIP.vs.ref.m2.frequency*. The reconstructed consensus sequence in FASTA format is in file *ChIP.vs.ref.m2.consensus*, where the bases whose coverage is less than 100 will be noted by lowercase letters.

9. getting results with 100% sensitivity

ZOOM designs a set of spaced seeds sets which can guarantee 100% sensitivity of mismatch detection. For speed consideration, when the number of mismatches is beyond two, ZOOM will use the spaced seeds set designed for two mismatches by default (For example, "*ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.m4 –mm 4*" will use the set of spaced seeds for two mismatches). If you want to achieve 100% sensitivity in the case of a larger number of mismatches, please use the "-sv" parameter to designate the mismatch threshold.

```
ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.mr3 –sv r3 –mm 3
```

ZOOM maps *BAC.fasta* to chromosome 6, allowing up to three mismatches with full sensitivity.

```
ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.mr4 –sv r4 –mm 4
```

ZOOM maps *BAC.fasta* to chromosome 6, allowing up to four mismatches with full sensitivity.

```
ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.mr4 –sv r4 –mm 6
```

ZOOM maps *BAC.fasta* to chromosome 6, allowing up to six mismatches using the spaced seeds set which has 100% sensitivity for no more than four mismatches.

Of course, more sensitivity means more time. One way to save time is to run ZOOM by default first, and then utilize the 100% sensitivity seeds only for those unmapped reads.

Quick start for ABI SOLiD data

1. mapping reads to one reference sequence with two mismatches

```
ZOOM –i Yoruban.15M.csfasta –g chr1.fa
-o Yoruban.15M.vs.chr1.m2 -mm 2
```

The color space reads are automatically detected. ZOOM maps the color space reads in *Yoruban.15M.csfasta* to chromosome 1 in the file *chr1.fa*, allowing two errors (either two genome differences and no sequencing errors on the color space read, or one genome difference and one color space sequencing error, or no genomic differences and two color space sequencing errors). Sequencing errors are corrected automatically in the mapping process. Uniquely mapping results are recorded in file *Yoruban.15M.vs.chr1.m2*. When n mismatches are required, use "–mm n".

2. mapping reads to multiple reference sequences

There are three ways:

a) The first way is to list all files containing different reference sequences after "*-g*" parameter:

```
ZOOM –i Yoruban.15M.csfasta –g chr1.fa chr2.fa chr3.fa \
–o Yoruban.15M.vs.chr1-3.m3 -mm 3
```

ZOOM maps reads in *Yoruban.15M.csfasta* to three reference sequences --- chromosome 1 (*chr1.fa*), chromosome 2 (*chr2.fa*) and chromosome 3 (*chr3.fa*). Unique mapping results are recorded in file *Yoruban.15M.vs.chr1-3.m3*.

b) The second way is to record the file names in one list file and put the list file name after "*-g*" parameter as input:

```
ZOOM –i Yoruban.15M.csfasta –g all.lst \
–o Yoruban.15M.vs.all.m2 –mm 2
```

Reads are mapped to all human chromosomes. The names of each chromosome file are listed in the file *all.lst*.

c) The third way is to input a file contain multiple reference sequences in FASTA format.

```
ZOOM –i Yoruban.15M.csfasta –g referenece.fa \
 –o Yoruban.15M.vs.ref.m2 –mm 2
```

reference.fa may contains three chromosomes.

3. dealing with multiple hits for each read

ZOOM maps reads to all candidate target positions within specified mismatch thresholds on the reference sequence. Users can choose to output the unique best mapping result or the top N best mapping results for each read.

```
ZOOM –i Ecoli_shiraz.csfasta –g ecoli_genome.fa \
–o Ecoli_shiraz.vs.all.m2 –mm 2 –mk 100
```

ZOOM maps reads in *Ecoli_shiraz.csfasta* to the Ecoli genome *ecoli_genome.fa,* allowing two mismatches and outputs the best 100 mapping results of each mapped read. The uniquely mapped reads will be recorded in the file *Ecoli_shiraz.vs.all.m2*, while the top 100 results for each read is reported in the file *Ecoli_shiraz.vs.all.m2.all.*

4. decode color space reads into nucleotides according to the mapping results

```
ZOOM –i Ecoli_shiraz.csfasta –g ecoli_genome.fa \
–o Ecoli_shiraz.vs.all.m4 –mm 4 –dcs
```

ZOOM maps the color space reads in *Ecoli_shiraz.csfasta* to the Ecoli genome *ecoli_genome.fa,* allowing four errors. Mapping results are recorded in the file *Ecoli_shiraz.vs.all.m4*. ZOOM decodes color space reads into base space and highlights both sequencing error positions and polymorphism positions in the file *Ecoli_shiraz.vs.all.m4.decode.*

5. assembly according to mapping results and decode reads into base space

When assembly is required, the decoding process will be automatically added. Assembly can be done by giving either reads file or mapping results.

a) assembly by reads file:

```
ZOOM –i Ecoli_shiraz.csfasta –g ecoli_genome.fa \
–o Ecoli_shiraz.vs.all.m4 –mm 4 –asbl –mc 10
```

ZOOM maps the color space reads in *Ecoli_shiraz.csfasta* to *ecoli_genome.fa*, allowing four errors, decodes the color space reads into base space and reconstructs a consensus

sequence using the decoded reads which are uniquely mapped. The unique mapping information is reported in *Ecoli_shiraz.vs.all.m4*. The coverage and frequency of {*A,C,T,G*} at each position of the consensus sequence is in the file *Ecoli_shiraz.vs.all.m4.frequency*. The reconstructed consensus sequence in the FASTA format is in file *Ecoli_shiraz.vs.all.m4.consensus*, where the bases whose coverage is less than ten will be marked by lowercase letters.

b) assembly by mapping results

```
ZOOM –i Ecoli_shiraz.csfasta –g ecoli_genome.fa \
–mii my.map –o Ecoli_shiraz.vs.all –asbl –mc 10
```

ZOOM loads mapping results, and decodes reads into nucleotides according to the mapping results from *my.map*. Then ZOOM performs assembly and outputs the consensus sequence in the file *Ecoli_shiraz.vs.all.consensus*. The coverage and frequency information of each position are reported in the file *Ecoli_shiraz.vs.all.frequency*. This option is useful especially when the data set is divided into several parts and run separately. Users can merge the mapping results into one file and do assembly using this option.

