Manual for CAM

Manual Content

Description

CAM is a QC pipeline for MNase-seq data. By applying this pipeline, CAM provides multiple informative QC measurements and nucleosome organization profiles on potentially functionally related regions for the given MNase-seq data. CAM also includes 268 historical MNase-seq datasets from human and mouse as a reference atlas for unbiased assessment. Here, we provide a detailed manual regarding the installation and usage of CAM. See the FAQ section if this section does not answer your questions.

Installation

Prerequire

1. python(version = 2.7)
2. R (version >= 2.14.1)
3. CAM will generate a summary QC report if you have pdflatex installed, otherwise you only get a package of QC plots and analysis results in the summary folder.

Install CAM

CAM uses Python's distutils tools for source installations. To install the CAM source distribution, unpack the distribution zip file and open up a command terminal. Go to the directory where you unpacked CAM and simply run the install script. We provide an example installation in the quick start section for users who are confused about the following description:

```
$ python setup.py install
```

By default, the script will install the python library and executable codes globally, which means you need to be the root or administrator of the machine to complete the installation. Please contact the administrator of the machine if you need help. If you need to provide a nonstandard install prefix or any other nonstandard options, you can provide many command line options to the install script. Use the --help option to see a brief list of available options.

```
$ python setup.py install --prefix /home/CAM
```

If you are not the root user, you will see “permission denied” when attempt to install CAM globally. Then, you can use --prefix parameter to install CAM to any directory in which you have write permission. You might need to add the install location to your PYTHONPATH and PATH environment variables. The process varies for each platform, but the general concept is the same across platforms.

Setup environment variables

PYTHONPATH

To set up your PYTHONPATH environment variable, you will need to add the value PREFIX/lib/pythonX.Y/site-packages to your existing PYTHONPATH. In this value, X.Y represents the major–minor version of Python you are using (for CAM it should be 2.7; you can find this with sys.version[:3] from a Python command line). PREFIX is the install prefix where CAM is installed. If you did not specify a prefix on the command line, CAM will be installed using the default Python system prefix. For example, if you specify the parameter "--prefix /home/CAM", you must add /home/CAM/lib/python2.7/site-packages to your PYTHONPATH (below is an example, DO NOT add spaces around the equal sign).

```
$ export PYTHONPATH=/home/CAM/lib/python2.7/site-packages:$PYTHONPATH
```

PATH

Similar to PYTHONPATH, you will also need to add a new value to your PATH environment variable to use the CAM command line directly. Unlike the PYTHONPATH value, you will need to add PREFIX/bin to your PATH environment variable. The process for updating this information is the same as described above for the PYTHONPATH variable:

```
$ export PATH=/home/CAM/bin:$PATH
```

To check your default PATH and PYTHONPATH, you can type:

```
$ echo $PATH

$ echo $PYTHONPATH
```

Automatically export PATH and PYTHONPATH

Does exporting the environmental variables every time bother you?

In Linux, you can include the new values in your PYTHONPATH using bash by adding these lines:

```
$ export PATH=/home/CAM/bin:$PATH

$ export PYTHONPATH=/home/CAM/lib/python2.7/site-packages:$PYTHONPATH
```

to your ~/.bashrc (or ~/.bash_profile). Then, the environment is automatically exported, and you do not need to export the environment in the future.

Now you have CAM installed on your computer. Type

```
$ CAM.py --help
```

to check your installation.

Prepare genome sequence file

To conduct CAM, a genome sequence file in fasta (.fa) or 2bit (.2bit) is the ONLY required file. The genome sequence file is used for both the mapping step (optional) and the QC components, and can be downloaded from the UCSC genome browser http://genome.ucsc.edu/cgi-bin/hgTables. Below is an example of downloading the genome sequence file of the hg19 genome version from the UCSC genome browser.

We also provide download link in the Quick Start section.

