TDCA: Time-Dependent ChIP-Sequencing Analyser

Authors: Mike Myschyshyn, Marco Farren-Dai, Tien-Jui Chuang and David Vocadlo.
Download: https://github.com/TimeDependentChipSeqAnalyser/TDCA

1. Overview

1.1 Background

Chromatin immunoprecipitation followed by sequencing (ChIP-seq) is an established and robust method to generate genome wide maps of DNA binding proteins. Recently, new methods have been developed allowing time resolved ChIP-seq experiments to be conducted, effectively allowing protein-DNA binding dynamics to be established. As a response to the increasing potential of time course ChIP-seq experiments, we developed the first robust software specializing in time course ChIP-seq analysis. Our software, Time-Dependent ChIP-seq Analyser (TDCA), produces biologically relevant output informing users of protein-DNA binding dynamics. TDCA reads alignment data in BAM file format and genomic coordinates in BED file format.

1.3 Implementation

TDCA functionalities were developed using C++ and its graph features were built-up under R. TDCA uses samtools for bam file depth calculations and bedtools for peak intersection with genome features. TDCA makes various calls to the command line while running such as sed, awk, find, and others. Some hard coded files are created as well which users should keep in mind while using TDCA in pipelines. Samtools, bedtools, and R must be accessible from the command line.

1.4 Note about the manual

The proceeding contents of the manual contain example commands including TDCA usage. If a line starts with the $ character, it is meant to imply a command run in the terminal with the appropriate files available in the working directory. The following instructions assume a basic understanding of terminal navigation and command execution. This manual is intended to teach uses how to effectively use TDCA.

2. Installation

2.1 Software Requirements

Before installing TDCA, we require users to install the dependencies listed below, in Table 1:

Table 1: TDCA dependencies and download links

Name	**Download Link**
R	https://cran.r-project.org/bin/windows/base/ (Windows)
R Package plot3D	install.packages(“plot3D”)
R Package drc	install.packages(“drc”)
R Package rgl	install.packages(“rgl”)
R Package ggplot2	install.packages(“ggplot2”)
BedTools	https://github.com/pezmaster31/bamtools
SamTools	https://github.com/samtools/samtools

TDCA requires bedtools (Quinlan and Hall, 2010) and samtools (Li *et al*., 2009) be installed on the user's local computer and set on the environmental variables. TDCA calls these programs in many of its calculations. The user can check if bedtools and samtools is accessible globally by typing “samtools” and “bedtools” in the command line. If the programs are accessible, relevant information regarding the program will print. Alternatively, the user can check their bashrc file, or equivalent, with the following command:

```
$gedit ~/.bashrc
```

An alternative text editor to gedit may be used. If samtools and bedtools are set in environmental variables then the path to each of these programs directories will be documented on a line in the bashrc file like this:

```
export PATH=$PATH:/software/folder/bedtools/bin
export PATH=$PATH:/software/folder/samtools/bin
```

Although depending on how software was installed, the above lines may not be present in your bashrc file. Again, the simple way to check if these programs are installed and accessible to TDCA is by typing “samtools” and bedtools” in the command line, because this is how TDCA try to call these programs.

TDCA uses the R packages drc (Ritz,C. *et al*. 2015) for sigmoidal curve fitting and ggplot2 for graphical output (H. Wickham, 2009). Installation of R packages can be conducted as follows:

```
$R
Installing package:
```

```
>install.packages("package_name")
```
Checking if package is installed:
```
>library ("package_name")
```

To our knowledge, the R package drc which is used for curve fitting requires R version 3.3.1 or later.

2.1 TDCA Installation Guide on Linux

Once all the dependencies are installed users can proceed to TDCA installation. Users can download a tar file from:
https://github.com/TimeDependentChipSeqAnalyser/TDCA

Unpack and navigate to the tdca directory. Assuming the unpacked tdca folder is in the home directory type:
```
$cd home/tdca
```
Run make:
```
$make
```
Now add tdca directory to environmental variables to allow accession from any directory:
```
$gedit ~/.bashrc
```
Write this line: export PATH=$PATH:home/tdca

Once tdca is added to the environmental variables, the program can be used from any directory like so:
```
$tdca <options>
```
If tdca is not added to the environmental variables, the full program path must be specified from the working directory. Ex:
```
$/home/tdca/tdca <options>
```

2.2 TDCA Installation Guide on Windows

2.2.1 Virtual Box

1. Download Oracle VM VirtualBox
2. Download Ubuntu Desktop
3. Install VirtualBox
4. In VirtualBox Manager, click New
5. Give a name to the operating system and select Linux for Type and Ubuntu 32/64 bits depending on the Windows OS
6. Fo RAM Memory size, give above 4GB (4096MB)
7. For Hard Disk, select Create a virtual hard disk now
8. For Hard Disk File Type, select VDI
9. Select Dynamically allocated
10. For File Location and Storage, give above 32 GB

11. In Storage, click on [Optical Drive] and select Choose a disk image
12. Open the ubuntu-version-desktop-amd32/64.iso
13. Click Start to initiate Ubuntu system in VirtualBox
14. Follow the steps to complete the Ubuntu installation
15. Once Ubuntu is successfully installed, run the command prompt and install all the necessary softwares for TDCA. Please read the installation guide on Linux in 2.1

2.3 Using TDCA

Calling tdca without any arguments results in an output of the flag options and a brief description, as shown in Table 2

Table 2: TDCA options and brief description

Command	Description
-v	Display program version and Exit program.
-h	Display a detailed list of all command line options and exits the program.
-bam	User specified folder containing sorted bam turnover files including index files.
-bed	User specified bed file containing loci of interest.
-i	User specified folder containing sorted input bam turnover files including index files.
-g	Genome name. Currently supported: mm10, mm9, hg38, hg19, dm6, dm3, ce11.
-3d	User specified gene file containing RefSeq gene names.
-s	depth threshold (allowable range from 0.5-0.95). Default = 0.85.
-t	Time point threshold consideration for data modelling (allowable range from 0-2). Default = 1.
-name	User specified name for output files. DEFAULT: turnover.exp.
-model	Data modelled based on prediction. Default is no data modelling.
-dm	Data matrix used to normalize user defined input files.

3. Core Algorithm Description

TDCA reads genomic coordinates provided in BED file format, extracts depth values at those coordinates from the provided BAM file format with SAMtools, and models the depth values to five parameter (5P) sigmoidal curves using the drc R package. The equation and description of parameters for a 5P sigmoid are shown in equation 1.

$$f(x) = d + \frac{(a-d)}{(1+(x/c)^{b})^{e}} \qquad \text{Eqn. 1}$$

Where,
a = lower asymptote
b = Hill's coefficient
c = Infection point
d = upper asymptote
e = asymmetry factor

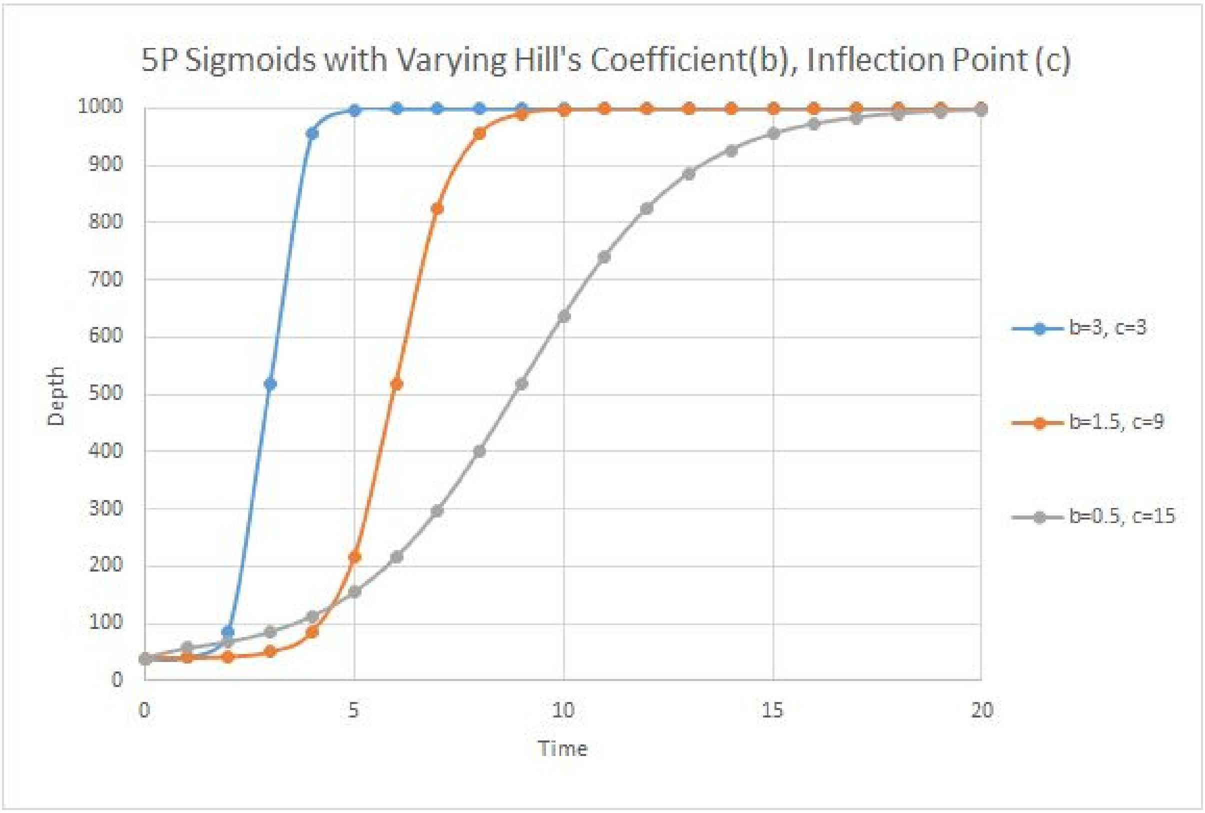


Figure 1: Graphical representation of the effect of changes in Hill's Coefficients on shape of the curve. All other parameters are the same for all the three curves (a=40, d=1000, e=1)

TDCA normalizes sequencing depth data before modelling. This is done in a two fold manner. Firstly, the values at each loci are normalized by the maximum sequencing depth at non-peak loci for all time points in a time course replicate. Using non-peak loci is meant to capture true background sequencing levels. Table 3 indicates this behaviour in tabular format.

Table 3: Normalization of time course Chip-seq experiments based on sequencing depth.

Time course experiment (relative time units)	Depth at non-peak loci	Normalization constant
1	5,000,000	1.20
2	6,000,000	1.00
3	5,500,000	1.09
4	4,000,000	1.50

The normalization constant is unique for each time point and is carried out throughout the program to correct for sequencing depth. The constant is multiplied by the values calculated at each loci. Additionally, TDCA can incorporate a standard input for normalization. ‘Input’ refers to sequencing data for an experiment wherein the protein-DNA complexes are not immunoprecipitated and the result is the baseline depth distribution. If input is provided, the input is normalized by the same manner except using depth across the entire genome rather than at non-peak loci. The input values for each time point are then subtracted from the experimental depth values at each loci. If replicates are included, these final values are averaged. Alternatively, TDCA can use specified normalization values (see -dm flag explanation, section 5.1.9: Reporting Turnover Rates with a Specified Depth Matrix).

