PING: Probabilistic Inference for Nucleosome Positioning with MNase-based or Sonicated Short-read Data.

Xuekui Zhang[*]and Raphael Gottardo[†]

September 6, 2012

A step-by-step guide to inferring nucleosome positioning with high-throughput sequencing data using the PING package in R.

Contents

[*]ubcxzhang@gmail.com

[†]rgottard@fhcrc.org

1 Licensing and citing

Under the Artistic License 2.0, you are free to use and redistribute this software.

If you use this package for a publication, we would ask you to cite the following:

- Xuekui Zhang, Gordon Robertson, Sangsoon Woo, Brad G. Hoffman, and Raphael Gottardo. (2012). Probabilistic Inference for Nucleosome Positioning with MNase-based or Sonicated Short-read Data. PLoS ONE 7(2): e32095.

2 Introduction

The structural unit for chromatin packaging is the nucleosome, which is composed of approximately 147 bps of DNA wrapped around a core histone octamer. Nucleosome-associated DNA is less accessible to regulatory proteins like transcription factors, and nucleosome positioning, as well as histone modifications and histone variants (e.g. H2A.Z, H3.3), are therefore influential in cellular processes that depend on chromatin accessibility. Because nucleosome positions depend on cellular processes as well as intrinsic factors (e.g. DNA sequence), understanding how these positions influence cell states can require determining nucleosome positions within individual genomic regions.

Currently, genome-wide nucleosome-based data are typically generated by high-throughput single-end short-read sequencing of DNA obtained by either MNase digestion (MNase-seq), or chromatin immunoprecipitation (ChIP-seq) of MNase-digested or sonicated DNA. MNase digests linker DNA with relatively high specificity, and this specificity is reflected in the narrow spatial distribution of aligned reads. However, sonication protocols are widely used; for example, in work to identify classes of functional genomic regions by integrated analysis of diverse sets of short-read sequence data.

PING is a probabilistic method that extends PICS (Zhang et al. Biometrics, 2011, 67(1):151-63), which was developed for ChIP-seq data for transcription factors. PING infers nucleosome positions from either MNase-digested or sonicated nucleosome-based short-read data. Like PICS, PING models bi-directional read densities, uses mixture models, and imputes missing reads; however, it uses a new prior specification for the spatial positioning of nucleosomes, has different model selection criteria, model parameters, and different post-processing for estimated parameters. In addition, PING also includes novel statistical methods to identify nucleosomes whose read densities are lower than those of neighboring nucleosomes.

3 PING analysis steps

A typical PING analysis consists of the following steps:

1. Convert aligned-read data to a 'GRanges' list for efficient processing
2. Segment the genome into candidate regions that have sufficient aligned reads via 'segmentPING'
3. Estimate nucleosome positions and other parameters with PING
4. Post-process PING predictions to correct certain predictions

As with any R package, you should first load it with the following command:

```
> library(PING)
```

4 Data Input and Formatting

The first step of the PING pipeline consists of converting the files representing aligned reads for IP and optional Control samples into a format that allows efficient segmentation of the genome into a set of candidate regions, each of which has at least a threshold number of Forward and Reverse reads. The data format could be a BED type dataframe, or an 'AlignedReads' object returned by the 'ShortRead' package's 'ReadAligned' function, which can read various file formats including Eland, MAQ, Bowtie, etc. Please refer to the 'ShortRead' package vignette for details. In the example listed below, we use tab-delimited BED type files, in which each line represents a single read, and has the following columns: space, start, end, strand.

In addition to the IP and Control datasets, we use an another datatype that represents a read-mappability profile for the genome. As explained in the PICS paper, for each chromosome, a mappability profile for a specific read length consists of a vector of zeros and ones that gives an estimated read mappability 'score' for each base pair in the chromosome. A score of one at a position means that we should be able to align a read of that length uniquely at that position, while a score of zero indicates that no read of that length should be uniquely alignable at that position. In ChIP-seq work, reads that cannot be mapped to unique genomic locations are typically discarded. For convenience, and because transitions between mappable and non-mappable regions are typically much shorter than these regions, we compactly summarize each chromosome's mappability profile as a set of non-mappable intervals in a BED-format file. PING makes use of such mappability profiles to estimate the reads that are missing because they could not be uniquely aligned. Mappability profiles for a range of read lengths and species are available from

`http:/wiki.rglab.org/index.php?title=Public:Mappability_Profile`

Once the data have been read in, we can create a list of 'GRanges' objects, in which each element of the list corresponds to a different chromosome, and contains the aligned IP and Control reads, and the mappability profile, for the corresponding chromosome. When reading the data, reads are automatically sorted, which is required for a PING analysis. As well, we recommend that the number of duplicates be controlled by using PING's default protocol-specific thresholds.

