Triform: peak finding in ChIP-Seq enrichment profiles for transcription factors

Karl Kornacker*and Tony Håndstad†

June 5, 2012

A guide for using the Triform algorithm to predict transcription factor binding sites from ChIP-Seq data

Contents

1 Licensing **2**

2 Introduction **2**

3 Overview of Triform **2**

4 Parameters and configuration file **3**

5 Preprocessing BED files **4**

6 Running Triform **4**

7 Session info **5**

*kornacker@midohio.twcbc.com
†tony.handstad@gmail.com

1 Licensing

This package is available under the GPL 2.0. license.

2 Introduction

Chromatin immunoprecipitation combined with high throughput sequencing (ChIP-Seq) is currently the method of choice for genome-wide mapping of binding sites for transcription factors on DNA. An essential step in the analysis of ChIP-Seq data is the genome-wide identification of enriched regions. The Triform algorithm represents an improved approach for automatic identification of peaks in ChIP-Seq enrichment profiles. Triform uses robust genome-wide statistical tests to detect three different forms of peak-like enrichment profiles, and takes advantage of multiple peak profile characteristics. These characteristics include the shift property, which occurs because the full sequence fragments, typically with an average length around 200bp, are sequenced only 25-50bp from each side. Triform can also use independent control samples, biological replicates, and is designed to separate overlapping enrichment profiles.

3 Overview of Triform

Usage of Triform is split in two steps. In the preprocessing step, information describing ChIP-seq tags in the form of BED-formatted files are converted to a format that describes the tag counts along the chromosomes on the different strands. The BED format is a tab-delimited format where each line describes the position of a mapped read (tag) in the form of space/chromosome, start, end, name, score, strand. Triform will ignore the name and score columns as they are not relevant here. Both control signal (i.e. ChIP-seq reads for control experiments without a TF-specific antibody) and up to several different TF signal files can be processed in the same run. After the preprocessing step, triform itself can be run and will then output the enriched regions.

Both the preprocessing step and the triform step consists of running a single function. Both functions require certain parameters. It is easiest to use a configuration file to supply these parameters, but the parameters can also be supplied directly to the preprocessing or triform function. The configuration file must be in YAML format. See `http://biostat.mc.vanderbilt.edu/wiki/Main/YamlR` for a description of YamlR and `http:`

//cran.r-project.org/web/packages/yaml/index.html for a description of the yaml R package. An example configuration file is available under the inst/docs directory in the triform package, and its contents is also shown below.

4 Parameters and configuration file

A total number of 12 parameters must be set to run Triform. These are most easily supplied using a configuration file in the YAML-format. Each line contains the parameter name and value separated by a colon. Some parameters can take multiple values, these values are then given one per line with a dash before the value. Below is an example of a configuration file. The text after the hashes are comments, explaining the purpose of the parameter.

```
READ.PATH : ./tmp  ## Path to source files (reads in BED format)
COVER.PATH : ./chrcovers ## Path for chromosome coverage files
OUTPUT.PATH : ./tmp/Triform_output.csv ## Path for output file (including filename)

TARGETS :
## Filenames for TF experiments
## Must include replicate name (_rep1 or _rep2), and .bed file ending
 - srf_huds_Gm12878_rep1.bed
 - srf_huds_Gm12878_rep2.bed

CONTROLS :
## Filenames for control/background experiments
## Must include replicate name (_rep1 or _rep2), and .bed file ending
 - backgr_huds_Gm12878_rep1.bed
 - backgr_huds_Gm12878_rep2.bed

READ.WIDTH : 100  ## Read width (used when preprocessing data) (w)
FLANK.DELTA : 150 ## Fixed spacing between central and flanking locations
MAX.P : 0.1       ## Minimum p-value, used to calculate min.z

MIN.WIDTH : 10  ## Minimum peak width (min.n)
MIN.QUANT : 0.375 ## Minimum quantile of enrichment ratios.
MIN.SHIFT : 10  ## Minimum inter-strand lag between peak coverage distributions

CHRS : ## Chromosomes to be used in Triform peak detection
```

```
- chrY
```

5 Preprocessing BED files

Start by loading the triform package.

```
> library(triform)
```

This will make the functions "preprocess" and "triform" available. Here, we will use sample data available in the vignette directory for the package. The configuration file is also available in the vignette directory, so this is given to the preprocess function.

```
>   preprocess("./config.yml")
```

Each replicate of each TF or control signal should be in its own BED file. The preprocessing will first convert all files with the .bed-extension in the READ.PATH directory to IRanges RangedData objects and save them as RData files. Thereafter, the preprocessing will use the READ.WIDTH parameter to divide each chromosome into segments and calculate for each signal and strand, the number of reads in each segment. The preprocessing ends by saving one file for each chromosome in the dataset, combining all signals and replicate information for the given chromosome in one file.

6 Running Triform

After preprocessing, Triform can be run similarly, by supplying the path to the configuration file to the triform function:

```
>   triform("./config.yml")
```

Triform will then process each chromosome and output each predicted peak region to a file whose path was given in the OUTPUT.PATH parameter.

Note that it is also possible to run preprocessing and Triform by supplying the parameters directly instead of using a configuration file. In that case, populate a named list with the parameters and consider setting the configPath parameter to NULL. Parameters supplied in the params list will overwrite the values set by any parameters in the configuration file.

```
preprocess(configPath=NULL, params=list(READ.PATH="./inst/extdata",
           COVER.PATH="./inst/extdata", READ.WIDTH=100))

triform(configPath=NULL, params=list(COVER.PATH = "./inst/extdata",
        OUTPUT.PATH = "./inst/extdata/Triform_output.csv",
        MAX.P = 0.1, MIN.WIDTH = 10, MIN.QUANT = 0.375, MIN.SHIFT = 10,
        FLANK.DELTA = 150, CHRS = c("chrY"), CONTROLS =
        c("backgr_huds_Gm12878_rep1.bed", "backgr_huds_Gm12878_rep2.bed"),
        TARGETS=c("srf_huds_Gm12878_rep1.bed", "srf_huds_Gm12878_rep2.bed")))
```

7 Session info

```
> sessionInfo()

R version 2.14.1 (2011-12-22)
Platform: x86_64-pc-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=C                 LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] triform_1.0    yaml_2.1.4     IRanges_1.12.6

loaded via a namespace (and not attached):
[1] tools_2.14.1
```