TDCA uses normalized depth values to model data. Without the -model flag, TDCA uses all time points at a given loci to model data which results in either a rise (5P sigmoid increasing in signal over time) or fall (5P sigmoid decreasing in signal over time). If the -model flag is specified, TDCA uses a prediction algorithm based on the time at which the absolute minimum depth value and absolute maximum depth value occurs in time. TDCA then checks if there are trailing data points (those occurring before in time), and leading data points (those occurring after in time) the absolute minima and maxima to decide if a single loci should be modelled as a hill or valley (candidate for double 5P sigmoid modelling). This is where the -s and -t flags come into play.

Given a loci with absolute maximum depth D_{max} and absolute minimum depth D_{min}, over time, the range of depth $R = D_{max} - D_{min}$.

In order to decide if a depth value at a certain time point is a trailing or leading time point of the absolute minimum or maximum, the user specified -s value (numerical value S, with a range of 0.5-0.95, default = 0.85) is considered.

For values trailing or leading the absolute maxima, R is multiplied by S to get a threshold and if the leading or trailing depth value is lower than this limit (outside of threshold, ie for default 0.85 the trailing/leading point has at least 85% of the depth range) it is considered a true leading or true trailing data point. This can be represented mathematically as time point T is trailing/leading the absolute maxima if $T \geq R * S + D_{min}$, where $R = D_{max}$ - D_{min}, $S$is the -s value and T is the depth value at time point T.

For values trailing or leading the absolute minima, R is subtracted by the product of R and S to achieve the limit L (L=R-R*S). If the leading or trailing depth value surpasses L (outside of threshold, ie for default 0.85 the trailing/leading point has at most 15% of the depth range) it is considered a true leading or trailing data point. This can be represented mathematically as time point T is trailing/leading the absolute minima if $T \leq (R - R * S) + D_{min}$, or $T \leq R(1 - S) + D_{min}$ where $R = D_{max}$ - D_{min}, $S$is the -s value and T is the depth value at time point T.

Given the implications of the -s flag, the user can tailor their desire for points to be considered as true leading or trailing data points of the absolute maxima and absolute minima. This causes TDCA to more likely define loci type as rises and falls rather than hills and valleys as the -s parameter becomes smaller, this is useful for data with high standard deviation is but the expected behaviour is of rises/falls.

The -s flag is essential in determining if a data point is considered a true leading or trailing data point of the absolute minima and maxima depth values. The -t flag also affects how TDCA will define loci type. -t decides how many of these true leading and trailing points (determined by -s) are necessary to define the categorical shift from rises and falls to hills or valleys, essentially determining if a single or double 5P sigmoid should be modelled to the depth values at a given loci. The default value of -t is 1 and is translated into the requirement of at least 2 data points necessary to lead or trail the absolute minima and/or maxima in order to shift how TDCA models data. The lowest value of -t is 0, which requires only 1 depth value defined as a leading or trailing value of the absolute minima or maxima depth. Figure 2 shows sequencing depth of a hypothetical loci over 6 time points.

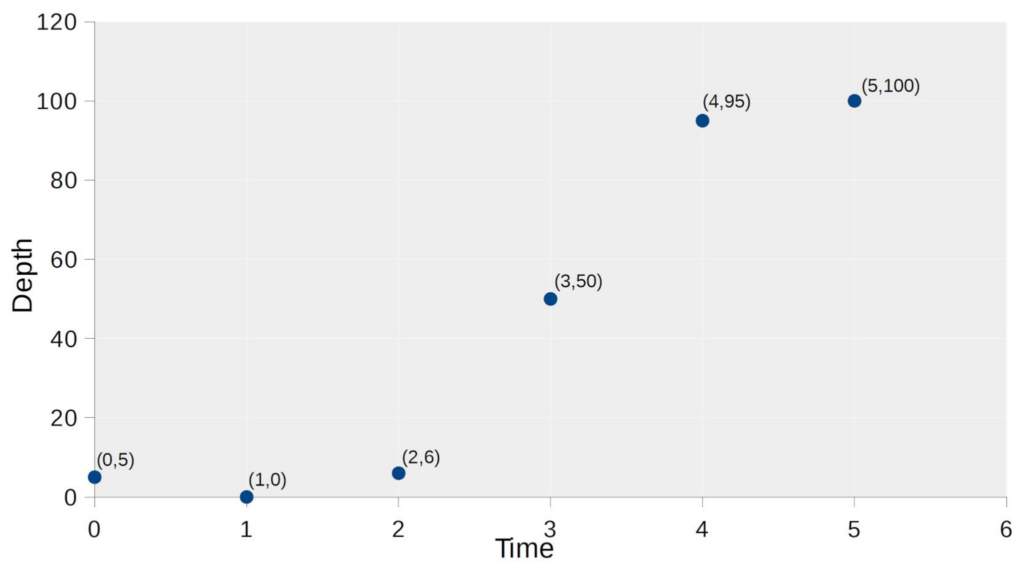


Figure 2: Sequencing depth of a hypothetical loci over 6 time points. Relative time units are indicated on the x-axis and depth on the y-axis.

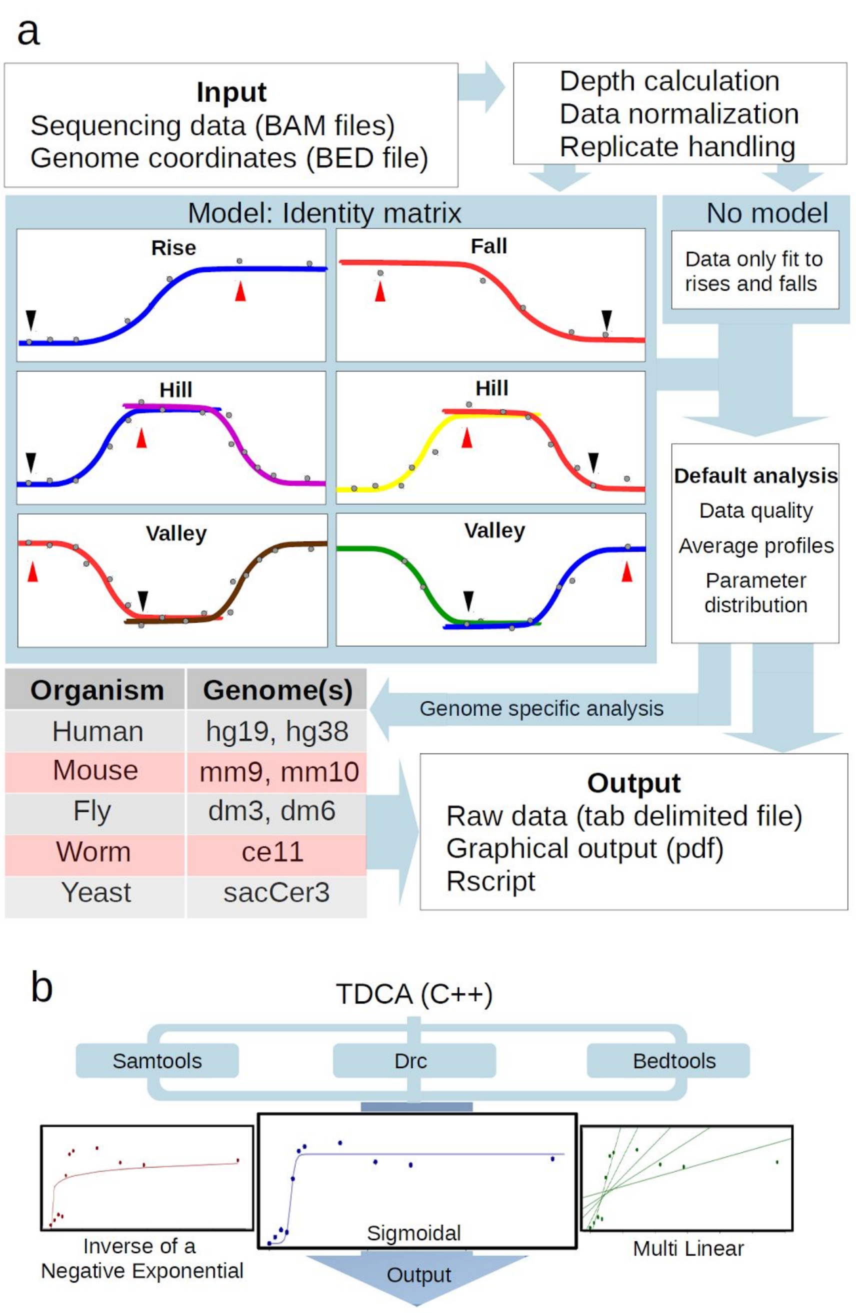
The loci in Figure 2 has a depth range, R, of 100 - 0 = 100. Since this loci has its absolute maximum at the last time point, it cannot have any data points leading the absolute maximum. In order for the data point (4,95) to be considered a true trailing data point of the absolute maxima, it must be greater than R*S+D_{min}. To calculate the value of S required to consider data point (4,95) a true trailing data point of the absolute maximum, make the depth value of this point equal to the limit: 95 = R * S+D_{min}. Since R = 100 and D_{min}=0, S must be 0.95. If the -s flag was set to a value lower than 0.95, the data point (4,95) would be considered a true trailing data point of the absolute maximum.

Similarly, for the absolute minimum data point (1,0), the -s flag must be set to a value lower than 0.95 for the data point (0,5) to be considered a true trailing data point. Also, the -s flag must be set to a value lower than 0.94 for the data point (2,6) to be considered a true leading data point. Notice how the lower the -s flag is set, the more likely data points will be considered true leading or trailing values.

TDCA then determines if data at each loci should be separated into two 5P sigmoid curves based on whether the number of leading and trailing data points of the absolute minima and maxima exceed the -t flag value. Ultimately, this results in the assignment of all loci as either a rise, fall, hill, valley, or undefined. Loci that are undefined contain data points that do not model as the other types. Undefined loci are, however, still analysed by TDCA as rises or falls.

Now that the TDCA has defined a type for each loci, data is modelled with drc. For data with increasing signal over time (rises, undefined rises, and inclines of hills and valleys), TDCA uses a 5P sigmoid equation as default. If this results in a lower asymptote of less than zero (biologically meaningless since one cannot sequence negative DNA), then TDCA forces the lower asymptote to zero. If, however, the lower asymptote is less than zero and the inflection point is five times greater than the maximum time point (error prone prediction), TDCA forces the lower asymptote to the minimum depth value. This recovers data closer to its true values based on simulated data testing. For data with decreasing signal over time (falls, undefined falls, and the declines of hills and valleys), TDCA uses a similar procedure as above except the upper asymptote is fixed to the max depth value.

