Umap Documentation

Release 0.1.0

Mehran Karimzadeh, Carl Ernst, Anshul Kundaje, Michael M. Hoffm

Jun 20, 2016

CONTENTS

Contents:

CHAPTER

ONE

INTRODUCTION

The free umap software package efficiently identifies uniquely mappable regions of any genome. Its Bismap extension identifies mappability of the bisulfite converted genome (methylome).

the mappability of a genome for a given read length *k*. First, it generates all possible *k*-mers of the genome. Second, it maps these unique *k*-mers to the genome with Bowtie version 1.1.0. Third, Umap marks the start position of each *k*-mer that aligns to only one region in the genome. Umap repeats these steps for a range of different *k*-mers and stores the data of each chromosome in a binary vector *X* with the same length as the chromosome's sequence. For read length *k*, $X_i = 1$ means that the sequence starting at X_i and ending at X_{i+k} is uniquely mappable on the forward strand. Since we align to both strands of the genome, the reverse complement of this same sequence starting at X_{i+k} on the reverse strand is also uniquely mappable. $X_i = 0$ means that the sequence starting at X_i and ending at X_{i+k} can be mapped to at least two different regions in the genome.

CHAPTER

TWO

MAPPABILITY OF THE BISULFITE-CONVERTED GENOME

To identify the single-read mappability of a bisulfite-converted genome, we create two altered genome sequences. In the first sequence, we convert all cytosines to thymine (C $\rightarrow$ T). In the other sequence we convert all guanines to adenine (G $\rightarrow$ A). Our approach follows those of Bismark and BWA-meth. We convert the genome sequence this way because bisulfite treatment converts un-methylated cytosine to uracil which is read as thymine. Similarly the guanine that is base-pairing with the un-methylated cytosine in reverse strand converts to adenine. However these two conversions never occur at the same time on the same read. We identify the uniquely mappable regions of these two genomes separately, and then combine the data to represent the single-read mappability of the forward and reverse strands in the bisulfite-converted genome.

Bismap requires special handling of reverse complementation of C $\rightarrow$ T or G $\rightarrow$ A converted genomes. Conversion of C $\rightarrow$ T on the sequence AATTCCGG produces AATT **TT** GG. In the Bowtie index, the reverse complement would be CCAAAATT. However for the purpose of identifying the mappability of the bisulfite-converted genome, we expect the reverse complement to be TTGGAA **TT**. The reason is that both forward and reverse strands undergo bisulfite treatment simultaneously. There is no DNA replication after bisulfite treatment. To handle this issue, Bismap creates its own reverse complemented chromosomes and suppresses Bowtie's usual reverse complement mapping.

Umap and Bismap each take approximately 200 CPU hours to run for a given read length. This can be parallelized in a computing cluster over 400 cores to take only 30 minutes.

2.1 Measures of mappability

Umap efficiently identifies the single-read mappability of any genome for a range of sequencing read lengths. The single-read mappability of a genomic region is a fraction of that region which overlaps with at least one uniquely mappable *k*-mer. The Bismap extension of Umap produces the single-read mappability of a bisulfite-converted genome. Both Umap and Bismap produce an integer vector for each chromosome that efficiently defines the mappability for any region and can be converted to a browser extensible data (BED) file. In addition to single-read mappability, we can measure the mappability of a genomic region by another approach. To quantify the single-read mappability of a given genomic region, we measure the fraction of potential uniquely mappable reads in that region. A region, however, can have 100% single-read mappability, but in practice require a high coverage sequencing to properly identify that region. For example, a 1 kbp region with 100% single-read mappability can be mappable due to a minimum of 10 unique 100-mers that none of them overlap or a maximum of 1100 unique 100-mers that highly overlap. Therefore, we define the multi-read mappability, the probability that a randomly selected read of length k in a given region is uniquely mappable. For the genomic region $G_{i:j}$ starting at *i* and ending at *j*, there are $j-i+k+1$ *k*-mers that overlap with $G_{i:j}$. The multi-read mappability of $G_{i:j}$ is the fraction of those *k*-mers that are uniquely mappable.

CHAPTER

THREE

QUICK START

We have tested Umap installation on a CentOS system using python 2.7.11. Bismap requires numpy and pandas and it uses other python modules such as:

- gzip
- os
- re
- subprocess

Umap uses mercurial version control. Make sure that mercurial (hg) is installed. Download Umap to the directory of your python packages using:

```
hg clone https://bitbucket.org/hoffmanlab/umap
cd umap
python setup.py install
```

Now we will run a test using a wrapper in the umap directory called ubismap.py and a toy genome stored under umap/data

```
cd umap
python ubismap.py data/genome.fa data/chrsize.tsv data/TestGenomeMappability all.q $BOWTIRDIR/bowtie-
sh test_run.sh
```

The scripts that are produced by **ubismap.py** assume that you are using a Sun Grid Engine computing cluster. You can use parameters of this script to adjust it to your own system. You may need to manually edit this file because many of the SGE settings are very different than other computing clusters. However, all of the Umap modules accept *-job_id* which allows you to use the modules without a cluster or if your cluster does not support job arrays.

Basically, the job array saves an environmental variable (defined by (*-var_id*) that Umap uses for paralellizing processes. You can run the modules in a for loop and set the *-job_id* manually. For example, in order to find *k*-mers of a genome with 10 million base pairs, the get_kmers and run_bowtie modules each need to be executed with -job_ids ranging between 1 to 10.

CHAPTER

FOUR

GET *K*-MERS

class umap.**GetKmers**(*out_dir*, *kmer*, *job_id*, *chr_dir*, *chrsize_path*)

CHAPTER

FIVE

RUN BOWTIE

class umap.**BowtieWrapper**(*kmer_dir*, *bowtie_dir*, *index_dir*, *index_name*, *job_id*)

CHAPTER

SIX

MERGE BOWTIE OUTPUTS

class umap.**UnifyBowtie**(*bowtie_outdir*, *chrsize_path*, *job_id*)

CHAPTER

SEVEN

MERGE DATA OF VARIOUS *K*-MERS

class umap.**CombineUmaps**(*kmer_dir*, *chrsize_path*, *out_dir*, *job_id*, *kmer_dir_2*)

CHAPTER

EIGHT

CONVERT NUMERIC VECTORS TO BED AND WIGGLE

class umap.**Int8Handler**(*in_dir*, *out_dir*, *C2T*, *G2A*, *chrsize_path*)

Int8Handler.**write_beds**()

Int8Handler.**write_as_wig**()

CHAPTER

NINE

REQUESTING GENOMES

In case you need these data for other genomes and do not have access to a Sun Grid Engine computing cluster, we may accept to do this for you. Please contact the software maintainer by email and we will do our best to assist you as soon as possible.

CHAPTER

TEN

CONTACT, SUPPORT AND QUESTIONS

For support of Umap, please user our mailing list. Specifically, if you want to report a bug or request a feature, please do so using the Umap issue tracker. We are interested in all comments on the package, and the ease of use of installation and documentation.

CHAPTER

ELEVEN

INDICES AND TABLES

- genindex
- modindex
- search

u

B

C

G

I

U

W