# *StereoGene* version 2.50

User manual

August 31, 2025

## Contents

# 1 Introduction

High-throughput sequencing methods generate massive amounts of data. The most common first step in interpreting these data is to map the data to genomic intervals and then overlap them with genomic annotations. A major interest in computational genomics is spatial, genome-wide correlation between genomic features (e.g., between transcription and histone modification). The key hypothesis here is that features that are similarly distributed along a genome may be functionally related.

Here we propose the *StereoGene* method, which rapidly estimates the genome-wide correlation of two genomic annotations. The method allows correlation of continuous data, eliminating the data loss that occurs when reducing to intervals. To incorporate the analysis of non-overlapping but spatially related features, we use kernel correlation. Another novel and powerful feature of our approach is the output of local correlation traces that allow overlapping with other correlations (correlation of correlations).

## 1.1 Software availability

The *StereoGene* version 2.50 C++ source code, program documentation, Galaxy integration scripts and examples are available on the github repository `https://github.com/favorov/stereogene`.

The repository includes four folders:

- src is the source folder
- www contains a fresh copy of the project web page
- galaxy contains the stub files for Galaxy (https://galaxyproject.org/) integration
- example is a set of files for a minimalistic run

## 1.2 Source code and license

The software is provided in a form of C++ source code with a makefile for GCC compiler. After cloning or downloading and unpacking the source, say command **`<make>`** from the shell in the source (**`stereogene/src`**) folder.

Our source code is covered by MIT artistic license. We use FFT code by Jens Jorgen Nielsen, its license is shown in the mixfft.c source file.

## 1.3 Input and output

The program reads standard (bed, wigg, etc.) formats for the features to be correlated and a tab-delimited text file with chromosome name and its length pairs on each line as the chromosome length file. The output will be a set of text files. One pair of them (statistics and parameters) is a report that grows line by line with each start of *StereoGene* in the current working folder. Others are text files that follow a naming scheme (see below) that allows batch post-processing. On request (using the **plotType** parameter), *StereoGene* generates an R script that creates a human-readable report in pdf or html format. If Rscript is not defined PATH (usually in Windows platform) it should be referred in the **Rscript** parameter (for example: `Rscript="C:/Program Files/R/R-3.6.0/bin/x64/Rscript.exe"`)

## 1.4 Easy start

*StereoGene* has a lot of parameters that change the behavior of the software. However, all of them have reasonable defaults, actually a run requires two track files and a file with chromosome lengths. The following example is started from the example (**stereogene/example**) folder. First, copy (link) the *StereoGene* executable you've biult by **make** in the source folder to the example folder. Then, at a shell prompt, type

```
./stereogene chrom=chromLength -r H3K4me1.bed H3K4me1.bed
```

The first parameter refers to the file with the length of the chromosomes, the parameter -r tells the program to generate an R script for visualizing the results, other parameters define the input tracks.

For convinces some parameters can be collected in the configuration file (syntax of this file see below, section 4.2). To include the configuration file use parameter **cfg=<config_file>**.

For more complicated case see Section 1.6.

## 1.5 Directory structure

The program creates many files. To manage the files, the directory structure is recommended. The user can define several directories:

Table 1: Directoty structure

| Path | Description | |
|---|---|---|
| **trackPath=<path>** | Path to the input tracks. | The program will search input files here |
| **profPath=<path>** | Path to profiles | directory for temporary binary tracks |
| **resPath=<path>** | Path to the result files | directory for the results |
| **report=<path>** | Path to the report files (pdf, html) relative to the resPath | directory for pdf/html reports |

This directory structure can be defined in the configuration file. Example:

```
trackPath=./Tracks
profPath=./profiles
resPath=./res
report=report
```

## 1.6 More complicated case

The example uses the tracks `{H3K4me3.bed H3K9ac.bed H3K27ac.bed}` – active histone modifications and `{H3K9me3.bed, H3K27me3.bed}` – repressive modifications. We analyze the correlation of each active modification with each repressive modification.

1. Find the directory (**`stereogene/example/example-section-1.6`**). Use it as work folder or creata a new one anywhere and copy (**`stereogene/example/example-section-1.6/Tracks`**) there.

2. Now, your work folder has **`Tracks`** folder and subfolders `Tracks/active, Tracks/repress` for the active tracks and repressive tracks.

3. Download from `https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/hg38.chrom.sizes` chromosome length file as `hg38.chrom.sizes` to the working directory

4. Create the configuration file `stereogene.cfg`:

```
chrom=hg38.chrom.sizes
# define the chromosome file
trackPath=Tracks                # define path to the tracks
profPath=profiles               # define path to temporary binary profiles
resPath=res                     # define path to the results
report=report                   # define path to the reports
plotType=pdf                    # define type of the reports
```

5. copy (link) the *StereoGene* executable here. Run *StereoGene* : `$./stereogene active repress`

6. Find and browse the report files `*.pdf` in the directory `res/active/report`

7. Open the file statitics.tsv in Excel or OpenOffice or R and analyse the result

You can just run the built *StereoGene* without copying it, so you run something like `/stereogene/src/stereogene` from the folder where you built it.

## 1.7 How to look at the results

After running *StereoGene* for tracks **`ARID2.bgr`**, **`ARID1B.bgr`** (these are proteins from the SWI/SNF complex) with the configuration file:

```
chrom=chr_len_hg38.tab
trackPath=../Tracks
profPath=../profiles
resPath=../res
report=report

wSize=1M
bin=200
KernelSigma=3K
crossWidth=30000

plotType=pdf
```

We obtain the pdf result file (Figure 1).

**Input**

Track1 = ARID1B
Track2 = ARID2
Window Size (kb) = 1000
Kernel width = 3000
Bin Size = 200
Max zero (%) = 99.90
Max NA (%) = 99.90

**Results**

Output files = ../res/ARID/ARID1B~ARID2
n Foreground comparisons = 2935
n Background comparisons = 2935
Foregr.  correlation = 0.40
Foregr. correlation std.dev = 0.24
Backgr.  correlation = 6.61e−03
Backgr. correlation std.dev = 0.17
Mann−Witney p−value = 0.00e+00