Once modelling is complete, TDCA verifies if the models match the prediction. If not, the loci are eliminated. Note that running TDCA with no model, there cannot be eliminated loci. Figure 3 shows a visual representation of the TDCA core algorithm. The graphical output mainly focuses on a modified version of the inflection point. This is indicative of the binding half life of a protein at a particular loci. TDCA adjusts the inflection point using the asymmetry value in the 5P sigmoid curve. This resulted in more accurate predictions of simulated symmetrical sigmoidal data. We call this adjusted value the turnover time index (TTI).



Organism	Genome(s)
Human	hg19, hg38
Mouse	mm9, mm10
Fly	dm3, dm6
Worm	ce11
Yeast	sacCer3

Figure 3: TDCA analysis workflow, requirements and performance. (a) Simplified workflow. Required input data are genomic coordinates in BED format and a folder containing BAM files of time course sequence files. TDCA normalizes data based on total sequencing depth of each time point and also handles input files and replicates using additional normalization procedures. Loci can be modelled as the following types of signal change: rise, fall, hill or valley. An identity matrix that predicts loci type based on the time at which absolute minimum depth (**black** arrows) and absolute maximum depth (**red** arrows) occurs, along with user defined thresholds. Alternatively, users can model their data as a single 5 parameter curve (rise or fall only). The resulting parameters from data fitting are then reported to the user along with raw depth calculations. Graphical output is provided to the user which can be enriched by specifying genome and genes. Rscripts are provided in case users

would like to change the look of default figures. (b) TDCA requires samtools for depth calculation of BAM files, bedtools for BED file manipulations, and R and the drc package for curve fitting to be accessible from the terminal. Plots show depth (y-axis) across time (x-axis) of loci with coordinates chromosome 1:5012338-5013264 in a H3.3 ChIP-seq experiment using previously applied modelling strategies of inverse negative exponential (left) and multilinear (right), and the novel sigmoidal fitting by TDCA.

Once normalization and modelling is complete, TDCA reports the raw data in a tab delimited file and uses the modelled parameters to create biologically insightful graphs that provide general information about the user's experiment as well as future direction.

4. General Usage Information

4.1 Supported File Format

4.1.1 BED File Format

The required BED file should contain only three columns. The format for -bed <bed_peaks.BED> is the following:

1. Chromosome, the name of the chromosome
    - Any string. ex. “Chr10”
    - Mandatory column
2. Start, the starting point
    - Any positive integer within the range of the genome of interest. ex. “23507998”
    - Mandatory column
3. End, the ending point
    - Any number that is greater than starting point in above, and is a positive integer within the range of the genome of interest. ex. “23508239”
    - Mandatory column

4.1.2 BAM File Format

The BAM folder for the -bam <bam_files_folder> flag requires bam files to be sorted and indexed and named with a “XXX_integer.bam” extension, where integer is the time in relative units of the time course experiment. ‘XXX’ should denote the additional file name.

4.1.3 Text File Format

The required text file for -3d <text_file> is a list of refSeq gene names separated by newlines (/n).

The required text file for -dm <text_file> is a list of integers separated by newlines (/n). The number of integers must equal the total of BAM files to be analysed. The depth value of integer in the -dm file will be assigned to BAM files based on the order of the BAM folder in the argument list and second by increasing time intervals. For further detail, see section 5.1.9: Reporting Turnover Rates with a Specified Depth Matrix.

5. TDCA Suite

5.1 TDCA

5.1.1 Usage and Options Information

Usage: $ tdca -bed <bed_peaks.BED> -bam <bam_files_folder>

Example: $tdca -bed ChIP-seq.peaks.bed -bam bamFolder -i bamInputFolder -g mm9 -3d gene_list.txt -name exp-name

Options	**Description**
-bed <bed_peaks.BED> -bam <bam_files_folder> (Mandatory)	BED file followed by BAM files folder. Sequencing depth of loci in bed_peaks.BED are extracted from each BAM file inside bam_files_folder in order to calculate the turnover rates.
-i <input_bam_files_folder> (Optional)	Input BAM files folder path. Input BAM files are used to normalize data.
-g <genome_name> (Optional)	Generates additional graphs including gene feature boxplot of TTI and ideogram TTI heat map. Supported genomes include: human (hg18, hg19), mouse (mm9, mm10), fly (dm3, dm6), worm (ce11), and yeast (sacCer3).
-3d <text_file> (Optional)	Text file. The -3d flag requires the -g flag. A series of compressed 3D scatter plot of depth for each gene listed in the given text file is generated as PDF.
-s <0.85> (Optional)	depth threshold (allowable range from 0.5-0.95). Default is 0.85. Value only applies if -model is called.
-t <1> (Optional)	Leading/trailing threshold (allowable range from 0-2). Default is 1. Value must be integer and only applies if -model is called.
-name <exp_name> (Optional)	Rename the output file as user-specific. Default is turnover.exp.

-model	Model data to rises, falls, hills, and valleys.
-dm <text_file> (Optional)	Text file containing integers for normalization. The number of integers must equal the total number of BAM files.

5.1.2 Default Behavior

Using only the mandatory parameters -bed and -bam, TDCA generates output containing three quality charts: pie chart of loci type (rises, falls, etc.), bar chart of absolute minimum and maximum depth as a percent occurrence of all loci, and a normalized depth heatmap across time points. The analysis graphs provided include average read profiles of each loci type, log2 upper asymptote and lower asymptote ratios for incline and declines of hills and valleys, and a scatter plot of TTI and Hill's coefficient for data separated by signal increase and signal decrease. Below are images and descriptions of the default output.

The following command would output the default graphs. The requirement of BAM file format is specified in 4.1.1 BAM File Format. BED file format is described in 4.1.2 BED File Format.

For example (-bed <bed_peaks.BED> -bam <bam_files_folder>):

```
$tdca -bed ChIP-seq.peaks.bed -bam bamFolder/
```

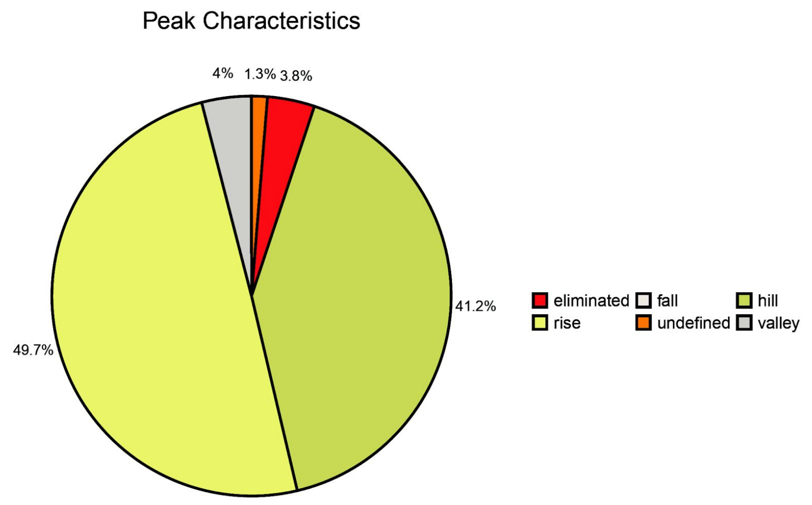


Figure 4: TDCA pie chart of data type. The diagram shows typical display when the -model flag is set. If the data is not modelled, the proportion of only rises and falls will be shown.

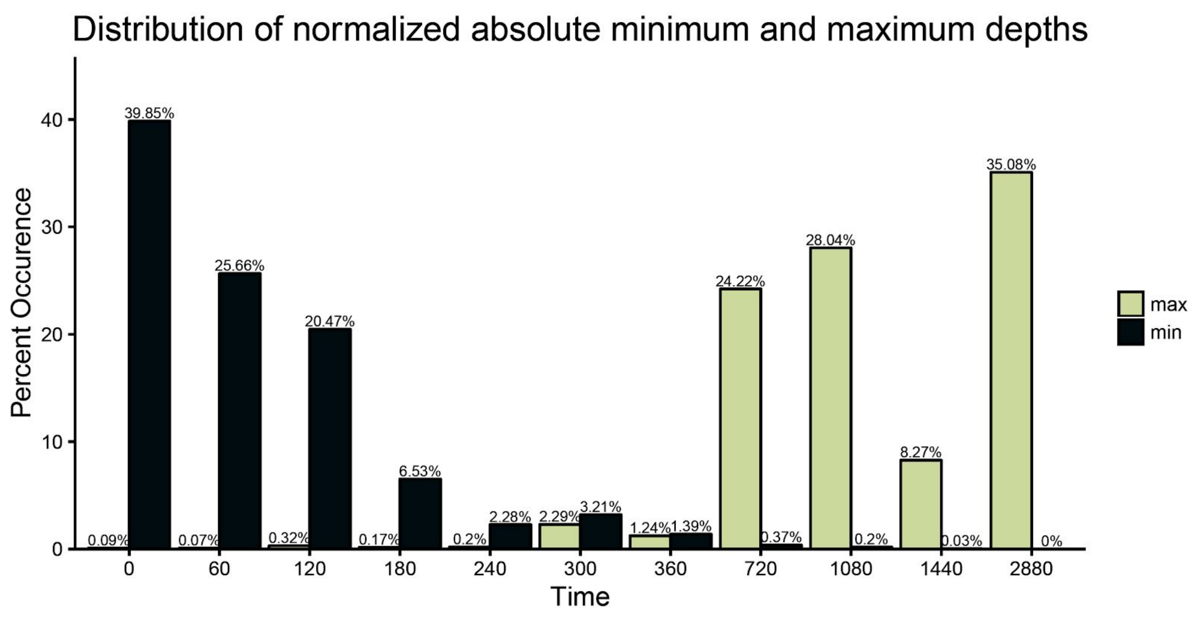


Figure 5: Percent occurrence of absolute minimum and maximum normalized depth across all loci at each time point. The behaviour shown is typical for data expected to model as rises.