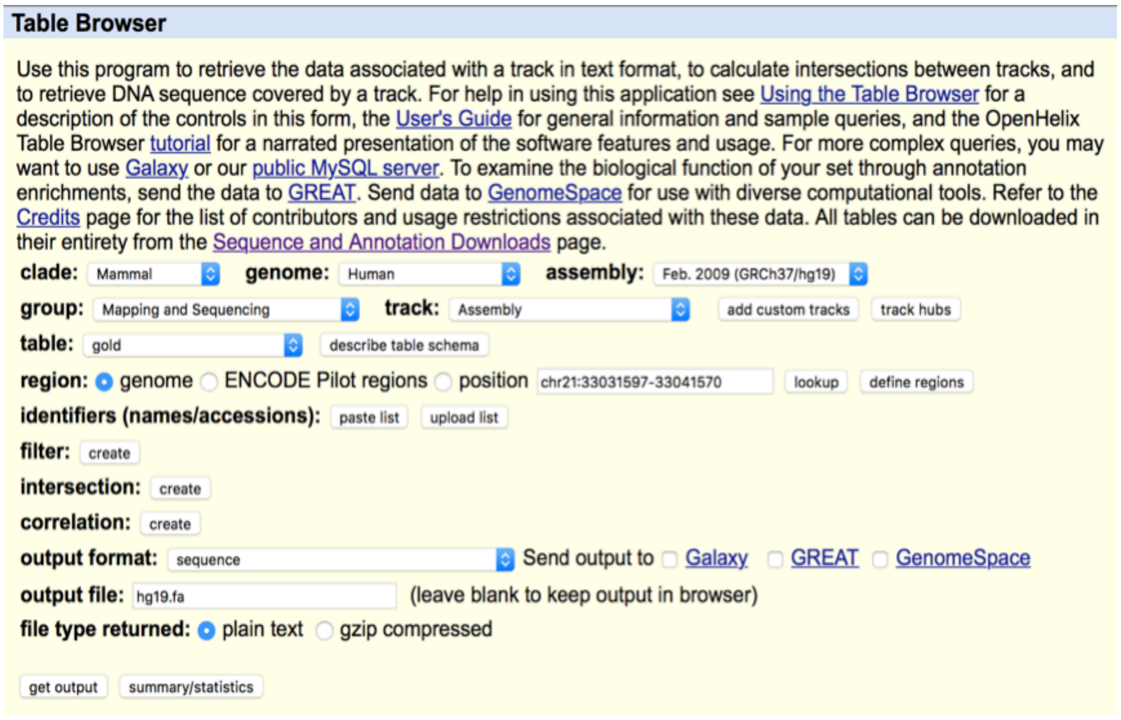

Table Browser

Use this program to retrieve the data associated with a track in text format, to calculate intersections between tracks, and to retrieve DNA sequence covered by a track. For help in using this application see Using the Table Browser for a description of the controls in this form, the User's Guide for general information and sample queries, and the OpenHelix Table Browser tutorial for a narrated presentation of the software features and usage. For more complex queries, you may want to use Galaxy or our public MySQL server. To examine the biological function of your set through annotation enrichments, send the data to GREAT. Send data to GenomeSpace for use with diverse computational tools. Refer to the Credits page for the list of contributors and usage restrictions associated with these data. All tables can be downloaded in their entirety from the Sequence and Annotation Downloads page.

clade: Mammal genome: Human assembly: Feb. 2009 (GRCh37/hg19)
group: Mapping and Sequencing track: Assembly
table: gold
region: [✓ genome] ENCODE Pilot regions position chr21:33031597-33041570
identifiers (names/accessions):
filter:
intersection:
correlation:
output format: sequence Send output to Galaxy GREAT GenomeSpace
output file: hg19.fa (leave blank to keep output in browser)
file type returned: [✓ plain text] gzip compressed

Prepare bowtie mapping index files

This step is not required for CAM but can save processing time, especially when users have many MNase-seq samples to process. Users can build mapping index in advance, and input mapping index to CAM to avoid the step of generating mapping index, which is time consuming.