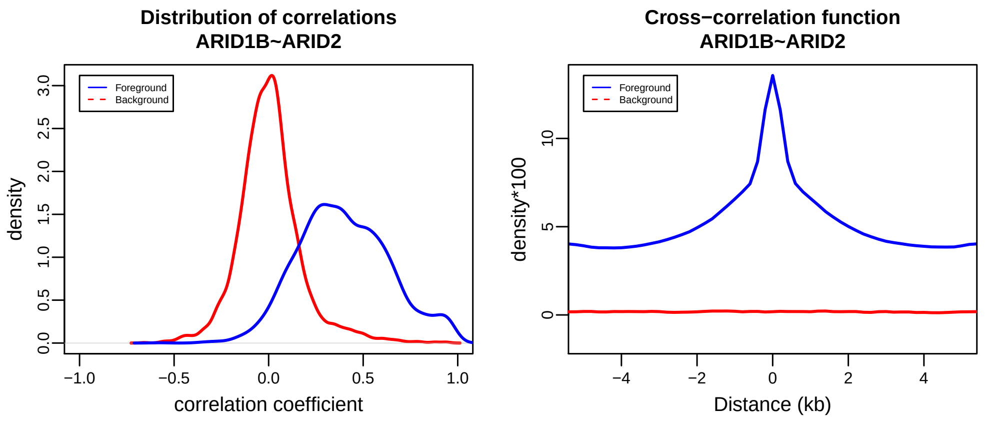


Figure 1: Result pdf file

The upper part of the figure 1 shows the input track names, the main parameters – window size and kernel width (left) and the results (right) – correlation coefficients for the track comparison (foreground) and for the background model. The next two plots are shown. The left plot shows the distribution of the correlation for the foreground (blue) and the background (red). The right plot shows the cross-correlations for the foreground and background.

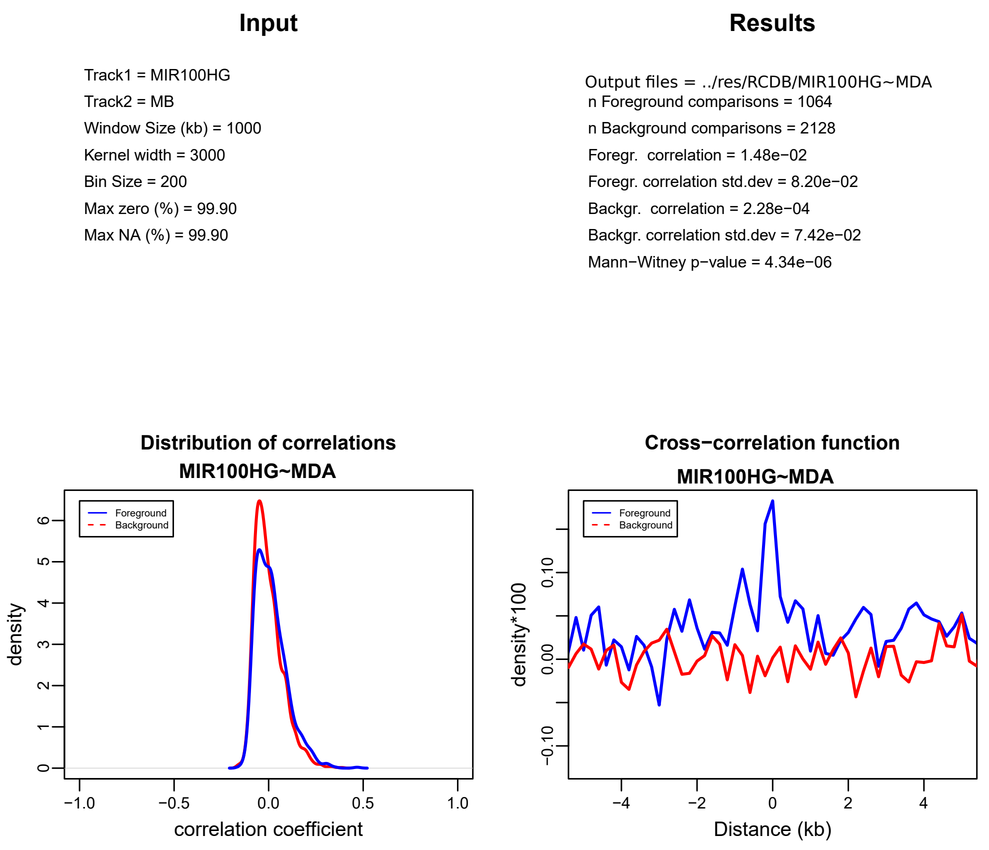
Looking at the plots, it is clear that we observe significant correlations here. Furthermore, the cross-correlation function shows that the dependence of the signals decreases with 1kb distance. This is very good case. But sometimes *StereoGene* find something like shown on the Figure 2

With a very good p-value ($4.3 \cdot 10^{-6}$), the correlation is very small ($1.48 \cdot 10^{-2}$). The distribution of the correlation coefficients is very close to the background distribution (left plot). The cross correlation plot (right) shows a small peak at zero. We can conclude that we are observing a very small but significant correlation. Usually this behavior reflects the fact that the data is sparse.

Another example is shown in the figure 3. Here we have a negative correlation, but the cross-correlation function has a sharp drop at zero. This means that two tracks avoid being together.

## 2 The calculation workflow

When started, the software first preprocesses the input data and then performs the requested calculations. The results of the pre-processing steps are stored to save computing time. The next runs of *StereoGene* on the same track can reuse the pre-processing results.



**Input**

Track1 = MIR100HG
Track2 = MB
Window Size (kb) = 1000
Kernel width = 3000
Bin Size = 200
Max zero (%) = 99.90
Max NA (%) = 99.90

**Results**

Output files = ../res/RCDB/MIR100HG~MDA
n Foreground comparisons = 1064
n Background comparisons = 2128
Foregr. correlation = 1.48e−02
Foregr. correlation std.dev = 8.20e−02
Backgr. correlation = 2.28e−04
Backgr. correlation std.dev = 7.42e−02
Mann−Witney p−value = 4.34e−06

Figure 2: Result pdf file

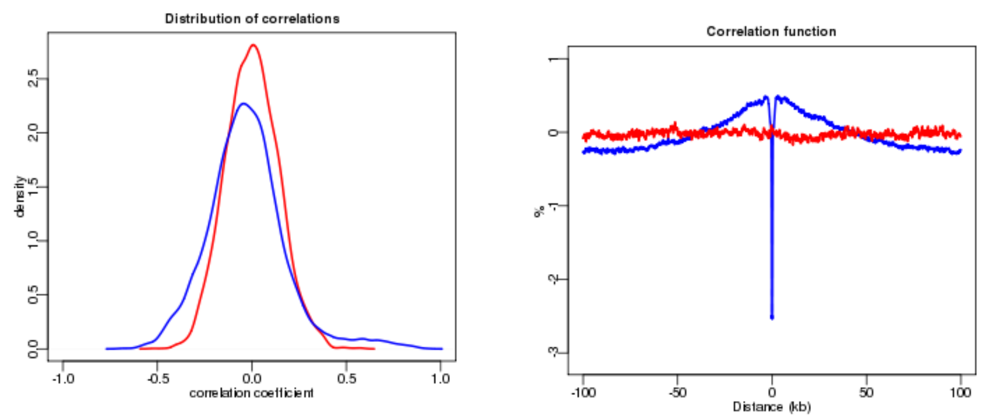


Figure 3: Result pdf file

## 2.1 Preprocessing

The input is read from the track text file in one of the standard genomic track formats (bed, wig, bedGraph, NarrowPeak, and broadPeak) and then converted into a proprietary binary format file for correlation analysis. The conversion averages all input profiles into small tiling windows (bins) defined by the `bin` parameter.