6. dealing with paired-end reads

Paired-end (mate-pair) reads can be inputted in two ways:

a) two input files recording reads from two ends separately:

```
ZOOM –if Ecoli_shiraz_F3.csfasta –ir Ecoli_shiraz_R3.csfasta\
–g ecoli_genome.fa –o  Ecoli_shiraz_paired.vs.all.m2 \
–pe –pemin 1495 –pemax 2027 –mm 2 –dcs
```

ZOOM will automatically pair the reads from *Ecoli_shiraz_F3.csfasta* and *Ecoli_shiraz_R3.csfasta* according to their name. Then ZOOM maps the reads in pairs to the *Ecoli* genome *ecoli_genome.fa*, allowing two errors on each mate of the pair. The mapping positions of the two mates of a pair should be on the same strand of the reference sequence and within the distance of 1495bp to 2027bp. The decoded mapped pairs are recorded in file *Ecoli_shiraz_paired.vs.all.m2.decode*.

b) One input file with two reads of a pair appearing in odd record and even record, respectively. (The first read and the second read form a pair; the third and the fourth form a pair…)

```
ZOOM –i Ecoli_paired_reads.csfasta –g ecoli_genome.fa \
-o ecoli_paired_reads.vs.all.m2 –pe –pemin 1495 –pemax 2027 –mm 2
```

ZOOM will map the read pairs in *Ecoli_paired_reads.csfasta*. Each read of one pair are mapped to the reference genome with two errors. When reads number is unacceptable for RAM, the best way is to mate up reads into one file and divide the file into several parts, then apply paired-end mapping as in b) to the several parts separately, and finally merge the mapping results and do the assembly. In later release of ZOOM, this procedure will be handled automatically.

7. assembly with paired-end reads

```
ZOOM –if Ecoli_shiraz_F3.csfasta –ir Ecoli_shiraz_R3.csfasta \
–g all.lst –o Ecoli_shiraz_paired.vs.all.m2 \
–pe –pemin 1495 –pemax 2027 –mm 2 –asbl –mc 5
```

After mapping the paired-end reads and decoding the mapped reads into nucleotides, ZOOM uses the decoded reads to construct a consensus sequence in file *Ecoli_shiraz_paired.vs.all.m2.consensus*. The coverage and frequency of {*A,C,T,G*} on each position of the consensus sequence is recorded in the file *Ecoli_shiraz_paired.vs.all.m2.frequency*, where the bases whose coverage is less than five will be marked by lowercase letters.

8. getting results with 100% sensitivity

ZOOM designs a set of spaced seeds sets which can guarantee 100% sensitivity of mapping results. For speed consideration, when the number of mismatches is beyond two, ZOOM will use the spaced seeds set designed for two mismatches by default (For example, "*ZOOM –i Yoruban.csfasta –g all.lst –o Yoruban.vs.all.m4 –mm 4*" will use the set of spaced seeds for two mismatches). If you want to achieve 100% sensitivity in the case of larger number of mismatches, please use "-sv" parameter to designate the mismatch threshold.

```
ZOOM –i Yoruban.csfasta –g all.lst –o Yoruban.vs.all.mr3 \
–sv r3 –mm 3
```

ZOOM maps *Yoruban.csfasta* to all human chromosomes, allowing up to three mismatches with full sensitivity.

```
ZOOM –i Yoruban.csfasta –g all.lst –o Yoruban.vs.all.mr4 \
–sv r4 –mm 6
```

ZOOM maps *Yoruban.csfasta* to all human chromosomes, allowing up to six mismatches, using the spaced seeds set which have 100% sensitivity for no more than four mismatches.

Of course, more sensitivity means more time. One way to save time is to run ZOOM by default first, and then utilize the 100% sensitivity seeds only for those unmapped reads.

Input of ZOOM

Reference sequence file format

Reference sequence file contains reference sequences in FASTA format. FASTA formatted files have the reference sequence label and/or description on the first line starting with the ">" character and have the actual sequence following underneath. The letters of the sequences are case insensitive.

If multiple reference sequences need to be scanned, you can choose to:

1. put multiple reference sequences in one file as multi-fasta format.

Example format of a reference sequence file:

```
>Reference_sequence_1_name
AGGACTATATTGCTCTAATAAATTTGCGGTTCTTAAAAACTCAATGT
TGTAAAAATGTCACTTCTTCCCAAA…
>Reference_sequence_2_name
TATTGCTCTAATAAATTTGCGGTATTGCTCTAATAAATTTGCGGCCC
TAATAAATTTGCG…
```

2. list all the file names after "*-g*" parameter

Example format of a reference sequence file:

```
>Reference_sequence_name
AGGACTATATTGCTCTAATAAATTTGCGGTTCTTAAAAACTCAATGT
TATTGCTCTAATAAATTTGCGGTATTGCTCTAATAAATTTGCGGCCC
TGTAAAAATGTCACTTCTTCCCAAA…
```

3. use multiple files to store the sequences and list the file names of each reference sequence in a list file.

Example format of a reference list file *human_genome.lst*:

```
chr1.fa
chr2.fa
chr3.fa
…
chr22.fa
```

Each line of a reference list file is the name of a reference sequence file.

ZOOM can recognize all the above formats, so when the chromosomes are stored separately, you do not need to merge these chromosome files into a large file using the second and the third way.

Sequence reads file format for Illumina/Solexa Reads

ZOOM accepts five types of Illumina/Solexa read files as input. These file formats are automatically recognized. The letters of the read sequences are case insensitive.

1. FASTA format

Example of FASTA format:

```
>read1A_1
AGGACTATATTGCTCTAATAAATTTGCCGGTTCTTA
>read1A_2
TCTAATAAATTTGCCGGTTCTTAAAAACTCAAT
>read1A_3
CTCTAATAAATTTGCCGGTTCTTAAAAACTCAATGG
>read1A_4
ATATTGCTCTAATAAATTTGCCGGTTCTTAATA
>read1A_5
ATTTGCCGGTTCTTAAAAACTCAATGGTGTAAAAAT
```

In ZOOM, FASTA format files have no sequencing quality scores and all the read bases including N are considered equally relevant.

2. One read per line with quality scores

Example format:

```
4_87_872_656 ACGTACNT 40 40 40 40 35 60 0 70
4_87_630_913 ACCCAC 40 40 40 20 29 38
```

The first column contains the read sequence label or description, the second column contains the sequence data of the read, and the third column contains the quality scores for each base. Note that the read sequence label or description should not have a space inside.

3. *_seq.txt and *_prb.txt Files

Example of *_seq.txt file format:

```
1     1     125    701 GCTACCCTTTAGGTTTAA
1     1     8      550 TCGGAATTCCTGATTTCC
```

Each line of the sequence file records the channel number, tile number, x position, y position of each sequence read, and the sequence of the read. The labels of each read sequence are in the format of <*channel number*>_<*tile number*>_<*x position*>_<*y position*>.

Example of *_prb.txt file format:

```
40 -40 -40 -40 40 -40 -40 -40 40 -40 -40 -40 40 -40 -40 -40...
40 -40 -40 -40 40 -40 -40 -40 40 -40 -40 -40 -40 40 -40 -40...
```

The *_prb.txt file contains the quality score of each possible nucleotide base for the given cycle number. Four numbers, such as *-40 40 -40 -40*, each separated by a space, are the sequencing quality scores associated for each possible nucleotide, *ACGT*, respectively. The tab character is used to separate the bases of each cycle. Each line of *.prb is associated with the corresponding line of *_seq.txt.