The parameter "--mapindex" should be the absolute path of index filename prefix (without trailing .X.ebwt). This parameter will be directly used as the mapping index parameter of bowtie.

eg: /home/user/bowtie_index/hg19 (then under your folder /home/user/bowtie_index/ there should be hg19.1.ebwt, hg19.rev.1.ebwt ....).

CAM will generate bowtie index files in the annotation/ folder if the user run CAM with a genome sequence file in .fa format (eg. --fa hg19.fa) (turn off --clean parameter). User can backup the index generated and input to --mapindex in the next run.

Install pdflatex

MacOS users can get download "MacTex" from http://www.tug.org/mactex/. After downloading the package MacTex.pkg from http://tug.org/cgi-bin/mactex-download/MacTeX.pkg, double click to install MacTex and get the pdflatex.

For linux users, you can type

```
$ apt-get install texlive-all
```

to install pdflatex on your server/computer.

Note that the installation of pdflatex on both the MacOS and Linux requires root privileges.

CAM Usage

An example mentioned in the Quick Start section

Now you can use the full functions of CAM, and the mapping index will be built automatically. See the following example with the CAM simple mode (see the example in the Quick Start section and the following section for the usage of the simple mode):

```
$ CAM.py simple -a GSM907784_1.fastq -b GSM907784_2.fastq -n GSM907784 -t PE -s
hg19 --fa /home/data/hg19.fa -c /home/data/CTCF_motif_hg19.bed
```

# --fa /home/data/hg19.fa: absolute path of genome sequence file. Note that the genome version (-s hg19) should corresponded to the genome sequence (--fa /home/data/hg19.fa) file. The genome sequence file can be downloaded in the Download section.

# -c /home/data/CTCF_motif_hg19.bed: an example for custom regions in hg19 version. User can generate nucleosome profile on the custom region by specify this parameter. The example can be downloaded in the Download section.

Simple mode and Standard mode

CAM provides two modes. You can edit and design a parameter complex suitable for your data with the standard mode, but you need to generate a configure file and edit it before you start CAM. The simple mode is designed for convenience and a quick start. With this mode, you can run CAM with a simple command line, but only major parameters can be edited.

Simple mode

You may become familiar with the simple mode in the Quick Start section. In simple mode, you only need to specify the input data, output name, genome sequence, and species (genome version) with a command line. CAM will generate the QC and analysis results with all default parameters.

All parameters in simple mode are also in standard mode and are described below:

```
$ CAM.py simple [-h] -a INPUTA [-b INPUTB] -n NAME -s SPECIES -t {SE,PE} --fa
GENOMESEQUENCE [-c CUSTOMREGION] [-p THREAD] [-f] [--clean]
```

Type

```
$ CAM.py simple -h
```

for a detailed description of all major parameters of the simple mode.

Standard mode

In standard mode you can edit all parameters and make CAM suitable for your data. First you should generate a configure file by typing:

```
$ CAM.py gen your_config_name.conf
```

Now you have generated a configure file (named your_config_name.conf in the example). If you open the generated configure file, you will find all of the changeable parameters listed with the pattern "parameter = value" (for example, inputa = inputfile_1.fastq) and corresponding description at the bottom of each panel (starts with "#").

We use a Python package called “ConfigParser" to parse the configure file, which requires a specific pattern of the input configure file. Thus, user should be cautious when editing the configure file:
1. Do NOT edit the description lines (the lines start with "#"), these lines should start with "#"
2. Keep the pattern "parameter = value" for the parameter lines. Do NOT remove the value for any parameter line if you do not have a specific value for the line. The space next to the equal sign should be retained.

Now, you can edit your configure file and make it suitable for your MNase-seq data (If you don't know the suitable parameter at first time, try default parameters).

After you complete editing of the configure file, you can use standard mode to run CAM with your edited configure file as input.

```
CAM.py run -c your_edit_config.conf -f --clean
```

# -c your_edited_config.conf: name of your input configure file. The configure file should be generated from CAM gen mode or copied from the output of the other CAM run mode (described in the following sections). The major parameters (parameter with “[required]” in the description line) include the input file location, species, genome sequence, sequencing type and output name.
# -f: force_overwrite, CAM will overwrite the output result if the output folder already exists when -f is added. Otherwise, CAM will exit.
# --clean: CAM will remove the intermediate result if --clean is added.