The BroadPeak and NarrowPeak formats standard defines a list of possible values for each nucleotide position – *score*, *signal*, *log-p-value*, and the user can specify which of this Peak data types are to be used (prmbpType parameter).

The program defines the file format using the file name extension:

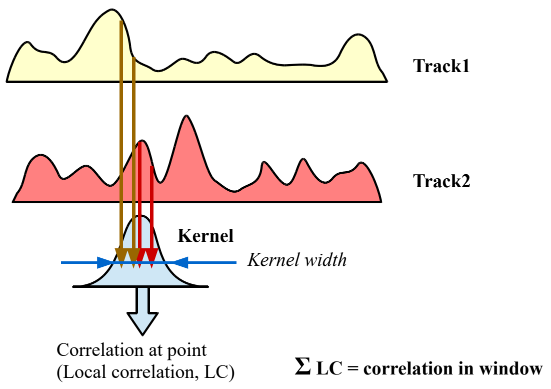
Table 2: Allowed file-name extensions for input tracks

| | |
|---|---|
| `bed` | Bed format |
| `wig` | Wigg format |
| `bedgraph` | Bedgraph format |
| `bed_graph` | |
| `b_graph` | |
| `bgr` | |
| `b_peak` | Broad Peak format |
| `bpeak` | |
| `broad_peak` | |
| `broadpeak` | |
| `n_peak` | Narrow Peak format |
| `npeak` | |
| `narrow_peak` | |
| `narrowpeak` | |
| `model` | Model file (see Model) |
| `mod` | |
| `mdl` | |
| `bigbed` | **Binary formats do not allowed!** |
| `bigwig` | |

The software allows users to combine several input tracks in a linear combination and use the results of the linear model as one of the tracks used for correlation analysis (see "Model" section).

## 2.2 Correlation calculation

*StereoGene* calculates the kerneled correlation, which means that it does not only consider the values of the tracks in the same position, but it integrates all nearby values with weights defined by the kernel. (Figure 4).



$$LC(x) \sim g(x)\int_G \rho(x-t)f(t)dt + f(x)\int_G \rho(x-t)g(t)dt$$

$$corr \sim \int_G\int_G \tilde{f}(x)\tilde{g}(y)\rho(x-y)dxdy$$

Figure 4: Kernelled correlation

Instead of correlating the whole genome, we compute the correlation of the preprocessed tracks within large windows of fixed size ( 1,000,000 bp). The size of these windows is specified by two input parameters (**`wSize`** and **`wStep`**). In these parameters, prmwSize is the width of the window and **`wStep`** is the distance between the start coordinates of each window. If **`wSize`** is equal to **`wStep`** (default), the windows are tiled and do not overlap. The windows are smaller than whole chromosomes to increase efficiency and to obtain a distribution of values from which to estimate correlation p-values. **`wSize`** is a technical parameter. If it is too large, there will be too few windows for statistical estimation and the program will be too slow; however, if **`wSize`** is too small, we will get a very wide variety of local correlation measures instead of a common whole-genome correlation. We recommend using **`wSize`** = **`wStep`** = 100k..10M (Figure 5A).

The data on the input tracks is routinely gapped. Some wig files contain zeros. They also contain genomic regions that are not covered by data. There are several ways to interpret this missing data. For example, the coverage of a region could be truly unknown due to a mapping problem. On the other hand, the level could really be 0. Depending on how we interpret the empty areas (NA parameter), we either write zeros in these areas or cover them with random values defined by the **`noiseLevel`** parameter. The data written to the profile is defined by the

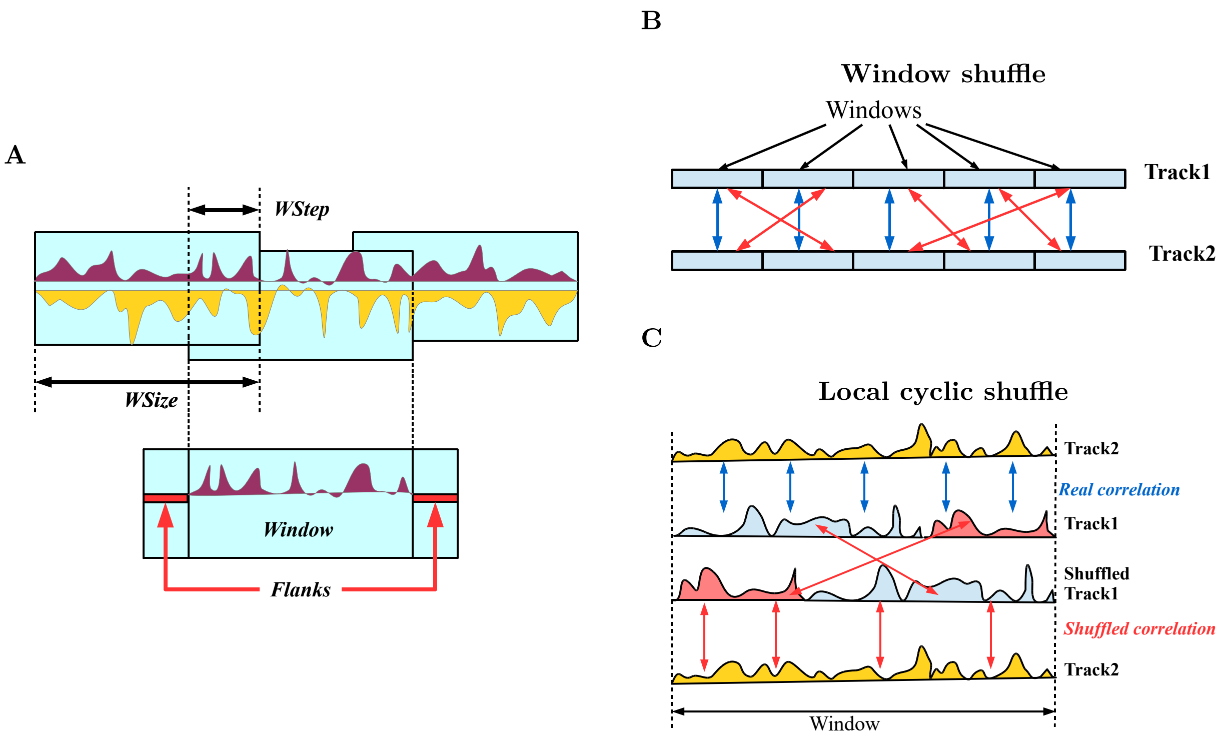


Figure 5: Window size and step parameters (A) and shuffling procedure (B,C)

formula $x = Norm(e, noiseLevel * sd)$, where Norm - is a random normally distributed value, e, sd - mean and standard deviation of the input data. If the input data is very sparse, the windows containing only 0 or NA values will appear. The correlation in these windows is undefined, so the program ignores them. There are parameters (**`maxZero`**, **`maxNA`**, percents) that define which windows should be treated as non-informative.