4. *_prb.txt Files

Example file:

```
40 -40 -40 -40 40 -40 -40 -40 40 -40 -40 -40 40 -40 -40 -40...
40 -40 -40 -40 40 -40 -40 -40 40 -40 -40 -40 -40 40 -40 -40...
```

*_prb.txt file is the same with description of part 3 above. ZOOM can process this file without corresponding *_seq.txt file as input. The difference is that the labels of each read sequence are automatically assigned.

5. FASTQ Format

ZOOM automatically recognizes the Sanger FASTQ format, whose quality score of each position equals ord($q) – 33.

Example of FASTQ format:

```
@071113_EAS56_0053:1:1:756:463
GTGATTAGTGAAACATAAAATAGTTTCATGTTGAAA
+
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIGIAI
@071113_EAS56_0053:1:1:813:752
GTGGGAAAAACTGAAATACATTGCTTATGGATTCAT
+
IIIIIIIIIIIIIIIIIIIIIIIIIIIIA?II/%I
@071113_EAS56_0053:1:1:556:366
TTTTCTTGATTTCATTTTAGCACAATCATTAATTAC
+
```

ZOOM will extract sequence and quality score for each read automatically. However, ZOOM will utilize the quality scores only when instructed by parameter "*-rh*".

Sequence reads file format for ABI SOLiD Reads

Applied Biosystems SOLiD *.csfasta File

ABI represents their reads in color spaces. Any two adjacent nucleotide bases from the reads are represented by one of four colors. The mapping relationship between base space (nucleotide) and color space is denoted in the definitions section under "color space".

In this release, ZOOM accepts the color space data (*.csfasta) from SOLiD, in which each read is a numeric string prefixed by a single base. The base that precedes the numeric (color code) data is the final base of the sequencing adapter.

Example of *.csfasta file format:

```
>1_6_678_F3
T003001100000212032222022 3
>1_6_1142_F3
T101101032131312332102222 2
>1_6_1616_F3
T222001221312132222311332 0
>1_6_1634_F3
T002221213012022001111011 1
>1_6_1701_F3
T000131001000003100011103 1
>1_6_1813_F3
T311203003023132211330032 3
```

Output of ZOOM

Output for Illumina/Solexa reads

By default, ZOOM will output the mapping results of each mapped read from the input data set. If assembly process is required by the users, ZOOM will reconstruct a consensus sequence using the mapping information and output the frequency of {*A, C, G, T, N*} on each position of the reconstructed consensus sequence.

<output>: results of uniquely mapped reads - default output

This file contains the read mapping information. Each line of the file corresponds to a mapped read.

Each line contains four fields delimited by the *tab* character:

```
<read label> <reference offset> <strand> <mismatch number>
```

`Read label`: the name of the mapped read

`Reference offset`: the position that the read mapped on the reference sequence, starting from zero. By default, the leftmost position is always returned, no matter whether the read is mapped to the positive or negative strand. For multiple reference sequences, there are three choices for users to report the offset:

- **sequence_label:offset.** The offset in each reference sequence is counted separately. ZOOM reports the reference sequence label along with the offset inside it. For example, suppose three reference sequences *(chr1.fa, chr6.fa, chr8.fa)* are used. A read is mapped to the 123467 position (0-based) of the *chr6.fa*. The label will be *chr6:123467*

- **The offset is uniformly concatenated**, with 286 Ns padded between adjacent sequences. In the above example, if the length of chr1.fa is 160,000,000, the reference offset of this mapped read is 160000000+286 +123467= 160123753.

- **sequence_index:offset.** The relative index (0-based) of the reference sequences is used. For the above example, 1:123467 will be reported. The index of the three reference sequences is 0, 1, and 2, respectively.

When indels are present, the indel information is stored inside <reference offset> using the following format:

```
<reference offset>_<location of indel>_<length of indel>
```

The first field is the reference offset, the second field is the indel position (positive for an insertion and negative for a deletion), and the last field is the indel length. For the mapping on the reverse strand, the reference offset is still the leftmost position. For example:

```
              ____ reference offset = 1021
             /
             |
1021_3_2:    accacgtacgtacgtacgtacgta     <=reference sequence
             acc  gtacgtacgtacgtacgta     <=read

1021_-3_2:   acc  gtacgtacgtacgtacgt  <=reference sequence
             accacgtacgtacgtacgtacgt  <=read
```

`Strand`: the strand of the reference sequence that the read is mapped to. A "+" means the read is mapped to the positive strand of the reference sequence. A "-" means the read is mapped to the negative strand of the reference sequence.

`Mismatch number`: the Hamming distance, or the number of mismatches between the read and the target region on the reference sequence it maps to. When edit distance is permitted, this field still contains the mismatches between the read and its target region on the reference sequence; while the length of the indel is reported in the reference offset part as described above.

An example output file:

```
4_87_923_316 30634114     +     0
4_87_239_596 30677447     +     1
4_87_933_894 30625278     -     2
4_87_699_645 30624778_-8_1+     0
4_87_156_721 30605547_18_2+     1
```

Only uniquely mapped reads are reported. An uniquely mapped read is a read that can be mapped to only one reference sequence position with the smallest edit distance, or for equal edit distances, the shortest indel length (under the consideration that indel operation is less acceptable than mutation). Or, as MAQ states, an unique mapping is the mapping that is always better than other mappings.

For example, let A and B be two reference positions:

- If a read can be mapped to position A and position B on the reference genome, with two mismatches for A, and one mismatch for B, then B is reported as the unique mapping position for this read.
- If both A and B contain two mismatches, then this read is not reported

- If there are two mismatches and an indel of length one for A, one mismatch and an indel of length two for B, then A is reported.
- If there are two mismatches for both A and B, an indel of length one for A, an indel of length two for B, then A is reported.
- If, there are two mismatches and an indel of length one for both A and B, then this read is not reported

How to interpret the results:

1. `4_87_933_894  30625278  -  2`

   The read whose label is *4_87_933_894* is uniquely mapped to the reverse complement sequence of the target region starting from 30625279 (if the first position is 1), with two mismatches.

2. `4_87_699_645  30624778_-8_1 + 0`

   The read *4_87_699_645* is mapped to the 30624779^{th} nucleotide base of the positive reference sequence, with zero mismatch and a deletion of length one at the 8^{th} nucleotide.

3. `4_87_156_721  30605547_18_2 + 1`

   The read *4_87_156_721* is uniquely mapped to the 30605548^{th} nucleotide base of the positive reference sequence, with one mismatch and an insertion of length two at the 18^{th} nucleotide.

<output>.all : the top N mapping results of each read

If those reads mapped to multiple positions on the reference sequences are required to be reported, then an *<output>.all* file will be generated along with the *<output>* file. This file will contain the best N mapping results of each read. That means, if a read can be mapped to multiple positions on the reference sequences within the mismatches or edit distances threshold specified by user, the top N mapping results out of all mapped positions will be outputted in *<output>.all*.