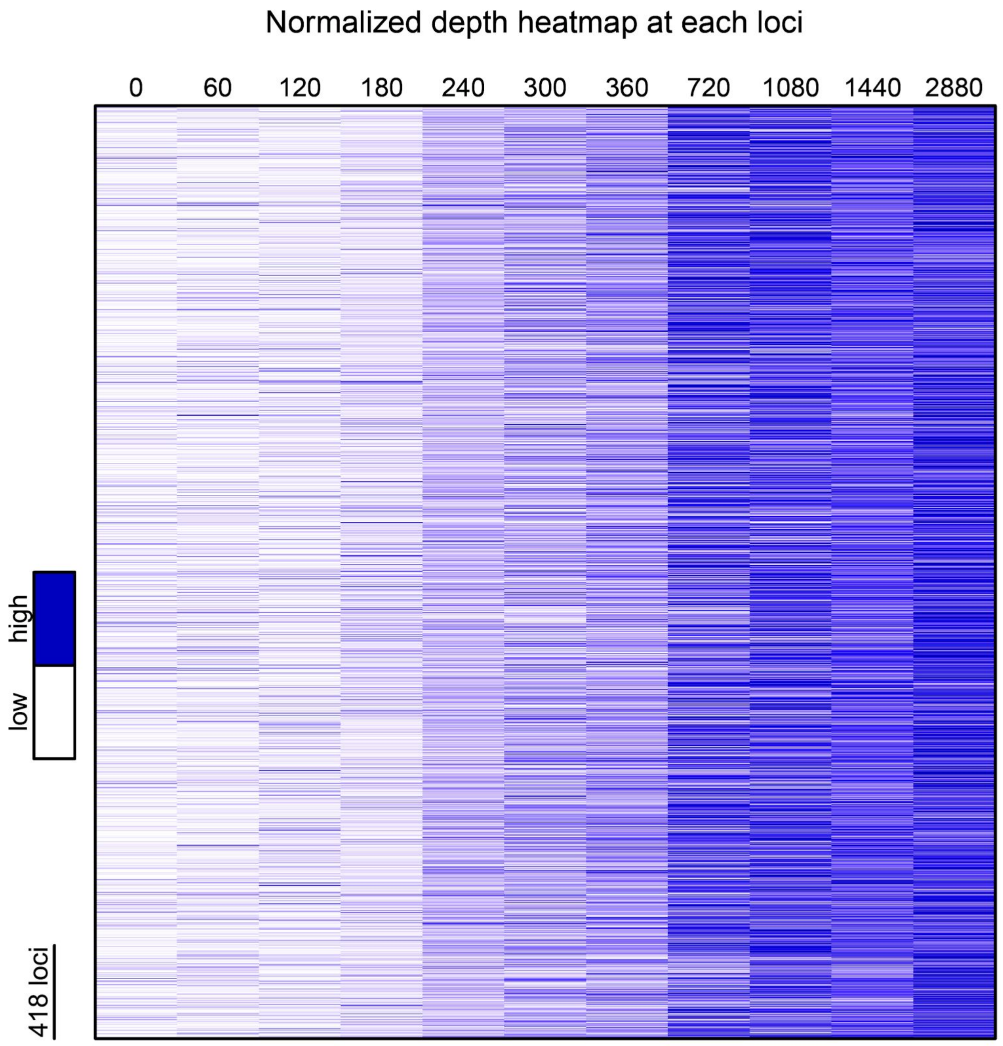


Figure 6: Normalized depth heatmap across time points. Depth of each loci across time points is shown as horizontal lines (see scale bar for loci width). Depth is normalized from 1 (max) to 0 (min) for each loci so that loci can be compared to each other.

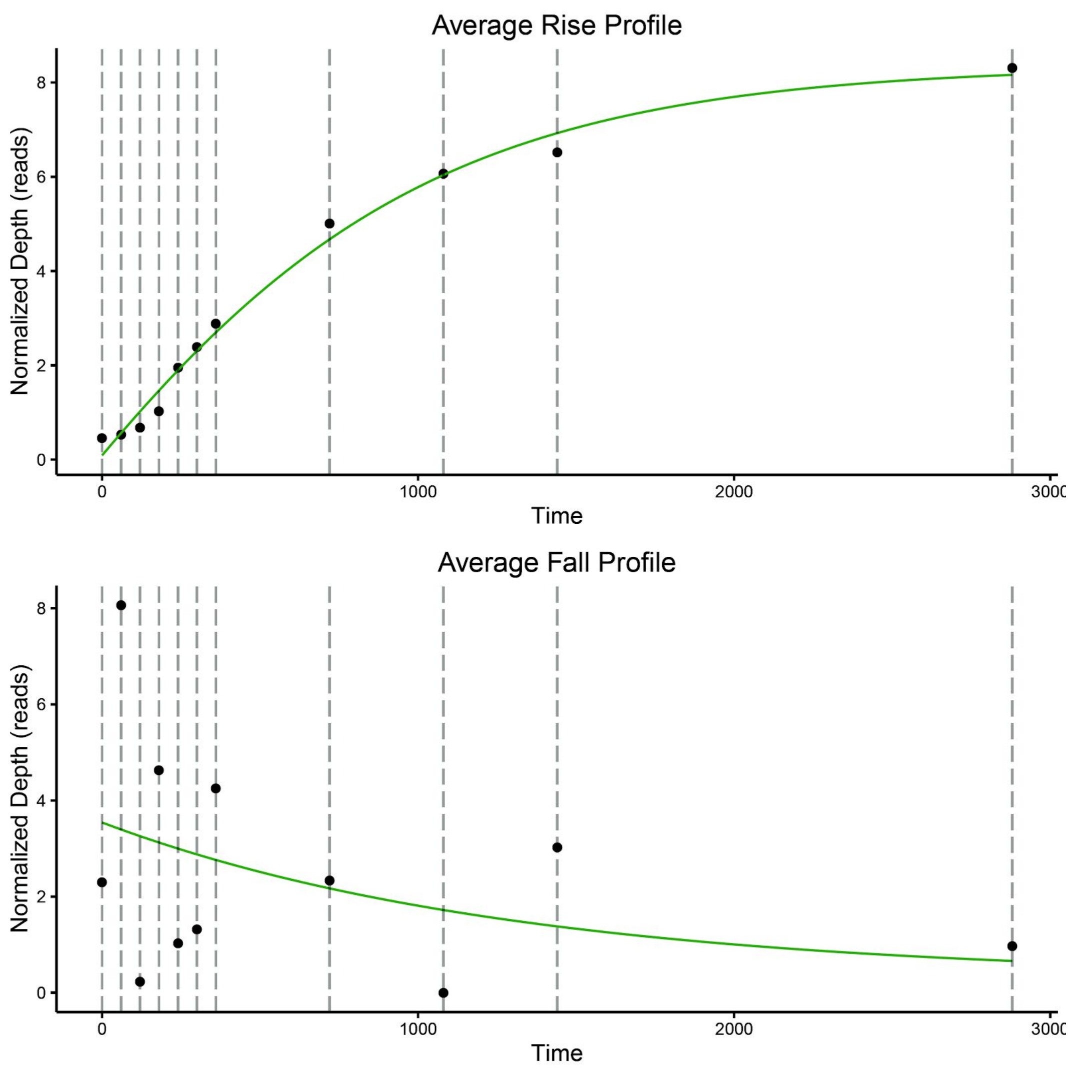


Figure 7: Average rise and fall profiles. TDCA analyses all loci modelled as rises and falls and produces an average profile as a fold enrichment of reads with zero as the lower limit rather than one. The vertical dashed lines indicate the time points used in the analysis. The dots indicate individual averaged values for each time point and the line is the modelled rise or fall. If the data contains no rises or falls then a blank graph will show up with a description indicating so.

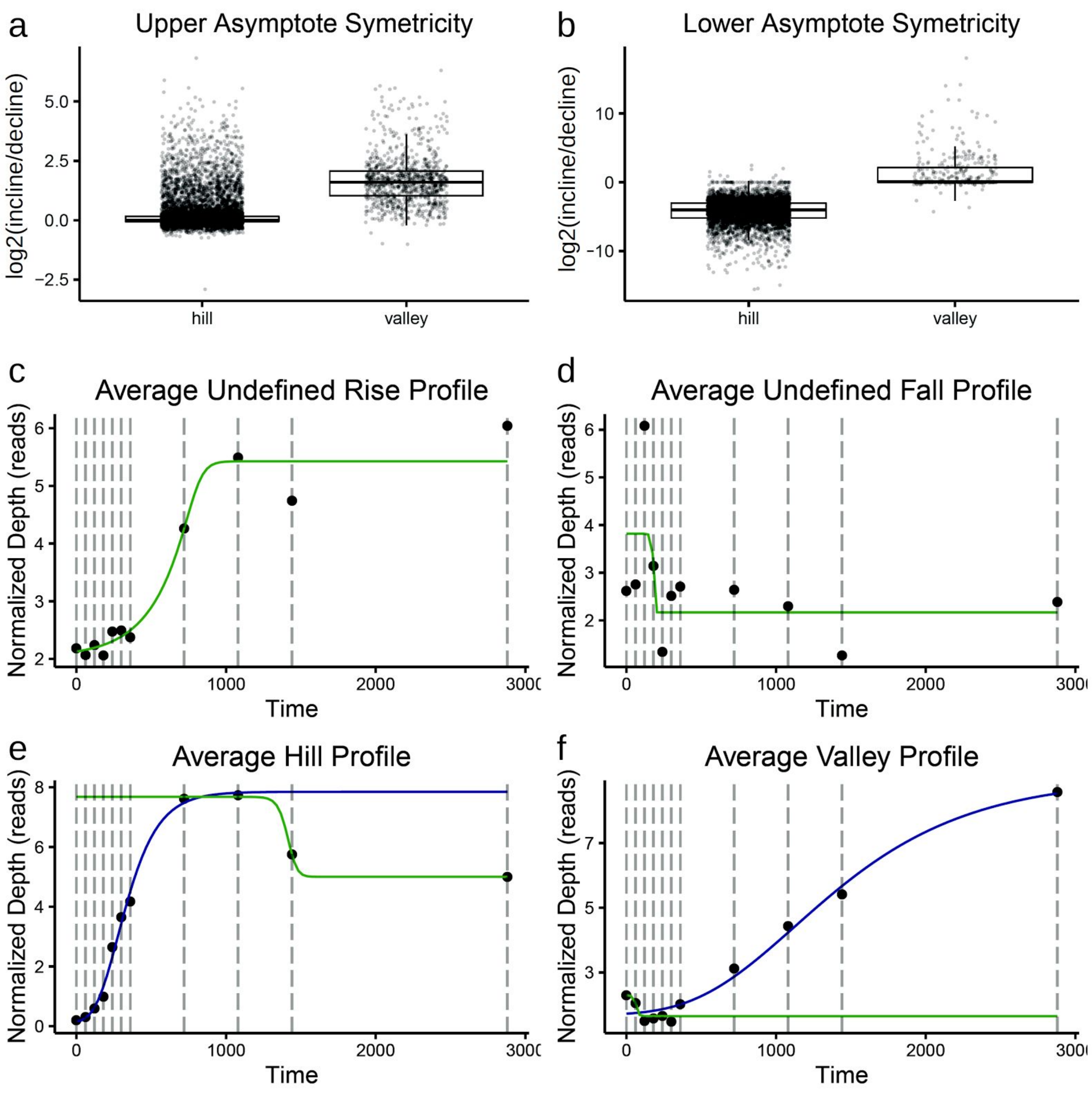


Figure 8: (a) log2 upper asymptote ratios for incline and declines of hills and valleys. (b) log2 lower asymptote ratios for incline and declines of hills and valleys. The log2 asymptote ratios are an indication of the symmetricity of hills and valleys. (c-f) Average profiles for undefined rises, undefined falls, hills, and valleys as in Figure 7. Blue lines in the hills and valleys profiles indicate the model for signal increase (incline) and green lines indicate the model for signal decrease (decline).