We supply a demonstration set of MNase-seq data for chr1 from budding yeast GSM351492, NOCL_R4 (Kaplan et al, Nature, 2009, 458(7236):362-6). The demonstration data is in the 'extdata' folder of the PING installation: .../PING/inst/extdata.

```
> path <- system.file("extdata", package = "PING")
> dataIP <- read.table(file.path(path, "GSM351492_R4_chr1.bed"),
+     header = TRUE)
> dataIP <- as(dataIP, "GRanges")
```

5 PING analysis

5.1 Genome segmentation

We segment the genome by pre-processing aligned read data from a single-end ChIP-Seq experiment to detect 'candidate regions' that have a minimum number of forward and reverse reads. Each candidate region will then be processed separately by PING. The segmentation 'minReads' parameter can heavily influence the number of candidate regions returned. In principle, it is better to use a small value for 'minReads' in order not to miss any nucleosomes. However, too small a value will likely return many false positive regions, which can result in a longer computing time for the `PING` function.

If users are not sure what values to use, they can assign ‘minReads=NULL’, so that the value will be estimated from data. See below for an illustration.

In order to improve the computational efficiency of the PING package, if you have access to multiple cores we recommend that you do parallel computations via the `parallel` package. In what follows, we assume that `parallel` is installed on your machine. If it is not, you could omit the first line, and calculations will occur on a single CPU. By default the command is not run. Note that the `segmentPING` and `PING` functions will automatically detect whether you have initialized a cluster and will use it if you have.

```
> library(parallel)
```

```
> seg <- segmentPING(dataIP, minReads = NULL, maxLregion = 1200,
+     minLregion = 80, jitter = TRUE)
```

```
Performing segmentation for single end reads
[1] "We automatically calculated minReads, which is 5."
maxstep=450
step=2
width=150
minReads=5
```

5.2 Parameter estimation

After having segmented the genome into candidate regions, we use the `PING` function to probabilistically detect positioned nucleosomes. The function returns position estimates, confidence intervals, etc. In our case, assuming that we have already segmented the genome using `segmentPING`, we can proceed with the following command:

```
> ping <- PING(seg)
```

6 Post-processing PING results

Given the variation in nucleosome-based short-read data, some of PING’s predictions may be inaccurate, and we need to detect and resolve such problematic cases. For example, a nucleosome that is supported by relatively strong aligned-read signals may be adjacent to nucleosomes that have weaker signals. In such a case PING might fit two forward mixture components and one reverse mixture component, which will cause a mismatch of forward/reverse peaks of other nucleosomes. We can detect such a problem by screening for abnormal estimated δ values. As another example, a few adjacent nucleosomes with weak signals might be described by one wide forward/reverse peak. To detect such problems, we can screen for two large estimations of σ_f or σ_r. When we detect candidate regions with such problems, we reanalyze them with stronger prior on δ, σ_f, or σ_r to penalize atypical estimations.

When initial segmentation returns too long a candidate region, recursive cutting is automatically applied to split the regions into smaller ones. However, such splitting can result in two nucleosome predictions that are too close to each other on each side of the cutting boundary. Such cases may be predictions of the same nucleosome in the adjacent candidate regions. To address this, we merge their reads and refit the model, and let the model selection method decide whether these two predictions are the same nucleosome.

We post-process results with the following command:

```
> PS = postPING(ping, seg)
```

```
[1] "No regions with pingerror"

 The 10 Regions with following IDs are reprocessed for atypical delta:
[1] 144 120  22 162 266  76

 The 8 Peaks with following IDs are reprocessed for atypical sigma:
[1] 620 622 628 657 728 774

 The 12 regions with following IDs are reprocessed for Boundary problems:
[1] 166 183 333 351 409 662
```

The result output PS is a dataframe that contains estimated parameters of each nucleosome, users can use write.table command to export the selected columns of the result.

7 Result output

7.1 Exporting the results

To facilitate data processing, we make use of the `IRanges` package to summarize our results as `RangedData` objects and prepare them for export with different formats.

The function makeRangedDataOutput take a pingList or the data.frame returned postPING and a type of file as arguments, as well as an optional list of filters. The 'fixed' type is of bed format with a fixed size for the predicted nucleosome while the 'bed' type will use the predicted delta to infer the ranges.

```
> rdBed <- makeRangedDataOutput(ping, type = "bed")
> rdFix <- makeRangedDataOutput(PS, type = "fixed")
```