An example of *<output>.all*:

```
4_100_1001_323      152362 +      0
4_100_1001_323      46358  -      0
4_100_1001_323      84922  -      1
4_100_1001_500      9916   +      1
4_100_1001_500      31388  -      2
4_871_1561_721      8567   -      2
4_871_1561_721      30605547_18_2+      1
```

<output>.frequency : coverage information of each position of the reconstructed consensus sequence

If the additional assembly process (specified by *–asbl* argument) is carried out, <*output*>*.frequency* contains the reads coverage information for each nucleotide from the reference sequence on each line of the file. Uniquely mapped reads in file <*output*> on each position are used to count the number of nucleotides on this position. In the paired-end reads case, uniquely mapped read pairs are used. The output format has nine fields:

```
<reference offset>  #A #C #G #T #N #deletion #insertion #coverage
```

Explanation of each field:

<reference offset>	reference sequence offset, 0-based
#A, #C, #G, #T, #N	the nucleotide frequency calculated from the mapped reads at that position
#deletion	the number of reads having a deletion at that position.
#insertion	the number of reads having an insertion after that position.
#coverage	the reads coverage at that position.

The following list contains an example output of the <output>.frequency

#ref_offset	#A	#C	#G	#T	#N	#del	#ins	#cov
0	0	0	539	0	0	0	0	539
1	760	0	0	1	0	0	0	761
2	28	2	4	889	0	0	0	923
3	8	1037	0	0	0	0	0	1045
4	1185	0	4	1	0	0	0	1190
5	24	1315	5	1	0	0	0	1345
6	1478	0	9	2	0	0	0	1489
7	3	0	1618	0	0	0	0	1621
8	3	1	1749	1	0	0	0	1754
9	8	7	63	1894	0	0	0	1972
…	…	…	…	…	…	…	…	…

If multiple reference sequences are used, the frequency file <*output*>.*frequency* will output the above table for each reference sequence respectively.

<output>.consensus : the reconstructed consensus sequence

<*output*>.*consensus* contains the assembly of reads data, based on <*output*>.*frequency*. There is a parameter called <*mincov*> (specified by the –*mc* command line argument) which stands for "minimal reads coverage". Those positions whose coverage is below <*mincov*> will be regarded as an unreliable assembly position and will be denoted in lowercase.

If multiple reference sequences are used, multiple consensus sequences will be output in multi-fasta format.

The assembly process constructs a consensus sequence using the following process:

If [#deletion] > [#A+#C+#G+#T+#N], then there is a deletion at this position, otherwise the nucleotide with the highest frequency will be chosen. If the read coverage is less than <*mincov*>, the letter is lowercased (unreliable base), otherwise it is uppercased (reliable).

If [#insertion] > [#continuous] (the number of reads who doesn't agree that there should be an insertion), then there is an insertion after this position, and the sequence segment with the highest frequency (collected from reads) will be inserted into the consensus sequence.

Output for ABI SOLiD reads

ZOOM can map ABI SOLiD reads within a given Hamming distance, which is the number of mismatches allowed between the read and its target region on the reference sequence. ABI SOLiD reads use the color space format. The differences between the read in color space and the reference sequence are caused either by sequencing error or genomic differences, such as mutation or SNP. Sequencing errors may cause some reads to be mapped incorrectly to the reference sequence. ZOOM is able to distinguish sequencing errors from genomic differences by correcting the sequencing error, and allows more reads to be correctly mapped. ZOOM is also able to decode mapped color space reads after error correction and highlight both genomic differences and sequencing errors.

<output>: results of uniquely mapped reads - default output

This file contains the read mapping information. Each line of the file corresponds to a mapped read.

Each line contains four fields delimited by *tab*:

```
<read label> <reference offset> <strand> <total error number>
```

`Read label`: the name of mapped read

`Reference offset`: the position that the read mapped on the reference sequence, starting from zero. By default, the leftmost position is always returned, no matter whether the read is mapped to the positive or negative strand. For multiple reference sequences, there are three choices for users to report the offset:

- **sequence_label:offset.** The offset in each reference sequence is counted separately. ZOOM reports the reference sequence label along with the offset inside it. For example, suppose three reference sequences *(chr1.fa, chr6.fa, chr8.fa)* are used. A read is mapped to the 123467 position (0-based) of the *chr6.fa*. The label will be *chr6:123467*

- **The offset is uniformly concatenated**, with 286 Ns padded between adjacent sequences. In the above example, if the length of chr1.fa is 160,000,000, the reference offset of this mapped read is 160000000+286 +123467= 160123753.

- **sequence_index:offset.** The relative index (0-based) of the reference sequences is used. For the above example, 1:123467 will be reported. The index of the three reference sequences is 0, 1, and 2, respectively.

  When indels are present, the indel information is stored inside <reference offset> using the following format:

```
<reference offset>_<location of indel>_<length of indel>
```

  The first field is the reference offset, the second field is the indel position (positive for an insertion and negative for a deletion), and the last field is the indel length. For the mapping on the reverse strand, the reference offset is still the leftmost position. For example:

```
              ____ reference offset = 1021
             /
            |
1021_3_2:    accacgtacgtacgtacgtacgta     <=reference sequence
             acc  gtacgtacgtacgtacgta     <=read

1021_-3_2:   acc  gtacgtacgtacgtacgt  <=reference sequence
             accacgtacgtacgtacgtacgt  <=read
```

  `Strand`: the strand of the reference sequence that the read is mapped to. A "+" means the read is mapped to the positive strand of the reference sequence. A "-" means the read is mapped to the negative strand of the reference sequence.

  `Total error number`: Total error number is the summation of the number of mismatches due to genomic differences and sequencing errors of the read. The <*-dcs*> option will decode the color space read into nucleotides, in order to separate genomic differences from sequencing errors.

  Again, only uniquely mapped reads are reported. A uniquely mapped read is a read that can be mapped to only one reference sequence position with the smallest error number of the summation of genome differences and sequencing errors.

  An example of <*output*> file:

```
589_59_92_F3 300369       +     0
589_59_361_F31502269      -     0
589_59_425_F31175665      +     2
589_59_437_F32168137      -     1
```

Interpretation of the results:

589_59_92_F3 is mapped to the 300369 position of the positive strand of the reference sequence with zero polymorphisms and zero sequencing errors.

589_59_425_F3 is mapped to the 1175665 position of the positive strand of the reference sequence with two errors, which is the number of polymorphisms on the base space plus sequencing error numbers.

<output>.all : the top N mapping results of each read

If those reads mapped to multiple positions on the reference sequences are required to be reported, then <*output*>*.all* file will be generated besides the <*output*> file. This file will contain the best N mapping results of each read. That means, if a read can be mapped to multiple positions on the reference sequences within the mismatch threshold specified by user, the top N mapping results out of all mapped positions will be outputted in <*output*>*.all*.