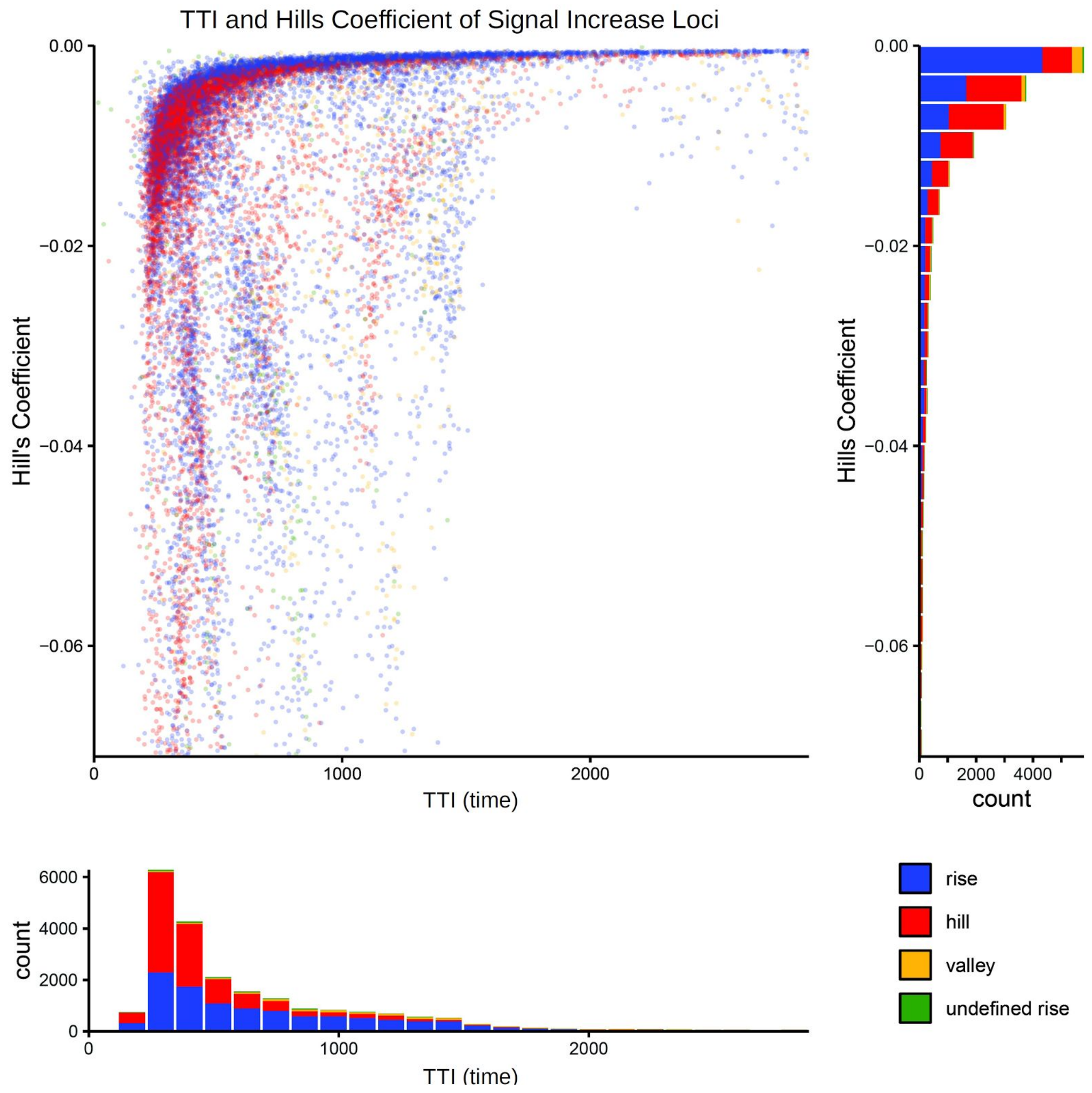


Figure 9: Scatter plot of TTI and Hill's coefficient for data separated by model types that display signal increase. A separate figure is output for loci that display signal decreases (not shown here). Histograms show loci count, of different loci type that fall within binned regions of TTI and Hill's coefficient.

5.1.3 Reporting the Turnover Rates with Multiple Replicate BAM Files

With additional replicated BAM files given by the user, TDCA will take extra time to compute the analysis and graphs. TDCA generates an extra page of graphs for users to perform data quality comparison, shown in Figure 10.

The requirement of BAM file format is specified in 4.1.1 BAM File Format. BED file format is described in 4.1.2 BED File Format.

For example (-bed <bed_peaks.BED> -bam <bam_files_folder> -bam <replicated_bam_file_folder_1>):

```
$tdca -bed ChIP-seq.peaks.bed -bam rep1-bamFolder/ -bam rep2-bamFolder/
```

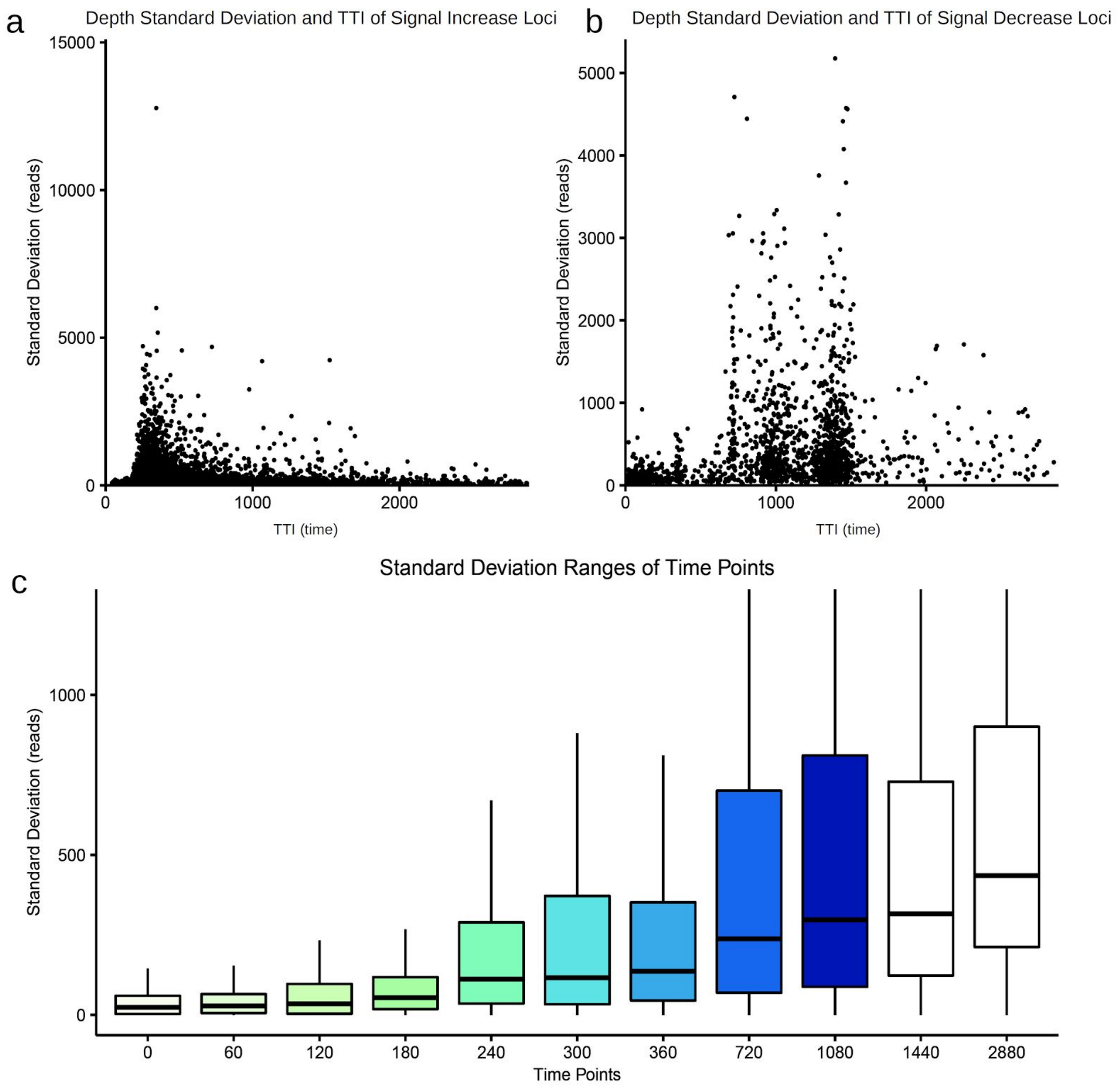


Figure 10: TDCA output with replicates. (**A**) Correlation of standard deviation and upper TTI of loci with signal increase. (**B**) Correlation of standard deviation and upper TTI of loci with signal decrease. (**C**) Distribution of standard deviation at across time points.

5.1.4 Reporting the Turnover Rates of Given BAM Files with Inputs (-i)

Given BAM input files from the user, additional normalization via subtracting depth of input to a lower limit of zero will be computed. No additional graphs are created.

The requirement of BAM file format is specified in 4.1.1 BAM File Format.
Example:

```
$tdca -bed ChIP-seq.peaks.bed -bam rep1-bamFolder/ -i rep1-bamInputFolder/ -bam rep2-bamFolder/ -i rep2-bamInputFolder/
```

5.1.5 Reporting the Turnover Rates of Given BAM Files with a Specific Genome (-g)

Given a user-specified genome that is supported, TDCA generates a box plot of TTI values at genomic features. Default genome features include: 3'UTR exon and 1000bp upstream of transcriptional start site, 5'UTR exon and 1000bp downstream of transcriptional end site, coding exons, CpG islands, introns, whole gene - characterized as 1000bp upstream of transcriptional start site to 1000bp downstream of transcriptional end site, and intergenic region - characterized as reciprocal coordinates of whole gene. Additional gene features can be included by the user in a bed file format. This process is described in section 5.1.10 Expanding Genome Feature Libraries. Supported genomes include human (hg19, hg38), mouse (mm9, mm10), fly (dm3, dm6), nematode (ce11), and yeast (sacCer3). Contact the authors to request additional genomes.

In addition, TDCA generates an ideogram heatmap of TTI values at each canonical chromosome.