If the DNA strand is defined (e.g. for genes), the **`complFg`** parameter indicates whether the information is included in the correlation. The finite size of the windows may cause edge effects; to avoid this, the user can ask the program to add random signal flanks to each window (**`flankSize`** parameter). However, our experience shows that for large window sizes (e.g., the default 100,000 bp), the flanks have minimal effect on the correlation.

It is challenging to analytically estimate the null distribution of correlation coefficients that takes into account the dependencies of all features at neighboring positions of the genome. We use a permutation to test the statistical reliability of the result. To do this, we randomly sample pairs of windows, first from one track, then from the other, and evaluate the correlation value for each pair (Figure 5B). The number of sampled pairs is determined by the parameter **`nShuffle`**.

Another sparse data approach based on cyclic permutation of non-zero windows. It uses all the features of the standard approach. Instead of comparing different windows, a random cyclic permutation of the tracks is used to compute the background distribution. To use this option, set the parameter **`localShuffle`**=1. This option is useful for sparse data (Figure 5C).

List of calculation parameters shown in the Table 3

Table 3: Calculation parameters

| Parameter | Sense | default value |
|---|---|---|
| **`bin`** | Bin size. All the tracks are bined with this value. | 100 |
| **`wSize`** | Window size. Correlation will be done in window. | 300000 |

| | | |
|---|---|---|
| `wStep` | Window step. Windows Shift. | 0 |
| `noiseLevel` | level of the noise to fill NA values | 0 |
| `maxZero` | Maximum number of zeros in the window (percent) | 99 |
| `maxNA` | Maximum number of NA in the window (percent) | 99 |
| `complFg = <COLLINEAR \| COMPLEMENT \| IGNORE_STRAND>` | Flag:<br>IGNORE_STRAND – ignore strand;<br>COLLINEAR – compare collinear strands;<br>COMPLEMENT – compare opposed strands.<br>If the binary profiles are not separated by strand that works in IGNORE_STRAND | `IGNORE_STRAND` |
| `flankSize` | Size of added flanks (in nucleotides). Necessary to avoid side effects. Recommended value $2 \cdot KernelSigma$ | 0 |
| `KernelSigma` | Width of the kernel | 1000 |
| `nShuffle` | Number of shuffle to calculate the background distribution | 10000 |
| `localShuffle` | Use local shuffle instead of window shuffle | 1 |

## 3 Input and output

### 3.1 Input

Here we present the input data in more detail. The basic input file is a track file in one of the formats listed in the table 2. The program can accept two or more tracks as input. If more than two tracks are defined, all possible track pairs are analyzed (figure 6A).

To analyze many tracks, the program can use a batch mode. There are two ways of using this option. First, you can define a list of tracks as input (6B,C). Format of the list file – each line define input track. The filename extension of the list file should be *.list or *.lst. Second, you can define the directories as input. In this case, all allowed files in each directory will form a list.

A.

| | Tr1 | Tr2 | Tr3 |
|---|---|---|---|
| Tr1 | ▨ | Cor | Cor |
| Tr2 | ▨ | ▨ | Cor |

B.

| | | list1 | | |
|---|---|---|---|---|
| | | Tr1 | Tr2 | Tr3 |
| list1 | Tr1 | ▨ | Cor | Cor |
| | Tr2 | ▨ | ▨ | Cor |

C.

| | | list1 | | list2 | | list3 | |
|---|---|---|---|---|---|---|---|
| | | Tr1 | Tr2 | Tr3 | Tr4 | Tr5 | Tr6 |
| list1 | Tr1 | ▨ | ▨ | Cor | Cor | Cor | Cor |
| | Tr2 | ▨ | ▨ | Cor | Cor | Cor | Cor |
| list2 | Tr3 | ▨ | ▨ | ▨ | ▨ | Cor | Cor |
| | Tr4 | ▨ | ▨ | ▨ | ▨ | Cor | Cor |

Figure 6: Pairs of tracks for which the correlation is calculated in different inputs. A. 3 input tracks defined. B. one list defined. C. Three input lists are defined

### 3.1.1 The chromosome file

The chromosome file is necessary to run *StereoGene* . It is a tab separated file. First field – chromosome name, second field – chromosome length.

**Important! The chromosome names should be the same as in the track files.**

Example:

```
# Human Genome hg38
chr1 248956422
chr2 242193529
chr3 198295559

...
```

## 3.2 Output

The results are written to the results directory. The results of the comparison are written to the files whose names are constructed from the input tracks by the rule

$$< \text{outFileName} >=< \text{trackName1} >\sim< \text{trackName2} >$$

where `trackName1`, `trackName2` are the names of the track files without extensions. Output files are listed in the Table 4, See for details Section 9.

The *StereoGene* results are also written to the cumulative files '`statistics.tsv`', '`statistics.xml`', '`params.tsv`'. Each *StereoGene* run has a unique ID. The cumulative files refer to the run ID. File '`statistics.tsv`' is the tab-delimited file with the main information of the run – run ID, date/time, input file names, correlation coefficients for tracks and for shuffled tracks, p-value, important parameters `wSize` and `kernelSigma`. The file `statistics.xml` contains the same information in xml format. The `params.tsv` file contains other parameters.

Table 4: Output files

| Output File | Description | Parameter |
|---|---|---|
| `<trackName>.spect` | Average spectrum density | `outSpectr=1` |
| `<trackName>.auto` | Autocorrelation function for tracks | `AutoCorr=1` |
| `<outFileName>.bkg` | Background distribution for correlation function (shuffled frames). | |
| `<outFileName>.fg` | Foreground distribution for correlation function (coherent frames). | |
| `<outFileName>.dist` | Distance distributions (cross-correlation function) | `Distances=1` |
| `<outFileName>.chrom` | Statistics for chromosomes – average input signal values for the tracks and average window correlation. | `outChrom=1` |
| `<outFileName>.wig` | Standard wig-file with Local Correlations | `outLC=1` |
| `<outFileName>.LChist` | Histogram for the distributions of the Local Correlation | `outLC=1` |
| `<outFileName>.r` | R script for visualization of the results | `-r , plotType=R` |
| `<outFileName>_pdf.r` | R script for pdf report generation | `plotType=pdf` |
| `<outFileName>_html.r` | R script for html report generation | `plotType=html` |
| `<report>/<outFileName>.pdf` | report in the pdf format | `plotType=pdf` |
| `<report>/<outFileName>.html` | report in the html format | `plotType=html` |
| `<report>/<outFileName>.svg` | plots for the html report | `plotType=html` |