An example of <*output*>*.all*:

```
589_59_92_F3 300369       +     0
589_59_92_F3 4145838      +     0
589_59_92_F3 2529174      -     1
589_59_361_F31502269      -     0
589_59_361_F33449163      +     2
589_59_425_F31175665      +     2
589_59_437_F32168137      -     1
```

<output>.decode : decoded color space reads after error-correction

Every three lines in the file <*output*>*.decode* denote one mapped color space read in the following format:

```
1.  <read_label> <genomic differences> <sequencing errors>
2.  decoded nucleotide sequence
3.  mark of sequencing error position
```

1. The first line after *">"* contains the read label, then genome difference number, and finally the sequencing error number. All are separated by a space.

2. The second line contains the decoded nucleotide sequence of the read after error correction. Genomic differences will be highlighted by lowercase letters. Notice

that the first position of the color space read is coded by the first base of the read and the last base of the adapter. ZOOM doesn't include the last adapter base before the decoded sequence.

3. The third line marks the positions of sequencing errors by *"1"*, the positions without sequencing errors are denoted by *"0"*.

For example:

```
>1_45_1104_F3 0 2
GAGATGGAGTCTTGCTCTGTCATC
000100000000001000000000
>1_274_1271_F3 1 1
TGCAtCCTTGAGAGAAAGCCCGCT
000000000000000000100000
```

For read *1_45_1104_F3*, after correcting two sequencing errors on the color space read, it will be mapped to its target region with no mismatches.

For read *1_274_1271_F3*, after correcting one sequencing error on the color space read, the read will be mapped to the reference sequence with one polymorphism between its base space sequence and its target region. Since one polymorphism on the base space will cause two adjacent changes on the color space, there will be three mismatches between the color space read and the color space of the target region.

If both top N mapping results and decoded reads are required to output, ZOOM will output four files: *<output>, <output >.decode, <output>.all, <output>.all.decode.*

<output>.frequency : Coverage information of each position of the reconstructed consensus sequence

If the additional assembly process (specified by *–asbl* argument) is carried out, *<output>.frequency* contains the reads coverage information for each nucleotide from the reference sequence on each line of the file. Uniquely mapped reads in file *<output.decode>* on each position are used to count the number of nucleotides on this position. In the case of paired-end reads, uniquely mapped read pairs are used. The output format has nine fields:

```
<reference offset>  #A #C #G #T #N #deletion #insertion #coverage
```

Explanation of each field:

<reference offset>	reference sequence offset, 0-based
#A, #C, #G, #T, #N	the nucleotide frequency calculated from the mapped reads at that position
#deletion	the number of reads having a deletion at that position.
#insertion	the number of reads having an insertion after that position.
#coverage	the reads coverage at that position.

The following list contains an example output of the <output>.frequency

#ref_offset	#A	#C	#G	#T	#N	#del	#ins	#cov
0	0	0	539	0	0	0	0	539
1	760	0	0	1	0	0	0	761
2	28	2	4	889	0	0	0	923
3	8	1037	0	0	0	0	0	1045
4	1185	0	4	1	0	0	0	1190
5	24	1315	5	1	0	0	0	1345
6	1478	0	9	2	0	0	0	1489
7	3	0	1618	0	0	0	0	1621
8	3	1	1749	1	0	0	0	1754
9	8	7	63	1894	0	0	0	1972
…	…	…	…	…	…	…	…	…

If multiple reference sequences are used, the frequency file <*output*>.*frequency* will output the above table for each reference sequence respectively.

<output>.consensus : the reconstructed consensus sequence

<*output*>.*consensus* contains the assembly of reads data, based on <*output*>.*frequency*. There is a parameter called <*mincov*> (specified by the –*mc* command line argument) which stands for "minimal reads coverage". Those positions whose coverage is below <*mincov*> will be regarded as an unreliable assembly position and will be denoted in lowercase.

If multiple reference sequences are used, multiple consensus sequences will be output in multi-fasta format.

The assembly process constructs a consensus sequence using a major vote criteria, ie. the nucleotide with the highest frequency will be chosen. If the read coverage is less than <*mincov*>, the letter is lowercased (unreliable base), otherwise it is uppercased (reliable base).

Output for paired-end reads data

The output format is same as the output of single-end reads described in the above two sections. The only difference is that the reads in paired-end reads data are mapped in pairs. Each read of a pair is mapped to the reference sequence within the allowed mismatches or edit distances, as is done for the single-end read case. However, only if the two reads from the same paired-end read are mapped to the same strand of the reference sequence and the mapping positions are within the specified range, the mapping result will be reported. The error number of each read pair is represented by the summation of error numbers between each read and its target region on the reference sequence. If a pair is mapped to multiple positions in a reference sequence, then the best paired-end read map results is the one with the least error. In <*output*>.*all*, the top N best mapping results for each pair are reported.

Note that the two ends of a pair of Illumina/Solexa paired-end data should come from forward strand and reverse strand respectively, while the two ends of a pair of ABI SOLiD paired-end data should come from the same strand.

CMD-Line using ZOOM

Usage and Common Options

```
ZOOM [options] -i <reads_file> -g <reference_genome> -o <output>
```

Necessary Parameters

Parameter	Short Description
-i <input>	The input sequence reads data file.
-g <refseq>	The reference sequence or the reference sequence list file.
-o <output>	The output file name (actually, the name prefix for many output files).

Options for Paired-End Reads

ZOOM supports paired-end reads. ZOOM will detect the paired-end reads according to the reads label/name in the two input read files containing the forward direction and reverse direction separately. If you want to know the mapping results of the non-paired reads or those pairs mapping to different locations far away from each other, extract these reads and run ZOOM again using non-paired reads mapping options.

Parameter	Short Description
-pe	To map reads in pairs, the <-pe> option must be assigned. Two paired reads are considered mapped only if they are mapped onto the same strand and within the offset range specified by <–pemin> and <–pemax>. If this option is chosen, the <–permin> and <–pemax> parameters must be specified.

	Input reads are contained in two files containing one of the paired-end reads in different directions separately. ZOOM will detect the paired reads according to the reads name. Example: ./ZOOM –if read_F.fa –ir read_R.fa –o read.output –g ref.fasta –pe –pemin 1495 –pemax 2027
-if <input > -ir <input>	Input for paired-end data sets. Reads data from each end of the paired-end reads are stored in two input reads data file. The data are inputted through the two parameters respectively. The options are bound together. They are available only when <–pe> options is chosen. Example: ./ZOOM –if read_Forward.fa –ir read_Reverse.fa –pe –pmin 1014 –pmax 1500 –g all.lst –m 2 –o read.pair.vs.all.m2
-pemin <int>	The minimal distance between the mapping positions of the two reads that make up the read pair. Example: ./ZOOM –i read1.fna –o read.output –g ref.fasta –pe –pemin 1495 –pemax 2027 In this case, the mapping results of paired-end reads whose mapping position distance is less than 1495 from each other will not be reported.
-pemax <int>	The maximum distance between two mapping positions of the two reads that make up the paired-end reads. Example: ./ZOOM –i read1.fna –o read.output –g ref.fasta –pe –pemin 1495 –pemax 2027. In this case, the mapping results of paired-end reads whose mapping position distance is greater than 2027 will not be reported.