DIY template configure file to save your specific parameters

If editing the configure file for the same parameters for each CAM run is too complicated, the template configure file can be edited in the CAM package prior to installation to save changes to some major parameters (e.g., edit species, genome_fasta and other parameters) in the template configure file. If you previously installed CAM, you can edit the template configure file and reinstall the program.

First, you should find the template configure file, which is located in the “lib/Config/” folder of the CAM package and named “CAM_template.conf” (do NOT change the file name of the template configure file).

After you find the template configure file, you can edit it to change some commonly used parameters. For example, if you change the parameter “species = hg19” in the template configure file, the configure file you generate next time will always have “species = hg19” and does not need to be reedited. Generally, we suggest editing the species and genome_fasta. You can also change other parameters for specific usages (after you know the usage of each parameter from the parameter section). The method used to edit the template configure file is similar to the method used to edit the generated configure file.

Both standard mode and simple mode can inherit the edits to the template configure file.

Full parameter description

Below is a description of all changeable parameters. The value after the equal sign is the default (suggested) value of the parameter. [required] means this parameter should be specified for different samples.
A more convenient method is to read the same version in configure file while you are editing configure file and specify the parameters.

General

# In General panel we describe major parameters of CAM
inputa =
inputb =
# [required]inputa: (ABSOLUTE path)main input file of CAM, accept FASTQ (raw MNase-seq data), SAM (bowtie alignment output) and BED (alignment output transformed from sam, 6 column, 5th column is bowtie map quality and 6th column is strand) format, format is decided by the extension of input file(.fastq, .sam or .bed)

# inputb: (ABSOLUTE path)This parameter is for the 2nd part of paired-end FASTQ files. Required only if input file is FASTQ file and sequencing type is paired-end sequencing, only accept raw FASTQ file input. The extension should be .fastq. Note that if your data in paired end. User should also specify "seqtype"(below) as PE

seqtype = SE

# [required]seqtype: sequencing type of your sample, choose from SE(single end, default) and PE(paired end). Note that if you specify PE in seqtype and your data is FASTQ, you should give parameter to inputb. If your data is in SAM/BED (alignment) format, you don't need to specify inputb (it will be ignored if specified).

outputdirectory =

# [required]outputdirectory: (ABSOLUTE path) Directory for all result. Default is current dir "." (If was left blank) , but not recommended.

outname =

# [required]outname: Name of all your output results, your results will looks like outname.pdf, outname.txt

customregion =

# [optional]customregion: (ABSOLUTE path) user defined custom region to display nucleosome pattern.

# Custom region should be in absolute path, bed format and suffix should be .bed (e.g. /home/user/customregion/XXX.bed)

# Custom region require a 3 column bed or 6 column bed file, if it contains 6 or more columns, the 6th column should be strand (+/-), in that case CAM will display strand specific signal (e.g. flip signal for - strand).

# Limit of custom region is 50k, if the number of custom region is greater than 50k, CAM will take top 50k regions for nucleosome profile