Continued on next page

**Table 4 – continued from previous page**

| Output File | Description | Parameter |
|---|---|---|
| `statistics` | Summary file with the results. For each run it contains: the names of the source files, the name for the resulting file, the parameters of the Mann-Whitney statistics . | `./statistics` |
| `parameters` | Summary file with the parameters of runs. Correspondence with the statistics is established using id. | `./params` |
| `log` | a log-file | |

## 4 Parameters

### 4.1 Passing parameters to the program

The parameters are passed to the program by a command line. The parameter can be defined as

**`<paramName>=<value>`**

no space character allowed or as

**`-<paramName> <value>`**

Example of a command line (parameters definition with '=')

```
$./StereoGene H3K4me1.wig H3K4me3.wig chrom=a.sizes log=sgLog -v
```

Or equivalent command line

```
$./StereoGene H3K4me1.wig H3K4me3.wig -chrom a.sizes -log sgLog -v
```

### 4.2 Configuration file

For convenience, the parameters can be defined in a configuration file. The reference to the configuration file can be defined in the command line attribute **`cfg=<fname>`**. There is a default configuration file name **`stereogene.cfg`**. If this file exists, *StereoGene* will take the parameters from it first and does not require to define the configuration file on the command line. The program can take multiple configuration files as input. In this case, are parsed in the following order:

**`stereogene.cfg < config1 < config2 <... < Command_line`**

The parameter value can be overridden by configuration files in command line order. Finally, the parameter can be overridden from the command line.

Every line in the configuration file has the format:

**`<parameter> = <value>`**

The '#' symbol is treated as a comment. Empty lines and lines starting with '#áre ignored. Example configuration file:

```
chrom=../data/chrhg38          # define the chromosome file
trackPath=./Tracks             # define path to the tracks
profPath=./profiles            # define path to temporary binary profiles
resPath=./res                  # define path to the results
report=report                  # define path to the reports
```

```
plotType=pdf                      # define type of the reports
bin=500                           # bin size
wSize=1M                          # Window size
KernelSigma=10k                   # Kernel width
#KernelSigma=5000                 # this is a comment line and has no effect
verbose=1                         # The program will show the calculation process
#-------------------------------- define where is the Rscript -------- this is a comment line
Rscript="C:/Program Files/R/R-3.6.0/bin/x64/Rscript.exe"
```

Note that the value **`wSize`** is defined as 1M and **`KernelSigma`** is defined as 10k. It is possible to use the symbols 'k' and 'm' to define thousands and millions.

## 5 Advanced features

### 5.1 Aliases

Sometimes the input file names are quite long, for example **`MIR100HG_GRID_MDA_MB_231`**, **`GRID_MDA_MB_231_sorted`**. The output file will be named

**`MIR100HG_GRID_MDA_MB_231~GRID_MDA_MB_231_sorted`**

It is quite difficult to work with such long names, especially in batch mode. For convenience, *StereoGene* can change the output file name by replacing parts of the name. To do the substitution, *StereoGene* uses the aliases table defined by the **`aliases`** parameter. The format of the file is :

**`Old_name_fragment=new_name_fragment`**

For example:

```
MIR100HG\_GRID\_MDA\_MB\_231=MIR100HG
GRID\_MDA\_MB\_231\_sorted = GRID\_s
```

If this table is used, the output files will be named as

**`MIR100HG~GRID_s`**

### 5.2 Confounder

A positive correlation between tracks may mean that both tracks are dependent on a third track, the confounder. Such a confounder could be nucleosome positioning or chromatin accessibility. To eliminate the confounding effect, a partial correlation can be used. To compute the partial correlation, the data should be projected onto the hyperspace orthogonal to the confounder data.

$$Proj = a - c \cdot \frac{(a \cdot c)}{(c \cdot c)}$$

where $a$ – the data, $c$ – confounder. To eliminate the confounder effect, use the *StereoGene* track projection on the confounder track:

$$f'(x) = f(x) - c(x) \cdot \frac{\int_G (f(t) \cdot c(t))dt}{\int_G (c(t) \cdot c(t))dt}$$

and the correlation is calculated using projections:

$$corr \sim \int_{x,y \in G} f'(x) \cdot g'(x) \rho(x-y) dxdy$$

To define the confounder track parameter **`confounder`**.

## 5.3 Model

The program can take not only a single track file, but a combination of tracks - a model. The model is defined in the model file (*.mod or *.model). The file contains a formula for calculating the model tracks. Each track should be included in the square brackets ([track_name]). The track in the model is a function of position. The track term in the model can be defined explicitly as a function of position, such as [K4me3](x+1000), or implicitly as [K4me3], i.e. [K4me3](x). The user can use arithmetic operations and functions (log, exp, sin, cos, tan, sqrt, abs, sign, atan) You can also use the internal variables and numbers. The variable '$x$' means the absolute position on the genome. An example of the model description:

```
K27=[H3K27m3.wig];
K4=[H3K4m3.wig](x+1000);
K27*K4;
```

Here K27 and K4 are internal variables; $x$ – genome position. The formula can be presented in a single line as well in multiple lines. This model is equivalent to the following model:

```
[H3K27m3.wig](x) * [H3K4m3.wig](x+1000)
```

This model means – take value of H3K27m3 and multiply it by the value of H3K4m3 shifted by 1k.

Another example of the profile binarization model with threshold 20:

```
K4=[H3K4m3.wig];
1+sign(K4-20)
```

Example of normalization of the H3K4me3 track by the INPUT track:

```
[H3K4me3.wig]/(INPUT + 0.01)
```

where 0.01 – a pseudocount.

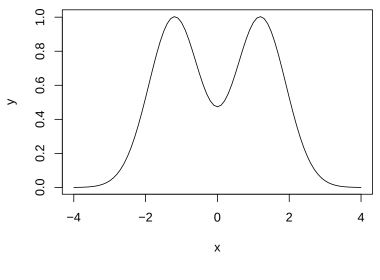
Usage of the model decreases program speed about twice.

## 5.4 Custom kernel

The default algorithm uses the Gaussian kernel function with defined mean (**`kernelShift`**) and standard deviation (**`kernelSigma`**), but the user can define a custom kernel function using the customKern parameter. The user can put here any expression containing arithmetic operations and standard functions: sin, cos, tan, exp, log, sqrt, abs, sign. Two variables are predefined by the corresponding parameters $e$=**`kernelShift`**, $sigma$=**`kernelSigma`**. A formula can contain expressions that define internal variables. These expressions should be separated by ';'. The expression without '=' gives final value. Example:

```
customKern="y1=(x-e)/(2*sigma); y2=(x+e)/(2*sigma); exp(-y1*y1)+exp(-y2*y2)"
```

This kernel is a bimodal combination of Gaussians with standard deviation **`kernelSigma`** shifted left and right by **`kernelShift`**:



# 6 Additional programs

## 6.1 Biner

The program takes input track(s) and produces binned output track(s). Output tracks have bedGraph format. The name of the output is formed using the input name by adding binsize: **`<output>=<input_fname>_<binsize>`**. The results will be written to the directory defined by **`binPath`**.