Options for Output Format

Parameter	Short Description
-gl	The mapped position of the read to the reference genome in the mapping result is denoted in the format of genome_name:offset. For example, if a read is mapped to a reference genome named chr6, then the result will be chr6:1229387. This is the default output of ZOOM. Example: ./ZOOM –i read1.fna –o read.output –g ref.lst –gl
-gi	The mapped position of the read to the reference genome in the mapping result is denoted in the format of genome_index:offset. The index starts from zero. For example, if reads are mapped to 22 human chromosomes and the read is located on the 1229387 of chromosome 6, then the result will be 5:1229387. Example: ./ZOOM –i read1.fna –o read.output –g ref.lst –gi
-mk <int>	The best N mapping results are reported for each read. By default, ZOOM only outputs those reads which are uniquely mapped to the reference genome. Example: ./ZOOM –i read1.fa –o read.output –g ref.fa –mk 100 The 100 best mapping results for each mapped read is reported in the file <read.output.all>, while the uniquely mapped reads is recorded in the file <read.output>.

Options for Assembly Reads and Finding Difference

ZOOM can assemble the reads according to the mapping result. Only uniquely mapped reads are used to reconstruct the consensus sequence. For ABI SOLiD color space data, the base space reads decoded from the color space are used. The

assembled sequence is the consensus sequence constructed by using the nucleotide with the highest frequency, according to the reads that were mapped to that position. <*output*>.*frequency* contains the frequency of the four nucleotides at each position calculated from the mapped reads.

Parameter	**Short Description**
-asbl	Performs an additional assembly process and generates heterozygote information reports. Example: ./ZOOM –i read1.fna –o read.output –g ref.fasta –asbl The assembled consensus will be in FASTA format in the file read.output.consensus. The heterozygous information reports will be named read.output.frequency.
-mc <mincov>	This stands for the minimal coverage threshold. The positions on the reference genome whose coverage is less than this threshold will be viewed as untrusted and denoted as lowercase letters in the assembled consensus. This options is available only if <-asbl> is used. Example: ./ZOOM –i read1.fna –o read.output –g ref.fasta –asbl –mc 20

Special Options for Illumina/Solexa Reads

Options for Mapping Criteria

Both Hamming distance and edit distance can be specified between the read and its target region on the reference sequence. Users can decide which mapping criteria to

use and the maximum mapped error number. By default, the mapping criteria that ZOOM adopted is allowing two mismatches. Any number of mismatches and one insertion/deletion of any length are allowed in ZOOM.

Parameter	**Short Description**
-mm <int>	The maximum mismatches allowed between the read and its target region on the reference sequence. Any number of mismatches is allowed as long as it is less than the length of reads. Example: ./ZOOM –i read1.fa –o read.output –g ref.fasta –mm 4
-ed <int>	The maximum edit distance allowed between reads and their target region on the reference sequence. Only one gap is allowed in ZOOM; however, the length of the indels is not restricted. The edit distance here means the summation of the number of mismatches and the length of the indels. Example: ./ZOOM –i read1.fa –o read.output –g ref.fasta –ed 3
-id <int>	If the length of the indels is assigned by the user, the combination of <-mm> and <-id> are required. <–id> means the maximum length of the indel. Example: If you want to find the target region of the reads with at most two mismatches and one indel with maximal length one, the following command is used: ./ZOOM –i read1.fa –o read.output –g ref.fasta –mm 2 –id 1

Options for Using Quality Scores

The quality scores are associated with the sequencing confidence of each nucleotide base in an Illumina/Solexa read. A low confidence score denotes low sequencing quality at that position in the read. Therefore, mismatches occurring at positions with high quality scores are more trusted than those at low quality scores positions. Quality

scores are required in ZOOM to enhance mapping results. When quality scores are available for use, ZOOM will only count the mismatches occurring on high quality score read positions. If sequencing quality scores are not included, all the bases in the reads will be considered as having a high confidence score. Sequencing quality scores can be also used to mark bases in the reads as wild cards, by providing <-rh> with a threshold quality score larger than the bases that you wish to use as wild cards. In our experiments, we observed that more reads were uniquely mapped when quality scores were included. In later release of ZOOM, assessment and chosen of alignment according to the quality score similar to MAQ will be incorporated.

Parameter	**Short Description**
-rh <rhthresh>	The value of this parameter specifies the threshold to differentiate the high quality bases from the low quality bases in the read. Nucleotide bases whose quality scores are beyond the threshold will be viewed as high quality bases. We recommend that the threshold should be defined according to the user's experience and ultimately trial and error. The use of this parameter also specifies that the quality scores will be used to enhance the mapping results. Example: ./ZOOM –i read.fa –o read.output –g ref.fa –rh 4 ./ZOOM –i read.fa –o read.output –g ref.fa –rh 8
-mhn <minhqnum>	The minimum number of high quality bases in the read that are required for a read to be mapped to the reference genome. The default value is 8 if –rh is assigned. Example: ./ZOOM –i read.fa –o read.output –g ref.fa –rh -5 –mhn 10

Special Options for ABI SOLiD Reads

Options for Mapping Criteria

In this release, ZOOM will map ABI SOLiD reads within a given number of errors, without indels. For ABI SOLiD reads, there are two types of errors:

- Genomic differences, which are the mismatches between the base space of the reads and its target region on the nucleotide of the reference sequence.

- The other type of error is an actual sequencing error on the color space read.

Parameter	**Short Description**
-mm <int>	The maximum number of mismatches allowed. Basically, the number of genomic differences and sequencing errors allowed. Any number of mismatches is allowed as long as it's less than the length of reads. Example: ./ZOOM –i read1.csfasta –o read.output –g ref.fasta –mm 2

Options for Decoding Mapped ABI Color Space Reads

Parameter	**Short Description**
-dcs	Option to decode color space reads into nucleotides. If this options is chosen, the genomic difference number and the sequencing error number of each mapped read will be shown in the decoded file. Please refer to the output section of this manual. Example: ./ZOOM –i read1.csfasta –o read.output –g ref.fasta –mm 2 –dcs The decoded reads will be reported in the file read.output.decode

In later release of ZOOM, assessment and chosen of alignment according to the quality score similar to MAQ will be incorporated.

Examples of Analyzing Data Using ZOOM

Usage examples for Illumina-Solexa data

1. ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.m2 –mm 6

   ZOOM maps reads in *BAC.fasta* to *chr6.fa*, allowing six mismatches between each read and its target region on *chr6.fa*. The mapping results are recorded in the output file *BAC.vs.chr6.m2*.

2. ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.rh4.m2 –mm 2 –rh 4

   ZOOM maps reads in *BAC.fasta* to *chr6.fa*, allowing two mismatches at the position where the base quality score is no less than four. The uniquely mapped reads are reported in *BAC.vs.chr6.rh4.m2*.

3. ZOOM –i BAC.fasta –g all.lst –o BAC.vs.all.m2 –ed 2

   ZOOM maps reads in *BAC.fasta* to all chromosomes listed in *all.lst*, allowing two edit distances.