# For example, user can input a list of motif sites (e.g. CTCF motifs) to check nucleosome pattern on certain motifs.

# Left this parameter empty if you don't need this function

Step1_preprocess

# In preprocess panel we describe parameters related to data process and transformation

mapping_p = 8

```
# mapping_p: Number of alignment threads to launch alignment software
genome_fasta = /home/user/genomefasta/hg19.fa
# genome_fasta: (ABSOLUTE path) whole genome sequence in FASTA format,
# The genome sequence should be in fasta format or 2bit format, genome_fasta file
should be ends with .fa or .2bit and the genome version should be corresponding with -
s species. This is the ONLY annotation file required for CAM
mapindex = /home/user/bowtie_index/hg19
# (ABSOLUTE path) mapindex: Mapping index of bowtie, absolute path,
# Users can build mapping index in advance, and input mapping index to CAM to avoid
generating mapping index step, which is time consuming.
# bowtie mapindex should be absolute path of index filename prefix (without
trailing .X.ebwt). this parameter will be directly used as the mapping index parameter
of bowtie
# eg: --mapindex /home/user/bowtie_index/hg19 (then under your folder
/home/user/bowtie_index/ there should be hg19.1.ebwt, hg19.rev.1.ebwt .... ),
species =
# species: you can input species name instead of download and specify
gene_annotation & genome_length files. CAM can use build-in annotation
files(gene_annotation, geonme_length) instead of user specified annotation files
# Note: by default CAM support hg19, hg38, mm9, mm10 as input for species
# Note: users can add custom species name, to do this, users can prepare gene
annotation file and genome length file and add them into build-in folder, CAM can find
corresponded annotation file in build-in packages.
# ***NOTE: CAM will ignore gene_annotation and genome_length if this parameter is
set, no matter in template configure file or cmd line.
q30filter = 1
# q30filter: Use q30 criteria (Phred quality scores) to filter reads in SAM/BED files, set
this parameter to 1(default) to turn on this option, set 0 to turn off. Default is 0 (turn
off)
trim3end = 0
# trim3end: trim N bps from 3'end of each reads. Users can set e.g. 30bp to trim 30bp
from 3'end of each reads.
# It takes effect when sequencing reads is too long (e.g. longer than library size).
```

# For longer reads, trim3end parameter can help trim adapter to improve mappability. Default: 0 (trim 0 bp == turn off).

# Note that for sam/bed input this parameter is ignored.

# CAM require fastq reads length consistent in same sample.

# Read_length - trim3nd should be greater than 18bp (at least 18bp should left for mapping, called lengh filter in following description).

# If trim3end == 0 (trun off), length filter is also close.

fragment_length_limit = 250

# [only works for PE data] fragment_length_limit: limit of fragment length. This parameter works for both bowtie mapping (-X, for FASTQ input) and mappable fragment length selection (for SAM/BED input)

rpm = 1

# rpm: normalize signal in profile bigwig to reads per million (RPM). set 1 to turn on (default) this function and set 0 to turn off

Step2_QC

# In QC panel we describe parameters related to QC and analysis measurements

sample_down_reads = 10000000

# sample_down_reads: randomly sample down 10M(default) reads to save time for some time consuming QC step, e.g. reads quality, nucleosome relationship, nucleosome length distribution

# You can change the sample down reads number by modify this parameter, also you can turn off sample down function by leave this parameter blank (time consuming~!).

upstreamtss = 1000

downstreamtss = 2000

# upstreamTSS/downstreamTSS: up/down stream distance to TSS for calculating reads TSS profile,

# By default we suggest 1000bp upstream (left in the aggregate plot) and 2000bp downstream (right in the aggregate plot), the limitation is 500bp up/down stream

reads_distance_range = 250

# reads_distance_range: for single end data, this parameter represent range of distance from each original reads 5'end to all downstream 3'end,

# For paired end data, this parameter represent range of fragment length, and should

be same as "fragment_length_limit" in step1, default is 250bp

customregion_dis = 1000

# customregion_dis: distance to the center of each custom region to plot nucleosome profile, by default its 1000. User can modify this parameter but should be between 200 ~ 5000

# Note: this parameter will be ignored if customregion is not specified.

Step3_nucarray

# In nucarray panel we describe parameters related to detecting well-positioned nucleosome arrays

window_size = 70

# window_size: length(bp) of sliding window for Gaussian smoothing, 70 (default) means sliding window size is +-70 = 140bp

# Note: nucarray use wiggle track in 10bp step, so window size can only be 50, 60, and 70...

smooth_bandwidth = 30

# smooth_bandwidth: standard deviation (bps) for Gaussian smoothing, default is 30. See more detail in document.

array_fold = 2

# array_fold: fold change of array position score vs. background on each array, user can filter arrays by change this cutoff

array_length = 600

# array_length: minimum length of detected nucarray. CAM will discard all arrays with shorter length. By default its 600bp.