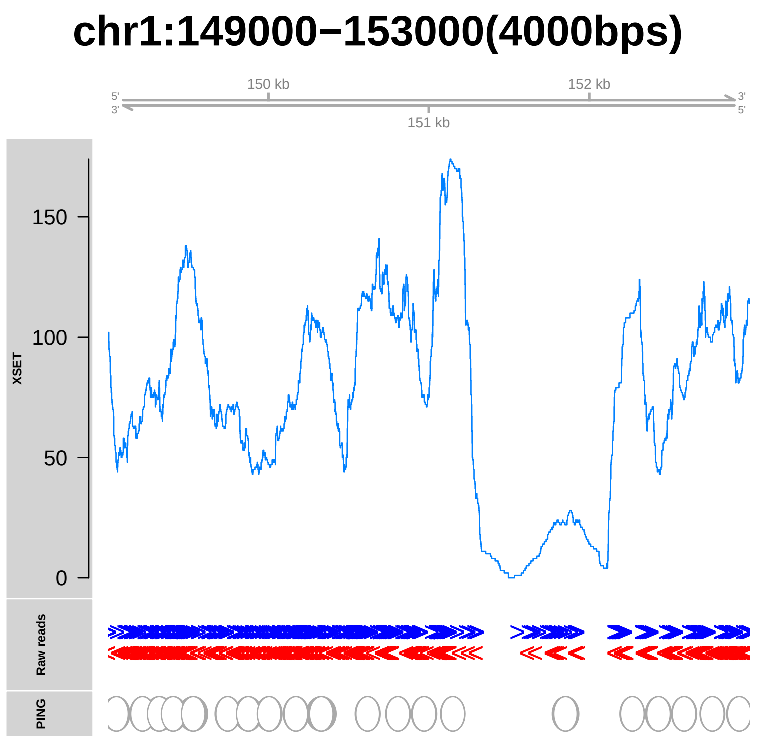
It returns a RangedData object that can be written into a file using the export function from rtracklayer

```
> library(rtracklayer)
> export(rdBed, file = "ping.bed")
> export(rdFix, file = "postPING.bed")
```

7.2 Visualizing the results

PING also offer tools to summarize the prediction for given ranges in a graph using the Gviz package.

```
> plotSummary(PS, dataIP, "chr1", "gen", from = 149000, to = 153000)
```



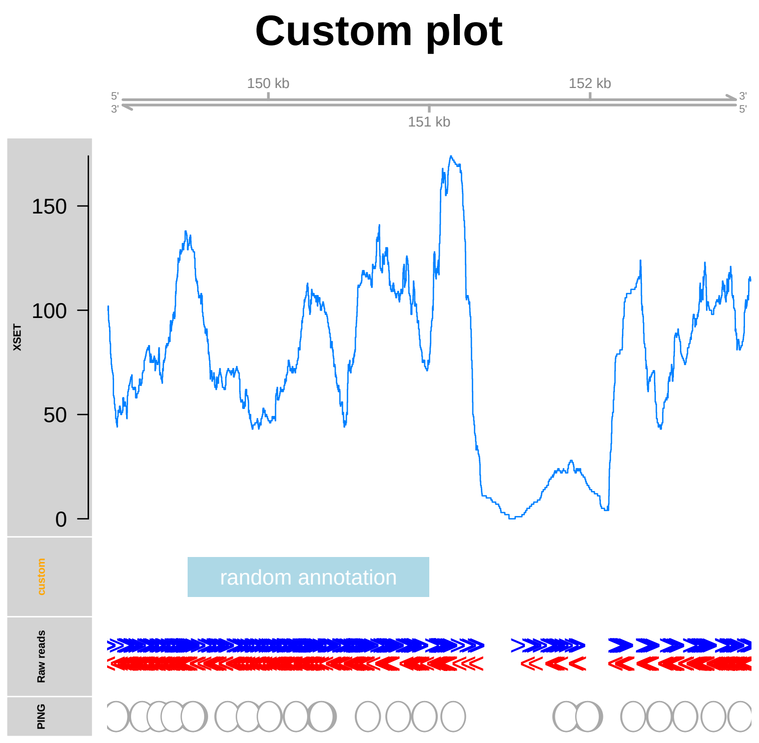
The plot shows the coverage of the reads used as input, the start position of the forward and reverse reads and the predicted position of the nucleosome. The gray border around each nucleosome represents the standard error associated with its position.

Alternately, users experienced with `Gviz` can create the tracks independently and add their own to create the desired plot. Available track creation functions are:

- CoverageTrack
- RawReadsTrack
- NucleosomeTrack

Here we add an Annotation track to the plot:

```
> library(Gviz)
> cTrack <- CoverageTrack(dataIP, "chr1", "gen")
> rTrack <- RawReadsTrack(dataIP, "chr1", "gen")
> nTrack <- NucleosomeTrack(PS, "chr1", "gen")
> gTrack <- GenomeAxisTrack(add53 = TRUE, add35 = TRUE)
> aTrack <- AnnotationTrack(range = IRanges(start = 149500, end = 151000),
+     showFeatureId = TRUE, id = "random annotation", col.title = "orange",
+     chr = "chr1", gen = "gen", name = "custom")
> plotTracks(trackList = c(gTrack, cTrack, aTrack, rTrack, nTrack),
+     main = "Custom plot", from = 149000, to = 153000)
```


Custom plot
150 kb
152 kb
151 kb
5'
3'
3'
5'
XSET
custom
random annotation
Raw reads
PING