4. ZOOM –i BAC.fasta –g chr6.fa –o BAC.vs.chr6.m2 –mm 2 –asbl –mc 3

   ZOOM maps reads in *BAC.fasta* to *chr6.fa*, allowing two mismatches and reconstructs a consensus sequencing using the uniquely mapped reads information. The uniquely mapped read information is reported in *BAC.vs.chr6.m2*. The coverage of each position of the consensus sequence is in the file *BAC.vs.chr6.m2.frequency*. The reconstructed consensus assembly is in FASTA format in the file *BAC.vs.chr6.m2.consensus*, where the bases whose coverage is less than three will be noted by lowercase letters.

5. ZOOM –i ChIP_Seq.fasta –g all.lst –o chip.vs.ref.m2 –mm 2 –asbl –mc 100

6. ZOOM –i Small_RNA.fasta –g all.lst –o RNA.vs.all.m2 –mm 2 –mk 100

   ZOOM maps reads in *Small_RNA.fasta* to all reference sequences listed in *all.lst*, allowing two mismatches and outputs the best 100 mapping results of each mapped read. The uniquely mapped reads will be recorded in the file *RNA.vs.all.m2*, while the top 100 results for each read is reported in the file *RNA.vs.all.m2.all*.

Usage examples for ABI SOLiD data

1. ZOOM –i Yoruban.15M.csfasta –g all.lst -mm 2

   ZOOM maps the color space reads in *Yoruban.15M.csfasta* to the human chromosomes listed in the file *all.lst*, allowing two errors (either two genome differences and no sequencing errors on color space read, or one genome difference and one color space sequencing error, or no genomic differences and two color space sequencing errors).

2. ZOOM –i A.15M.csfasta –g all.lst –mm 4 –de

   ZOOM maps the color space reads in *A.15M.csfasta* to the human chromosomes listed in the file *all.lst*, allowing four errors.

3. ZOOM –if Yoruban_F3.csfasta –ir Yoruban_R3.csfasta –pe –pemin 1495 –pemax 2027 –g all.lst –mm 2 –dcs

   ZOOM maps the color spaced paired-end reads in *Yoruban_F3.csfasta* and *Yoruban_R3.csfasta* to the human chromosomes listed in *all.lst*, allowing two errors. Two paired-end reads are mapped from 1495bp to 2027bp.

Chapter

4

Graphical User Interface

Please notice that ZOOM GUI part is not included in this release. This chapter is subject to change.

Chapter

5

Frequently Asked Questions

Question: Can I put reads of different lengths in a same file?

Answer: Yes, ZOOM will automatically call different parameter sets for different read lengths, and the results will be merged.

Question: Is the input read data case sensitive?

Answer: No, "a"= "A", "c"="C", "g"="G", "t"="T"="u"="U", and all other letters are "N". If you have different requirements, please contact us.

Question: Can I get all mapped positions for each read, besides the uniquely mapped information?

Answer: Yes, use the parameter <*-mk*> to output top N best mapping results for each read. Set N very large if you want all mapping results for each read.

Question: In which cases can ZOOM achieve 100% sensitivity?

Answer: ZOOM design a framework to construct the efficient spaced seeds sets which can achieve 100% sensitivity for a large range of read lengths and mismatch numbers. All cases in this release are listed in the section entitled "100% sensitivity cases". For cases with more mismatch numbers and cases with insertion/deletion, ZOOM also has good sensitivity. If you do need 100% sensitivity beyond the listed cases, please contact us.

Question: Can ZOOM find short indels?

Answer: Yes. However, ZOOM can only find one gap with any length. The speed is about five times slower than the mode only allowing mismatches when each indel is allowed.

Question: The quality of 3'-end reads is not so good, what should I do?

Answer: You can set a threshold between high quality bases and low quality bases using the "-rh" parameter, then ZOOM will neglect those low quality bases when mapping.

Question: What should I do if the read number is too large to fit in the RAM?

Answer: Please split the read file into several parts which could be fit in the RAM. If you have 8G RAM, 25 million reads in one file are recommended. If your memory doubles, then the number of reads in one file could double too. Run the several parts separately, and merge the mapping results into one result file. If you want to do assembly too, please use the option "–mii" to load in the merged result file to do assembly. If the data set is paired-end reads, please mate up the reads in one file and carry out the above operations.

Question: Can ZOOM output the structural variation according to the output of paired-end mapping?

Answer: Not yet. In this release, ZOOM only output those read pairs mapped on the right chains of the reference sequence in the right distance limit. Those reads mapped not in one pair are not reported. If you want to study those reads not mapped in one pair, please extract those reads unmapped in paired-end mode and map them again in single end mode. In latter release, we will consider finish this part automatically. If you have any suggestions on this part, please feel free to contact us. We'd like to make ZOOM suitable for you.

Question: Are there restrictions on the length of reads label?

Answer: No. But please don't use spaces inside the label for the 'one read per line' format, because ZOOM needs to know where the read data field begins.

Question: Can ZOOM deal with 454 data or Helico data?

Answer: ZOOM is optimized for Illumina/Solexa and ABI SOLiD data. ZOOM can get good mapping results on these two instruments. However, the sequencing error types of 454 instrument and Helico instrument are quite different, which contain much short indels. ZOOM can't guarantee good mapping results because now ZOOM can only handle one gap of any length other than many gaps. However, surely you could have a try since ZOOM can handle reads over 200bp and can deal with reads of variable lengths automatically. Any feedbacks are appreciating. We'd like to support these two instruments later.

Chapter

6

100% sensitivity cases

This chapter lists all cases where the current release of ZOOM can guarantee 100% sensitivity.

ZOOM designs a framework to construct the efficient spaced seeds sets which can achieve 100% sensitivity for a large range of read lengths and mismatch numbers. These spaced seeds sets guarantee great accuracy and speed of ZOOM. The cases that guarantee 100% sensitivity in this release are listed in the following table. For cases with more mismatch numbers and cases with insertion/deletion, ZOOM still has good sensitivity. If you do need 100% sensitivity beyond the listed cases, please contact us. We will design seeds specifically for your requirement.

For all the cases, you can allow more mismatches using "-mm" parameters and ONE gap whose length is assigned by "-id" or "-ed".

Cases for Illumina/Solexa data

Read Length range (bp)	Mismatch numbers
12-15	No more than 0
15-240	No more than 2
25-240	No more than 3
30-240	No more than 4
78-240	No more than 5
91-240	No more than 6

Cases for ABI SOLiD data

Since the data format of ABI SOLiD is color space, ZOOM extends the multiple spaced seeds set used for Illumina/Solexa data. Spaced seeds are used between the color space of reads and the color space of reference sequences. Note that in the following table, the mismatch number is the summation of the polymorphism number on base space and the sequencing error number on color space. However, since one polymorphism occurs on base space, there are two adjacent mismatches on the color space. So the mismatch number on the color space is in fact at most the summation of sequencing error number on color space and two times the polymorphism number on base space.

For example, if a read of length 50bp has four polymorphisms with its target region on the reference sequence, there are at most eight mismatches between the color space of this read and its target region.