Standard configuration file is **`stereogene.cfg`**. Example of the configuration file:

```
chrom=chr_hg38.tab
trackPath=Tracks
binPath=Bin

bin=200
verbose=1
```

**Usage: `$ ./Binner <input_fname>`**
Standard batch conventions are used. User can use list files and directories

## 6.2 Soother

The program takes a track and produces a smoothed track. The smoothing is done with given kernel (default – Gaussian).

$$f_{smooth}(x) = \int f(t)\rho(x-t)dt$$

The kernel width (smoothing width) defined by the **`kernelSigma`** parameter. The resulting filename will have the suffix "**`_sm`**". The smoothed tracks are written to the directory **`smoothPath`**. If the parameter **`smoothPath`** is not defined, the results are written to **`trackPath`**. To avoid boundary effects, the flanks are recommended for this program. The default configuration file is **`stereogene.cfg`**. Example configuration file:

```
chrom=chr_hg38.tab
trackPath=Tracks
smoothPath=Smooth

bin=200
wSize=1M
kernelSigma=10k
flankSize=50k
verbose=1
```

**Usage: `$ ./Smoother <input_fname>`**
Standard batch conventions are used. User can use list files and directories

## 6.3 ParseGenes

It is often interesting to correlate some ChIP-seq data with gene features. To facilitate this comparison, the **`parseGenes`** program parses the gtf file and produces 9 bed files:

| | |
|---|---|
| Gene body | **`<gtf_file>_gene.bed`** |
| Gene begins | **`<gtf_file>_g_beg.bed`** |
| Gene ends | **`<gtf_file>_g_end.bed`** |
| Exon body | **`<gtf_file>_exn.bed`** |
| Exon begins | **`<gtf_file>_e_beg.bed`** |
| Exon ends | **`<gtf_file>_e_end.bed`** |
| Intron body | **`<gtf_file>_ivs.bed`** |
| Intron begins | **`<gtf_file>_i_beg.bed`** |
| Intron ends | **`<gtf_file>_i_end.bed`** |

All this files will be written to the directory defined by the **`trackPath`** parameter. Program parameter **`gencodeLevel`** defines the maximum confidence level of the GENECODE annotation (`https://www.gencodegenes.org/pages/data_format.html`) to be output by the program:

- 1 (verified loci),
- 2 (manually annotated loci),
- 3 (automatically annotated loci)

Program parameter **`biotypes`** – comma separated string of biotypes to be printed by the program. A complete list of biotypes can be found on the GENECODE web site `https://www.gencodegenes.org/pages/biotypes.html`. Example configuration file:

```
chrom=chr_hg38.tab
trackPath=Tracks

biotypes="lncRNA,lincRNA,protein_coding"
pgLevel=1
```

**Usage: `$ ./ParseGenes <input gtf file>`**

## 7 Make confounder

### 7.1 Program Confounder

One of the possible confounders in histone modification analysis is nocleosome positioning. To find this confounder, we use the following algorithm. The program **`Confounder`** takes the set of all available histone modification and creates a covariation matrix $Cov$. Then the program searches for eigenvector $ev^{(1)}$ corresponding to the maximal eigenvalue.

Then, the confounder track is calculated as a linear combination of the tracks with the coefficients defined by the first eigenvector:

$$cnf = \sum ev_i^{(1)} \cdot track_i$$

Different tracks may have different scales. To eliminate such offsets, the tracks in this equation are normalized by the average value of the track.

The Confounder track is placed in the **`trackPath`** directory. Its name is defined by the **`confounder`** parameter. The default name is **`confounder.bgraph`**. The program also creates a file **`confounder.covar`** with the covariance matrix and the first eigenvector. This file is placed in the current working directory.

**Usage: `$ ./Confounder <File_list>`**
where **`<File_list>`** – list of the tracks or directory.

## 7.2 Program Projector

*StereoGene* can produce a projection orthogonal to the confounder. But for batch analysis it seems to be more convenient to prepare a projection files in advance. Program **`Projector`** makes such projections and creates a projection files. It adds new subdirectories **`<confounder_name>.proj`** to the **`Tracks`**, **`Profiles`** and **`Results`** directories. Then the program creates projections and places them in appropriate directories. The program creates a new configuration file **`confounder.cfg`** with modified parameters for directories and the parameter **`confounder`** is removed.

Old configuration

```
chrom=chr_hg38.tab
chrom=../chr_len_hg38.tab
trackPath=../Files/Tracks
profPath=../Files/profiles
resPath=../Files/res
report=report
confounder=confounder.bgraph

wSize=1M
bin=200
KernelSigma=30k
nShuffle=2000
maxNA=99.9
maxZero=99.9
localShuffle=1

plotType=pdf
crossWidth=30000
```

New configuration

```
chrom=chr_hg38.tab
chrom=../chr_len_hg38.tab
trackPath=../Files/Tracks/confounder.proj/
profPath=../Files/profiles/confounder.proj/
resPath=../Files/res
report=report
#confounder=confounder.bgraph

wSize=1M
bin=200
KernelSigma=30k
nShuffle=2000
maxNA=99.9
maxZero=99.9
localShuffle=1

plotType=pdf
crossWidth=30000
```

**Usage:** **`$ ./Projector <File_list>`**
where **`<File_list>`** – list of the tracks or directory.

# 8 Comprehensive list of parameters

Table 5: Full List of parameters

| Parameter | Description | Default value |
|---|---|---|
| | **Paths** | |
| **`trackPath=<path>`** | Path to the input tracks | |
| **`binPath=<path>`** | Path to the output binned tracks (See section 6.1) | |
| **`smoothPath=<path>`** | Path to the output smoothed tracks (See section 6.2) | |
| **`profPath=<path>`** | Path to profiles | |
| **`resPath=<path>`** | Path to the result files | |