For Example (-bed <bed_peaks.BED> -bam <bam_files_folder> -g <genome_name>):

```
$tdca -bed ChIP-seq.peaks.bed -bam bamFolder/ -i bamInputFolder/ -g mm9
```

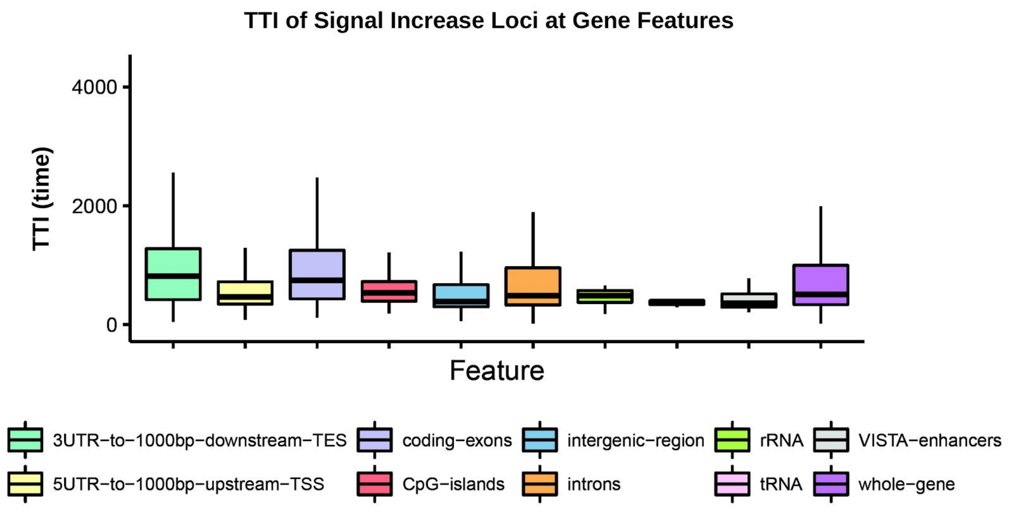


Figure 11: TDCA gene feature boxplot with specified genome. Distribution of TTI values of loci that display signal increase at different gene features. Lower lines, lower part of box, midline, upper part of box, and upper line are 1st quartile, 2nd quartile, median, 3rd quartile and 4th quartile respectively. A boxplot for signal decrease loci is also output (not shown here).

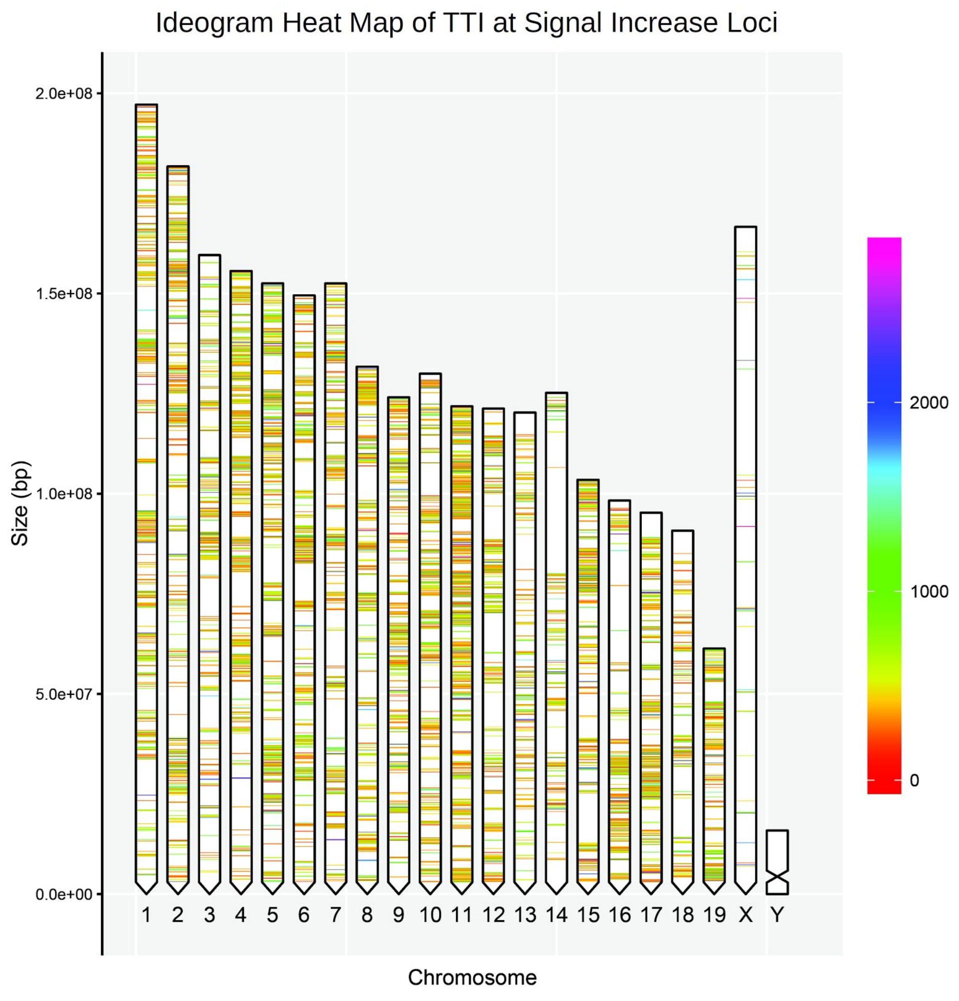


Figure 12: TDCA ideogram heatmap with specified genome. Chromosomes are shown as white polygons with labels at the bottom. Centromeres are located at crosses (most are telometric in mouse - shown here). Bands on the chromosomes indicate where user specified loci are with TTI indicated colorimetrically. Both signal increase loci (shown here) and signal decrease loci (not shown here) are output.

5.1.6 3D Depth Profiles of User Specified Genes

When a user-specified genome is provided, additional graphs can be printed as a series of 3D scatter plots displaying time dependent depth for select genes from a user specified gene list. The user-specified gene list is a text file with refSeq gene names separated by newline characters (\n).

The required format of the user-specified gene list is described in 4.1.3: Text File Format.

For Example (-bed <bed_peaks.BED> -bam <bam_files_folder> -g <genome_name> -3d <text_file>):

```
$tdca -bed ChIP-seq.peaks.bed -bam bamFolder/ -i bamInputFolder/ -g mm9 -3d chr.txt
```

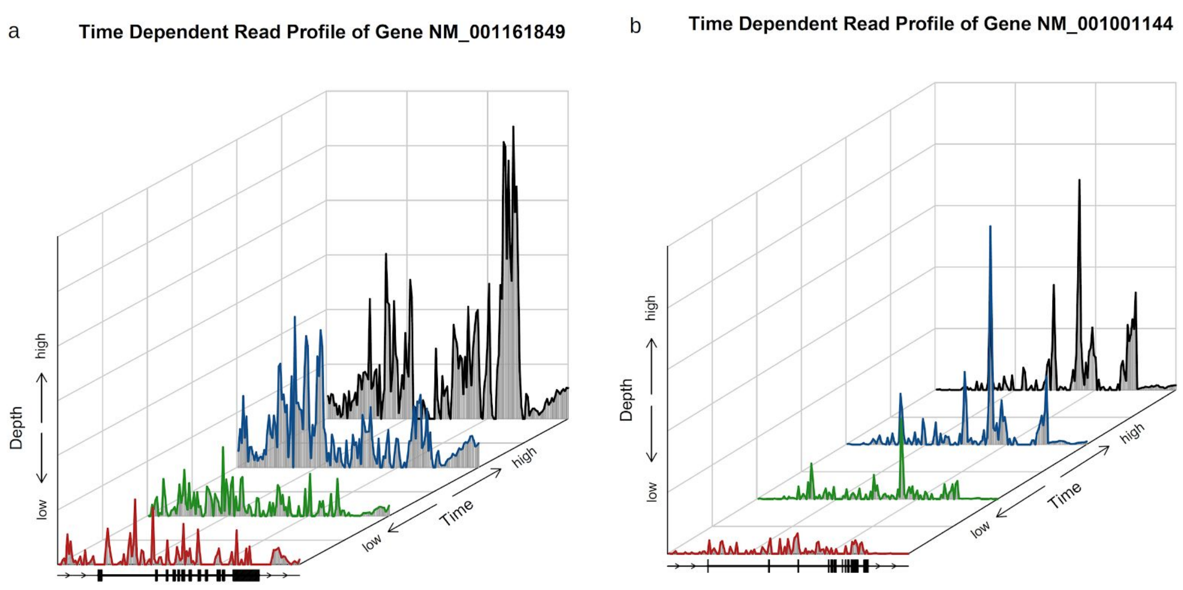


Figure 13: 3D depth scatter plots of NM_001161849 (a**)** and NM_001001144 (b).

The 3D scatter plot option shows time on the z axis (into the page). Data is compressed to show four time bins. This is done so that the plot is not cluttered. The compression processes is visually described in the Figure 14. The purpose of this diagram is to get a relative idea of turnover times for various peaks located at genes.

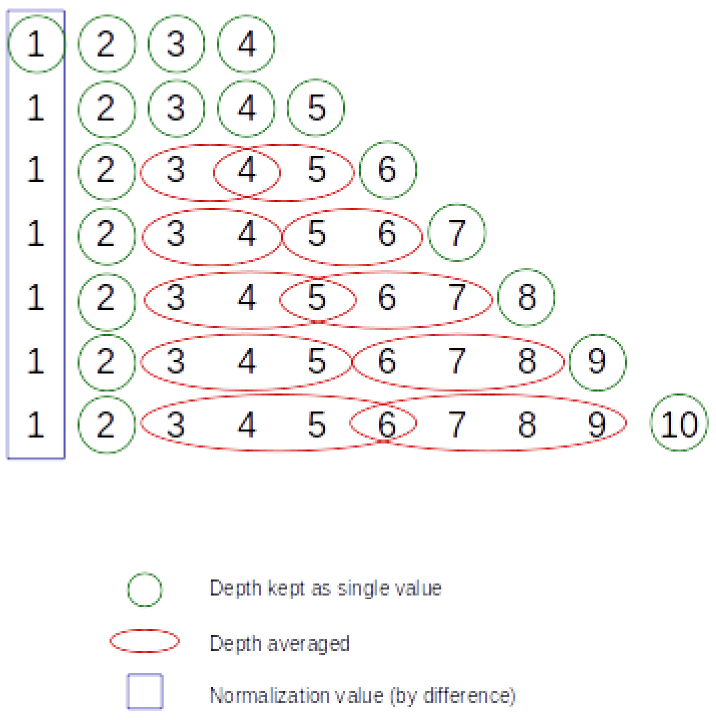


Figure 14: Visual display of 3D scatter plot algorithm. Numbers oriented in horizontal lines represent the number of time points in a given ChIP-seq time course experiment. The amount of time points shown in the scatter plot is set to four no matter how many additional time points are given as can be seen from the green or red circled time points, which represent depth values that are used in the plot (kept) or average, respectively. The first time point is used as a normalization point by subtracting its depth from the other data points. An experiment with only four time points would show all four, each normalized by the first, therefore the first time point would look like a flat line.

The x axis of all 3D genes show a picture of the exons in black boxes and introns in thicker black lines. 1000bp upstream and downstream regions of the gene are shown as thinner black lines on the left and right sides of the gene body respectively. An algorithm has been created to search a built in library of refSeq gene information. Keep in mind that refSeq genes have multiple isoforms of certain genes. Remember to choose the appropriate isoform. USCS browser is a great tool to visually inspect coordinates of isoforms (Kent,W.J. *et al*. 2002).

5.1.7 Reporting Turnover Rates with Different Depth Threshold

TDCA offers a depth threshold flag (-s). This threshold can be adjusted by the user to be between 0.55-0.95 (default = 0.85). The -s flag is discussed in section 3: Core Algorithm Description. -s plays a major role in the modelling procedure.