For example, for reads of length 35, ZOOM will find all the mapping results which have:

1. Three polymorphisms on the base space. (in total, six mismatches between the color space of reads and the reference sequence)

2. Two polymorphisms on the base space and one sequencing error on the color space. (in total, total five mismatches between the color space of reads and reference sequence)

3. One polymorphism on the base space and two sequencing errors on the color space. (in total, four mismatches between the color space of reads and the reference sequence)

4. Zero polymorphisms on the base space and three sequencing errors on the color space. (in total, three mismatches between the color space of reads and the reference sequence)

Read Length range (bp)	Mismatch numbers
15-240	No more than 2
30-240	No more than 3
42-230	No more than 4

Chapter

7

About Bioinformatics Solutions Inc.

BSI *provides advanced software tools for analysis of biological data.*

Bioinformatics Solutions Inc. develops advanced algorithms based on innovative ideas and research, providing solutions to fundamental bioinformatics problems. This small, adaptable group is committed to serving the needs of pharmaceutical, biotechnological and academic scientists and to the progression of drug discovery research. The company, founded in 2000 in Waterloo, Canada, comprises a select group of talented, award-winning developers, scientists and sales people.

At BSI, groundbreaking research and customer focus go hand in hand on our journey towards excellent software solutions. We value an intellectual space that fosters learning and an understanding of current scientific knowledge. With an understanding of theory, we can focus our talents on providing solutions to difficult, otherwise unsolved problems that have resulted in research bottlenecks. At BSI, we are not satisfied with a solution that goes only partway to solving these problems; our solutions must offer something more than existing software.

The BSI team recognizes that real people will use our software tools. As such, we hold in principle that it is not enough to develop solely on theory; we must develop with customer needs in mind. We believe the only solution is one that incorporates quality and timely results, a satisfying product experience, customer support and two-way communication. So then, we value market research, development flexibility and company-wide collaboration, evolving our offerings to match the market/user's needs.

Efficient and concentrated research, development, customer focus and market analysis have produced: PEAKS software for protein and peptide identification from tandem mass spectrometry data, RAPTOR and PROSPECT Pro software for threading based 3D protein structure prediction and PatternHunter software for all types of homology search sequence comparison.

Chapter

8

ZOOM Software License

This is the same agreement presented on installation. It is provided here for reference only.

If we are evaluating a time limited trial version of ZOOM and we wish to update the software to the full version, we must purchase ZOOM and obtain a full version registration key.

1. License. Subject to the terms and conditions of this Agreement, Bioinformatics Solutions (BSI) grants to you (Licensee) a non-exclusive, perpetual, non-transferable, personal license to install, execute and use one copy of ZOOM (Software) on one single CPU at any one time. Licensee may use the Software for its internal business purposes only.

2. Ownership. The Software is a proprietary product of BSI and is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. BSI shall at all times own all right, title and interest in and to the Software, including all intellectual property rights therein. You shall not remove any copyright notice or other proprietary or restrictive notice or legend contained or included in the Software and you shall reproduce and copy all such information on all copies made hereunder, including such copies as may be necessary for archival or backup purposes.

3. Restrictions. Licensee may not use, reproduce, transmit, modify, adapt or translate the Software, in whole or in part, to others, except as otherwise permitted by this Agreement. Licensee may not reverse engineer, decompile, disassemble, or create derivative works based on the Software. Licensee may not use the Software in any manner whatsoever with the result that access to the Software may be obtained through the Internet including, without limitation, any web page. Licensee may not rent, lease, license, transfer, assign, sell or otherwise provide access to the Software, in whole or in part, on a temporary or permanent basis, except as otherwise permitted by this Agreement. Licensee may not alter, remove or cover proprietary notices in or on the Licensed Software, or storage media or use the Licensed Software in any unlawful manner whatsoever.

4. Limitation of Warranty. THE LICENSED SOFTWARE IS PROVIDED AS IS WITHOUT ANY WARRANTIES OR CONDITIONS OF ANY KIND, INCLUDING BUT NOT LIMITED TO WARRANTIES OR CONDITIONS OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. LICENSEE ASSUMES THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE LICENSED SOFTWARE.

5. Limitation of Liability. IN NO EVENT WILL LICENSOR OR ITS SUPPLIERS BE LIABLE TO LICENSEE FOR ANY INDIRECT, INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER, EVEN IF THE LICENSOR OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE OR CLAIM, OR IT IS FORESEEABLE. LICENSOR'S MAXIMUM AGGREGATE LIABILITY TO LICENSEE SHALL NOT EXCEED THE AMOUNT PAID BY LICENSEE FOR THE SOFTWARE. THE LIMITATIONS OF THIS SECTION SHALL APPLY WHETHER OR NOT THE ALLEGED BREACH OR DEFAULT IS A BREACH OF A FUNDAMENTAL CONDITION OR TERM.

6. Termination. This Agreement is effective until terminated. This Agreement will terminate immediately without notice if you fail to comply with any provision of this Agreement. Upon termination, you must destroy all copies of the Software. Provisions 2,5,6,7 and 10 shall survive any termination of this Agreement.

7. Export Controls. The Software is subject at all times to all applicable export control laws and regulations in force from time to time. You agree to comply strictly with all such laws and regulations and acknowledge that you have the responsibility to obtain all necessary licenses to export, re-export or import as may be required.

8. Assignment. Customer may assign Customer's rights under this Agreement to another party if the other party agrees to accept the terms of this Agreement, and Customer either transfer all copies of the Program and the Documentation, whether in printed or machine-readable form (including the original), to the other party, or Customer destroy any copies not transferred. Before such a transfer, Customer must deliver a hard copy of this Agreement to the recipient.

9. Maintenance and Support. BSI will provide technical support for a period of thirty (30) days from the date the Software is shipped to Licensee. Further maintenance and support is available to subscribers of BSI's Maintenance plan at BSI's then current rates. Technical support is available by phone, fax and email between the hours of 9 am and 5 pm, Eastern Time, excluding statutory holidays.

10. Governing Law. This Agreement shall be governed by and construed in accordance with the laws in force in the Province of Ontario and the laws of Canada applicable therein, without giving effect to conflict of law provisions and without giving effect to United Nations Convention on contracts for the International Sale of Goods.

Chapter

9

Reference: ZOOM Paper

Please use the following references when publishing a study that involved the use of ZOOM.

ZOOM! Zillions of Oligos Mapped. Hao Lin, Zefeng Zhang, Michael Q. Zhang, Bin Ma, and Ming Li. *Bioinformatics* 2008 24(21):2431-2437

For technical support issues not found in this manual, please contact either your Sales Representative or any of our support service resources:

eMail: support@bioinfor.com
Tel: (519) 885-8288
Fax: (519) 885-9075
Web: www.bioinfor.com

Bioinformatics Solutions Inc.
470 Weber Street North, Suite 204
Waterloo, Ontario, Canada
N2L 6J2

zillions of oligos mapped