Continued on next page

**Table 5 – continued from previous page**

| Parameter | Description | Default value |
|---|---|---|
| `report=<path>` | Path to the report files (pdf, html) relative to the resPath | |
| | **Files** | |
| `<fname>` | Input file. The name of the appropriate path is added - trackPath. The Program can take up to 256 input files and calculate the correlation for every pair of the tracks. | mandatory |
| `chrom=<fname>` | Input file with names of chromosomes and their lengths (can ne downloaded from the Genome browser), the parameter can be set in the config file or in the command line or as an environment variable SG_CHROM. | mandatory |
| `cfg=<fname>` | Config file. It is possible to use more then one cfg-files. It is applied in order of appearance. Parameters defined in the command line overrides the parameters in the config file. | `./stereogene.cfg` |
| `confounder=<fname>` | The filename for confounder. The name is completed with trackPath. For more details see Advanced features. The file can be produced by the 'confounder' program or defined by user. See Section 5.2 | |
| `statistics=<fname>` | File with the cumulative statistic for all the programm runs. | `./statistics` |
| `params=<fname>` | File for save the parameters for all the programm runs | `./params` |
| `aliases=<fname>` | File with aliases for brief track names. See Section 5.1 | |
| `log=<fname>` | Log-file name. To toggle log output OFF use empty name for the log file. If the user starts the log-file name with '$' the '$' character will be replaced by the current output filename. | `./stereogene.log` |
| | **General** | |
| `verbose=<ON\|OFF>` | The program workflow is printed to standard output | `OFF` |
| `-v` | Is used for command line only. Equivalent to verbose=ON | |
| `silent=<ON\|OFF>` | If ON the general results are not printed to stdout. | `OFF` |
| `-s` | Is used for command line only. Equivalent to silent=ON | |
| `syntax=<ON\|OFF>` | If this flag is set ON the program will terminate with error on wrong line in input profile; the line will be ignored otherwise | `ON` |
| `-syntax` | Is used for command line only. Equivalent to syntax=OFF | |

Continued on next page

**Table 5 – continued from previous page**

| Parameter | Description | Default value |
|---|---|---|
| `BufSize` | Size of the buffer in swap-file-arrays | 100000000 |
| | **Preprocessing** | |
| `clear=<0 \| 1>` | Force profile preprocessing | 0 |
| `-c` | Is used in command line only. Is equal to clear=1 | |
| `bin = <num>` | bin for sliding frame. Optional. | 100 |
| `bpType=< SCORE \| SIGNAL \| LOGPVAL >` | The value used as a score for BroadPeak format – column Score, Signal, or pval | `SCORE` |
| | **Correlation** | |
| `wSize=<num>` | Window size (in nucleotides) | 100000 |
| `wStep=<num>` | Window step (in nucleotides) | 100000 |
| `kernelType=< NORMAL \| LEFT_EXP \| RIGHT_EXP >` | Kernel type:<br>NORMAL – Normal<br>LEFT_EXP – left exponent<br>RIGHT_EXP – right exponent | `NORMAL` |
| `customKern = <expression>` | An expression that defines the kernel function. See the section Section 5.4. If the custom kernel defined the prameter kernelType will be ignored. | |
| `KernelSigma=<num>` | Kernel span (in nucleotides) | 1000 |
| `kernelShift=<num>` | Kernel mean (for Gauss) or Kernel start for exponent | 0 |
| `kernelNS=<num>` | Inhibit Zero position of the kernel. The kernel values in the interval [-100,100] will be multiplied by (1-kernelNS) | 0 |
| `complFg=< COLLINEAR \| COMPLEMENT \| IGNORE_STRAND>` | Flag:<br>IGNORE_STRAND – ignore strand;<br>COLLINEAR – compare collinear strands;<br>COMPLEMENT – compare opposed strands.<br>If the binary profiles are not separated by strand that works in IGNORE_STRAND | `IGNORE_STRAND` |
| `NA=<0 \| 1>` | The way to treat uncertain values. If flag is 0, then uncertain (not covered with profile) values are considered as 0. Otherwise it is filled by a random noise (see noiseLevel parameter). This flag has no sense for BED tracks. For this type the flag is ignored and uncertain values are considered as 0. | 0 |
| `-na` | Is used in command line only. Is equal to NA=1 | |

Continued on next page

Table 5 – continued from previous page

| Parameter | Description | Default value |
|---|---|---|
| `flankSize=<num>` | Size of added flanks (in nucleotides). Nesessary to avoid side effects. Recommended value $2 \cdot KernelSigma$ | 0 |
| `noiseLevel=<num>` | Noise level used to fill flanks and uncertain values. The uncertain values is defined as $rExp() \cdot sd \cdot noiseLevel$, where rExp() – standard exponential random value, sd – standard deviation of track values. | 0.2 |
| `maxNA=<num>` | Maximum fraction of uncertain values in a frame (percent). If at least one of the compared frames has maxNA (or more) of uncertain values, then the comparison is not applied. | 99 |
| `maxZero=<num>` | Maximum fraction of zero values in a frame (percent). If at least one of the compared frames has maxZero (or more) of zero values, then the comparison is not applied. If parameter the takes 100, it is possible to get false peak for correlation value 1. | 99 |
| `nShuffle=<num>` | Number of permutations to build the background distribution. | 10000 |
| `threshold=<num>` | The threshold values for the binary profile. If the binary value is less than the threshold profile, it is replaced by 0. $0 < \text{threshold} < 256$ | 0 |
| `localShuffle` | Use cyclic window shuffle | 1 |
| | **Output definition parameters** | |
| `outRes=< XML \| TAB \| BOTH \| NONE>` | Output cumulative result and parameters:<br>XML – output in xml-like format<br>TAB – output in tabbed form<br>BOTH – output in both format<br>NONE – not output cumulative information | `BOTH` |
| `outPrjBGr=<0\|1>` | Flag: write bed-graph files for Projections. | 1 |
| `outDistr=<0\|1>` | Flag: write foreground and background distributions into files **`<track1>`∼`<track2>.fg`**, **`<track1>`∼`<track2>.bkg`** | 1 |
| `aliases=<file>` | Filename with rules for replacing track names in output filenames (see section 5.1) | |
| `outChrom=<0\|1>` | Write statistics by chromosomes (file **`<track1>`∼`<track2>.chrom`**) | 0 |
| `Cross=<0\|1>` | Flag: write Distance correlations (file **`<track1>`∼`<track2>.dist`**) If the parameter outChrom is ON the distance information will be given by chromosomes. | 1 |
| `WriteDistr=<NONE \| SHORT \| DETAIL>` | Write the Foreground and Background correlation distributions. If the value set to 'DETAIL' the positional information for foreground will be written. | `SHORT` |