5.1.8 Reporting Turnover Rates with Different Trailing/Leading Threshold

TDCA offers a trailing/leading threshold flag (-t). This threshold can be adjusted by the user to be between 0-2 (default = 1). The -t flag is discussed in section 3: Core Algorithm Description. -t plays a major role in the modelling procedure.

5.1.9 Reporting Turnover Rates with a Specified Depth Matrix

Users can specify their own numbers to normalize BAM files by, genome wide and at non-peak loci, for input and experiment files respectively. A newline(\n) separated file containing integers equal to the number of BAM files must be specified after the -dm flag. TDCA will assign these values to BAM files in order of replicates, input before experiment, then by chronological order. The user can double check if TDCA assigned the correct values to each file by inspecting the runtime output.

5.1.10 Expanding Genome Feature Libraries

Users can input their own BED file format genome feature into the appropriate genome folder located in the TDCA GenomeFeatures folder. UCSC table browser was used to get default libraries (Karolchik D., *et al*. 2004). TDCA will use the newly input file(s) in analysis in of TTI at genome features.

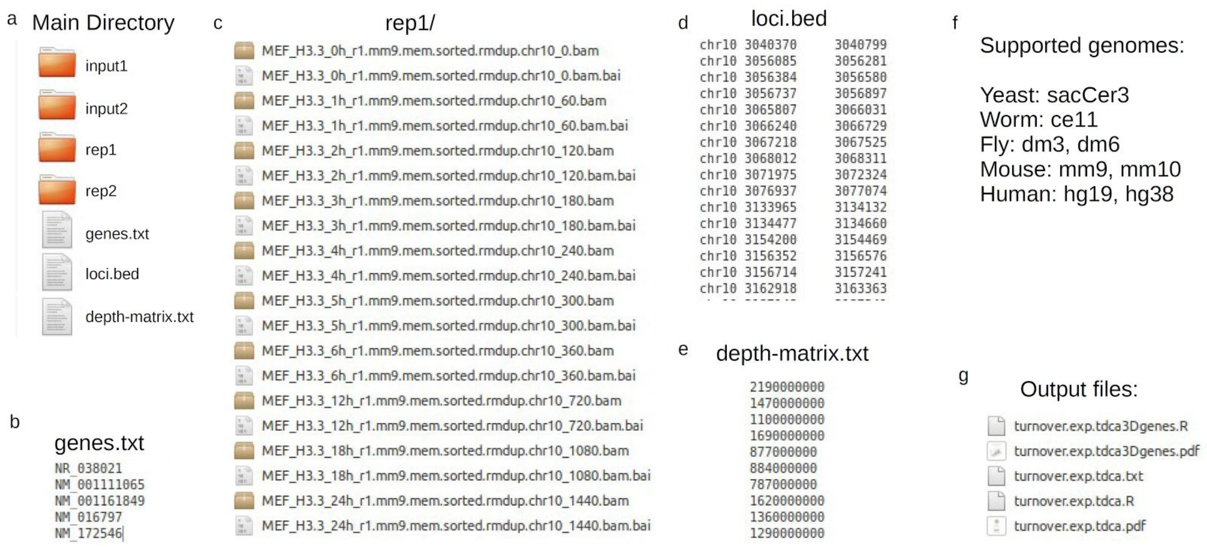
6. Example Usage

6.1 Comprehensive Example

In this short subsection we provide users comprehensive usage of TDCA including visual representation of the required files. The following command runs TDCA with 1 replicate, genes for 3D analysis, a depth-matrix, and specification of the mouse mm9 genome:

```
$tdca -bed loci.bed -bam rep1 -dm depth-matrix.txt -3D genes.txt -g mm9
```

Figure 15 shows a visual representation of required and additional files for this command.



```
NR_038021
NM_001111065
NM_001161849
NM_016797
NM_172546
```

```
MEF_H3.3_0h_r1.mm9.mem.sorted.rmdup.chr10_0.bam
MEF_H3.3_0h_r1.mm9.mem.sorted.rmdup.chr10_0.bam.bai
MEF_H3.3_1h_r1.mm9.mem.sorted.rmdup.chr10_60.bam
MEF_H3.3_1h_r1.mm9.mem.sorted.rmdup.chr10_60.bam.bai
MEF_H3.3_2h_r1.mm9.mem.sorted.rmdup.chr10_120.bam
MEF_H3.3_2h_r1.mm9.mem.sorted.rmdup.chr10_120.bam.bai
MEF_H3.3_3h_r1.mm9.mem.sorted.rmdup.chr10_180.bam
MEF_H3.3_3h_r1.mm9.mem.sorted.rmdup.chr10_180.bam.bai
MEF_H3.3_4h_r1.mm9.mem.sorted.rmdup.chr10_240.bam
MEF_H3.3_4h_r1.mm9.mem.sorted.rmdup.chr10_240.bam.bai
MEF_H3.3_5h_r1.mm9.mem.sorted.rmdup.chr10_300.bam
MEF_H3.3_5h_r1.mm9.mem.sorted.rmdup.chr10_300.bam.bai
MEF_H3.3_6h_r1.mm9.mem.sorted.rmdup.chr10_360.bam
MEF_H3.3_6h_r1.mm9.mem.sorted.rmdup.chr10_360.bam.bai
MEF_H3.3_12h_r1.mm9.mem.sorted.rmdup.chr10_720.bam
MEF_H3.3_12h_r1.mm9.mem.sorted.rmdup.chr10_720.bam.bai
MEF_H3.3_18h_r1.mm9.mem.sorted.rmdup.chr10_1080.bam
MEF_H3.3_18h_r1.mm9.mem.sorted.rmdup.chr10_1080.bam.bai
MEF_H3.3_24h_r1.mm9.mem.sorted.rmdup.chr10_1440.bam
MEF_H3.3_24h_r1.mm9.mem.sorted.rmdup.chr10_1440.bam.bai
```

```
chr10 3040370      3040799
chr10 3056085      3056281
chr10 3056384      3056580
chr10 3056737      3056897
chr10 3065807      3066031
chr10 3066240      3066729
chr10 3067218      3067525
chr10 3068012      3068311
chr10 3071975      3072324
chr10 3076937      3077074
chr10 3133965      3134132
chr10 3134477      3134660
chr10 3154200      3154469
chr10 3156352      3156576
chr10 3156714      3157241
chr10 3162918      3163363
```

```
2190000000
1470000000
1100000000
1690000000
877000000
884000000
787000000
1620000000
1360000000
1290000000
```

Supported genomes:

Yeast: sacCer3
Worm: ce11
Fly: dm3, dm6
Mouse: mm9, mm10
Human: hg19, hg38

```
turnover.exp.tdca3Dgenes.R
turnover.exp.tdca3Dgenes.pdf
turnover.exp.tdca.txt
turnover.exp.tdca.R
turnover.exp.tdca.pdf
```

Figure 15: Visual display of directory contents and files for TDCA input. (a) Contents of main directory where the user will run TDCA. The folders (rep1, rep2, input1, and input2) contain BAM files, appropriately named, that will be read by TDCA. (b)

Contents of the genes.txt file: RefSeq ID of five genes separated by a newline (\n) character. (c) Contents of the rep1 folder: 10 Bam files properly named (XXX_time.bam) with indices. The contents of the BAM folders rep2, input1, and input2 must contain the same number of BAM files with the same time points. (d) BED file containing 3 tab delimited column, chromosome, start, and end. There can be any number of loci here. (e) Depth matrix used to specify values to normalized BAM files to. There must be an equal number of integers separated by a newline (\n) character as there are BAM files in each folder. The integers are assigned to BAM files in chronological order. (f) Genomes supported by TDCA -g flag. (f) Hypothetical output files generated by TDCA. Along with these is a data folder that the Rscript reads in order to generate a pdf.

If the user wanted to include multiple replicates and input, such as:

```
	$tdca -bed loci.bed -bam rep1 -i input1 -bam rep2 -i input2 -dm
depth-matrix.txt -3D genes.txt -g mm9
```

The depth-matrix.txt file must contain the appropriate number of integers. Alternatively, the user can specify no depth matrix.

6.2 Getting data

During development, TDCA was tested on many published time dependent ChIP-seq data sets. One of the most robust experiments was performed using an inducible HA tagged H3.3. The accession number for project is GSE51505 (found at: http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE51505). The experiment was done in mouse MEF cells using 2 replicates on 11 time points, including input. In this next section we show users how to get this data and prepare it for TDCA processing. Alternatively, at the end of the section we offer users a link to this pre-processed data so that users can go straight to testing TDCA.

Accession numbers and names of data we will use for example:

GSM1246648	MEF_H3.3_0h_r1
GSM1246649	MEF_H3.3_1h_r1
GSM1246650	MEF_H3.3_2h_r1
GSM1246651	MEF_H3.3_3h_r1
GSM1246652	MEF_H3.3_4h_r1
GSM1246653	MEF_H3.3_5h_r1
GSM1246654	MEF_H3.3_6h_r1
GSM1246655	MEF_H3.3_12h_r1
GSM1246656	MEF_H3.3_18h_r1
GSM1246657	MEF_H3.3_24h_r1

GSM1246658	MEF_H3.3_48h_r1
GSM1246659	MEF_H3.3_72h_r1
GSM1246660	MEF_H3.3_0h_r2
GSM1246661	MEF_H3.3_1h_r2
GSM1246662	MEF_H3.3_2h_r2
GSM1246663	MEF_H3.3_3h_r2
GSM1246664	MEF_H3.3_4h_r2
GSM1246665	MEF_H3.3_5h_r2
GSM1246666	MEF_H3.3_6h_r2
GSM1246667	MEF_H3.3_12h_r2
GSM1246668	MEF_H3.3_18h_r2
GSM1246669	MEF_H3.3_24h_r2
GSM1246670	MEF_H3.3_48h_r2
GSM1246671	MEF_H3.3_0h_Input
GSM1246672	MEF_H3.3_1h_Input
GSM1246673	MEF_H3.3_2h_Input
GSM1246674	MEF_H3.3_3h_Input
GSM1246675	MEF_H3.3_4h_Input
GSM1246676	MEF_H3.3_5h_Input
GSM1246677	MEF_H3.3_6h_Input
GSM1246678	MEF_H3.3_12h_Input
GSM1246679	MEF_H3.3_18h_Input
GSM1246680	MEF_H3.3_24h_Input
GSM1246681	MEF_H3.3_48h_Input
GSM1246682	MEF_H3.3_72h_Input

Download the above SRA experiments and unpack using sra toolkit (Leinonen,R. *et al*. 2011) fastq-dump command. Data then needs to be aligned (Li H and Durbin R. 2009) to the mouse genome and peaks need to be called (Zhang,Y. *et al*. 2008). We used the following commands for this:

Align to mm9:
```
$bwa mem mm9.fa MEF_H3.3_0h_r1 > MEF_H3.3_72h_r1.mm9.mem_0.sam
```
Convert sam to bam:
```
$samtools view -bS MEF_H3.3_72h_r1.mm9.mem_0.sam >
MEF_H3.3_72h_r1.mm9.mem_0.bam
```
Sort bam:
```
$samtools sort MEF_H3.3_72h_r1.mm9.mem_0.bam
MEF_H3.3_72h_r1.mm9.mem.sorted_0.bam
```
Remove duplicates if any:
```
$samtools rmdup MEF_H3.3_72h_r1.mm9.mem.sorted_0.bam
MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup_0.bam
```

Create bam index:
```
$samtools index MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup_0.bam
MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup_0.bam.bai
```

Repeat this for each fastq file. Note that the bam files are named with the convention "XXX_integer.bam". This is naming convention is essential for TDCA to detect the time point in question. TDCA uses regex to do this and currently support only integer times in minutes. Call peaks using time point with longest treatment time - 4320 minutes (72 hours of doxycycline treatment).

Call broad peaks with macs2 (77531 peaks):

```
$macs2 callpeak -t MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup_4320.bam -c
MEF_H3.3_72h_input.mm9.mem.sorted.rmdup_4320.bam --broad -g mm -name
h3.3.72h-72hinput.macs2-broad.0.05 --broad-cutoff 0.05
```

The bed file required for TDCA input must contain 3 tab delimited columns: chromosome, start, and end for each peak. Copy and paste these into a text file from the macs2 xls output.

Once data is obtained, put all the bam files and indices with correct name extension for time points (XXX_integer.bam) in a folder for each replicate and input.

For example:
Make a directory for replicate 1 files:
```
$mkdir krauschaarr-rep1
```
Move files replicate 1 files to newly created directory:
```
$mv -t ./krauschaarr-rep1 MEF_H3.3_48h_r1.mm9.mem.sorted.rmdup_2880.bam
MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup.bam
MEF_H3.3_0h_r1.mm9.mem.sorted.rmdup_0.bam
MEF_H3.3_1h_r1.mm9.mem.sorted.rmdup_60.bam
MEF_H3.3_3h_r1.mm9.mem.sorted.rmdup_180.bam
MEF_H3.3_5h_r1.mm9.mem.sorted.rmdup_300.bam
MEF_H3.3_12h_r1.mm9.mem.sorted.rmdup_720.bam
MEF_H3.3_24h_r1.mm9.mem.sorted.rmdup_1440.bam
MEF_H3.3_48h_r1.mm9.mem.sorted.rmdup_2880.bam
MEF_H3.3_6h_r1.mm9.mem.sorted.rmdup_360.bam
MEF_H3.3_18h_r1.mm9.mem.sorted.rmdup_1080.bam
MEF_H3.3_2h_r1.mm9.mem.sorted.rmdup_120.bam
MEF_H3.3_4h_r1.mm9.mem.sorted.rmdup_240.bam
MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup.bam
MEF_H3.3_0h_r1.mm9.mem.sorted.rmdup_0.bam.bai
```

```
MEF_H3.3_24h_r1.mm9.mem.sorted.rmdup_1440.bam.bai
MEF_H3.3_4h_r1.mm9.mem.sorted.rmdup_240.bam.bai
MEF_H3.3_12h_r1.mm9.mem.sorted.rmdup_720.bam.bai
MEF_H3.3_2h_r1.mm9.mem.sorted.rmdup_120.bam.bai
MEF_H3.3_5h_r1.mm9.mem.sorted.rmdup_300.bam.bai
MEF_H3.3_18h_r1.mm9.mem.sorted.rmdup_1080.bam.bai
MEF_H3.3_3h_r1.mm9.mem.sorted.rmdup_180.bam.bai
MEF_H3.3_6h_r1.mm9.mem.sorted.rmdup_360.bam.bai
MEF_H3.3_1h_r1.mm9.mem.sorted.rmdup_60.bam.bai
MEF_H3.3_48h_r1.mm9.mem.sorted.rmdup_2880.bam.bai
MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup_4320.bam.bai
```

Repeat this for both replicates and input. If the user is working with the two replicates and input from Krauschaarr et al. (2013) then there should be three folders corresponding to the two replicates and and the input, each containing appropriately named am files and indices. The working directory should also contain a bed file of H3.3 peaks.

```
$ls
krauschaarr-rep1
krauschaarr-rep2
krauschaarr-input
H3.3.72h-72hinput.macs2-broad.0.05.chr10.bed
```

Note that the H3.3 data only has time point 4320 for replicate 1. TDCA will give an error if replicates have differing timepoints. So the name of the bam file for time point 4320 was changed from MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup_4320.bam to MEF_H3.3_72h_r1.mm9.mem.sorted.rmdup.bam. The absence of the “XXX_integer naming convention will make it invisible to TDCA.

Alternatively, we provide pre-aligned chromosome 10 data along with peaks for faster testing. This data can be found here: https://drive.google.com/open?id=0B5BFPUpdPrmhdG5jbkFUSlEtZDA

6.3 Running TDCA

These files now satisfy the basic input requirement to run TDCA. TDCA commands could be ran as follows:

Run replicate 1 with no genome specified (heatmap ideogram and gene features boxplot will not be created):

```
$tdca -bed H3.3.72h-72hinput.macs2-broad.0.05.chr10.bed -bam
krauschaarr-chr10-rep1/ -name r1.chr10.minimum
```

Run both replicates and input and specify genome:

```
$tdca  -bed H3.3.72h-72hinput.macs2-broad.0.05.chr10.bed -bam
krauschaarr-chr10-rep1/ -bam krauschaarr-chr10-rep2/ -i krauschaarr-chr10-input/ -i
krauschaarr-chr10-input/ -g mm9 -name r1.r2.input.chr10.mm9
```

7. FAQ

6.1 Installation fails for R package, “rgl” in Ubuntu environment because of X11 not found but required.

Run the following in command line, sudo apt-get install r-cran-rgl.

6.2 Generate pdf file without compiling tdca.

Once the R scripts are generated by tdca, run the following in command line, Rscript name_R_script. “xxx.tdca3Dgenes.R” is used to generate 3D graphs and “xxx.tdca.R” generates default graphs

6.3 Changing the look of output graphs.

R scripts are provided for each graphical output. Users may wish to change the look of certain graphs and can do so with a basic understanding or R and ggplot2. R scripts are generated in a modular fashion to facilitate single change options. Data generated from larger genomes may create extremely large R scripts and may not open well with all text editors. The format of all R scripts is the same given the same tdca flag calls. Thus R scripts can also be manipulated by line swapping using combinations of awk and cat or other commands.

6.4 What compiler do I need to install tdca?

TDCA is compiled using g++. Most later versions should work and version 4.9.3 has been tested exhaustively. C++ standard library 2014 (-std=c++14) is used in the TDCA Makefile, however C++ standard library 2011 also works. Users may change the -std=c++14 flag to -std=c++11 in the TDCA Makefile if they wish.

8. Runtime Dependencies

8.1 Number of Processors/BED File Peaks/ BAM files

TDCA is parallelized to run on all available processors. Runtime dependencies were tested using replicates from Krauschaarr et al. (2013) on the bugaboo server of westgrid computer cluster.

TDCA outputs the depth at non-peak loci for experiment files and the total depth of input files. If only 1 processor is in use, the order that the depth of the files are printed will be chronological. If, however, openmp is enabled, the files will be printed in an unpredictable way. This is one simple way to check if parallelization is working. Secondly, if parallelized, these files will be printed in clusters, rather than one by one in sequential blocks. When TDCA is running the drc R script, the user can check the directory that the program is running if there are R script named drcVersitile.X.R, where X is an integer, equal to the number of processors then parallelization is working. These are simple ways to ensure parallelization is working. Using openmp on cloud clusters may require additional efforts to ensure proper functioning. For example, on westgrid systems it is required that parallel jobs be ran on processors on the same node.

As shown in Figure 16, the run time of TDCA decreases as increased processing power is available.

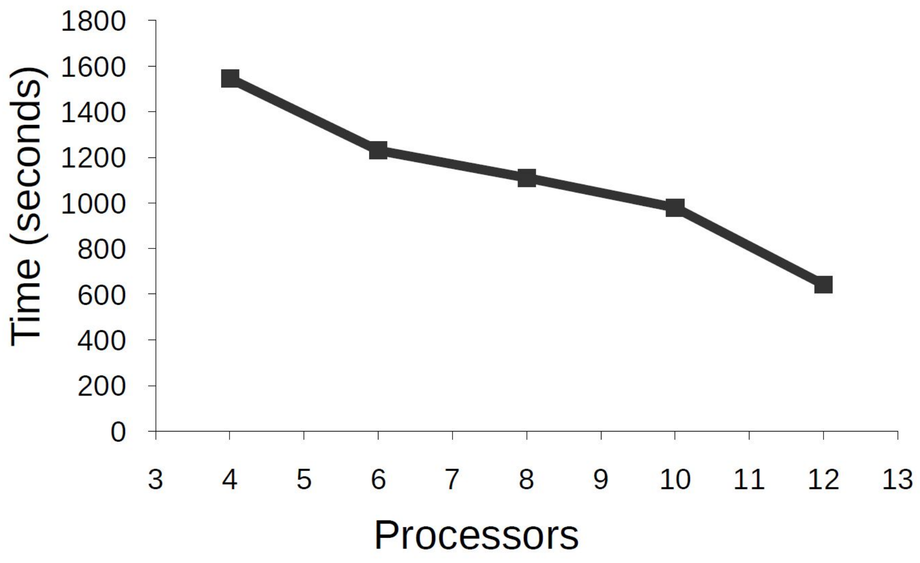


Figure 16: Processing time of H3.3 time course data using eleven time points on chromosome 10 loci (4180 loci) with 4, 6, 8, 10, and 12 processors. TDCA utilizes openmp to parallelize various algorithms in the program.

9. TDCA Support

Please submit bug reports and request for library expansions to: mmyschyshyn@gmail.com

10. References

Karolchik D., *et al*. (2004) The UCSC Table Browser data retrieval tool. *Nucleic Acids Res.* **1**, D493-6.

Kent,W.J. *et al*. (2002) The human genome browser at UCSC. *Genome Res*.**12**, 996-1006.

Kraushaar,D.C. *et al*. (2013) Genome-wide incorporation dynamics reveal distinct categories of turnover for the histone variant H3.3. *Genome Biol.*, **14**, R121.

Leinonen,R. *et al*. (2011) The Sequence Read Archive. *Nucleic Acids Res*. **39**, D19-D21.

Li H and Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **15**, 1754-60.

Li,H. *et al.* (2009) The Sequence alignment/map (SAM) format and SAMtools. *Bioinformatics*, **25**, 2078-9.

Wickham,H. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009.

Quinlan,A.R. and Hall,I.M. (2010) BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**, 841-842.

Ritz,C. *et al*. (2015) Dose-Response Analysis Using R. *PLos One*, **10**, e0146021.

Zhang,Y. *et al*. (2008) Model-based Analysis of ChIP-Seq (MACS). *Genome Biol.* **9**, R137.