Continued on next page

**Table 5 – continued from previous page**

| Parameter | Description | Default value |
|---|---|---|
| `outSpectr=<0\|1>` | Write average spectrum. File `<track1>`∼`<track2>.spect` | 0 |
| `AutoCorr=<0 \| 1 >` | Write autocorrelation. The autocorrelation function is written to a file Track.auto. If lAauto = 0, the autocorrelation is not considered. | 0 |
| `plotType=<NONE \| R \|PDF \| HTML \| ALL>` | Write the script to the R to plot the results. The name of the script file is generated from the output file name. Values mean:<br>R – R script for just plotting<br>PDF – R script for producing pdf file<br>HTML – R script for producing HTML report | 0 |
| `-r` | Is used in command line only. Equals to plotType=R | |
| `crossWidth` | Define width for cross-correlation plots | 10000 |
| `Rscript` | Command to run R script. If the parameter defined the program will call R automatically. | Rscript |
| `plotW` | Plot width | 4 |
| `plotH` | Plot height | 3 |
| `OutLC=<0 \| BASE \| CAENTER>` | Flag: Write local kerneled correlations into a bedgraph file. | 0 |
| `-lc` | Is used in command line only. Equals to outLC=BASE | |
| `LCScale=<LIN\|LOG>` | Scaling for the Local Correlation output:<br>LIN – Use linear scaling to the interval (0..1000)<br>LOG – Use logarithmic scaling:<br>$v = sign(w) * \log(1 + \|w\|)$ | `LOG` |
| `L_LC = <num>` | Threshold for local kernel correlation for low (usually negative) local correlations to be written into the local correlation file: $corr \leq L\_LC$ will be written. | -20 |
| `R_LC = <num>` | Threshold for local kernel correlation for high local correlations to be written into the local correlation file: $corr \geq R\_LC$ will be written. | 1 |
| `gencodeLevel=<1\|2\|3>` | Maximal confidence level in the GENCODE gtf file. See section 6.3 | 2 |
| `biotypes= <string>` | list of biotypes to be printed by ParseGenes program. See section 6.3 | |

# 9 Output file formats

## 9.1 Background correlations `<track1>`∼`<track2>.bkg`

`<track1>`∼`<track2>.bkg` – Array of correlations of shuffled windows

```
 0.320268
 0.178365
-0.419948
-0.124017
 0.298771
   ...
```

## 9.2 Correlations in coherent windows `<track1>~<track2>.fg`

Output depends on the parameter **`writeDistr`**.

### 9.2.1 Short format (`writeDistr=SHORT`)

```
 0.320268
 0.178365
-0.419948
-0.124017
 0.298771
   ...
```

### 9.2.2 Detailed Format (`writeDistr=DETAIL`)

```
chr1     7500000       7500100    -0.093813
chr1     7750000       7750100    -0.531958
chr1     8000000       8000100    -0.425593
chr1     8250000       8250100    -0.479256
chr1     9750000       9750100    -0.466759
...
```

**columns:** Chromosome, window start, window end, correlation

## 9.3 Distances distribution (correlation function) `<track1>~<track2>.dist`

**`<track1>~<track2>.dist`** – Distances distribution (cross-correlation function) for background, foreground, chromosomes. Output depends on the parameter **`outChrom`**.

### 9.3.1 Sort format (`outChrom=0`)

| | | |
|---|---|---|
| | dist | – Distance (nucl) |
| | Bkg | – background cross-correlation function |
| **Contains:** | Fg | – foreground cross-correlation function |
| | FgPlus | – cross-correlation function for windows with positive correlations |
| | FgMinus | – cross-correlation function for windows with negative correlations |

**Example:**

```
# Track1 vs Track2
dist      Bkg       Fg        FgPlus     FgMinus
-500000  -0.04668   -0.91321   -1.13892    0.42649
-499800  -0.03251   -0.90076   -1.11362    0.36262
-499600  -0.06510   -0.87523   -1.07763    0.32609
                   ...
499400   -0.06324   -0.89405   -1.10051    0.33138
499600   -0.04718   -0.89841   -1.10964    0.35531
499800   -0.05677   -0.90264   -1.12193    0.39892
```

### 9.3.2 Full format (`outChrom=1`)

In this case the file **`<track1>`**∼**`<track2>.dist`** will have additional information by chromosomes.

**Example:**

```
# SMARCA5_100 vs SMARCB1_100
 dist       Bkg         Fg          FgPlus      FgMinus     chr1      chr2    ...
-500000    -0.02757    -1.60256    -1.69173    0.07157    -1.418    -1.217   ...
-499800    -0.01842    -1.57518    -1.66986    0.20262    -1.392    -1.208   ...
                    ...
 499600    -0.01402    -1.58403    -1.68101    0.23669    -1.389    -1.223   ...
 499800    -0.02548    -1.60363    -1.69075    0.03198    -1.422    -1.210   ...
```

## 9.4 Local correlation track

**`<track1>`**∼**`<track2>.bgraph`** – a bed graph file for kerneled local correlations. This track can be used for kerneled correlation visualization with Genome Browser. The track can also be used for further analysis, such as peak calling and searching the correlation of this track with other tracks. Note, The values can be positive or negative and can be outside the interval [-1,1].

This output regulates by **`outLC`** parameter. This parameter can take 3 possible values:

| | |
|---|---|
| 0 | – No local correlation produced |
| **`BASE`** | – The local correlation will be based on zero. No negative values will be produced |
| **`CENTER`** | – The local correlation will be based on mean. Negative values will be produced |

The program also used 2 additional parameters for filter the output:

| | |
|---|---|
| **`L_LC`** | – Threshold for local kernel correlation for low (usually negative) local correlations to be written into the local co |
| **`R_LC`** | – Threshold for local kernel correlation for high local correlations to be written into the local correlation file: *corr* |

Format: Standard graph format (see Genome Browser documentation). The file may contain a comment line defining segments without data. These segments appear if the source tracks do not contain any data here.

```
track type=bedGraph name="track1~track2" description="Local correlation"
#chr1 0 860800 ?
chr1 861100 861600 1.5
chr1 861600 861700 1.4
        ...
```

If the parameter **`writeDistr=DETAILS`** the file **`<track1>∼<track2>.LChist`** is also written. This file contains the statistical information of the local correlations. The file represents a histogram and contains the following columns

| | |
|---|---|
| **`corr`** | local correlation |
| **`obs`** | observed density |
| **`nObs`** | number of observations |
| **`exp`** | Expected density |
| **`nExp`** | number of observations in the background |
| **`r_CDF_obs`** | $1 - cdf(corr)$ for observed values |
| **`r_CDF_exp`** | $1 - cdf(corr)$ for the background |
| **`R_FDR`** | FDR for value less than corr (percent) |
| **`l_CDF_obs`** | $cdf(corr)$ for observed values |
| **`l_CDF_exp`** | $cdf(corr)$ for the background |
| **`L_FDR`** | FDR for value to be greater than corr (percent) |

```
corr   obs        nObs      exp        nExp    r_CDF_obs  r_CDF_exp  R_FDR  l_CDF_obs  l_CDF_exp  L_FDR
0.01   3.58E+001  14530762  1.48E+000  602387  1.00E+000  1.00E+000  100    1.00E+000  4.14E-002  4.15
0.04   1.06E-004  43        1.62E-001  65825   2.92E-004  9.59E-001  100    1.00E+000  4.60E-002  4.6
0.07   6.40E-005  26        1.54E-001  62385   2.89E-004  9.54E-001  100    1.00E+000  5.03E-002  5.03
0.1    4.68E-005  19        1.31E-001  53067   2.87E-004  9.50E-001  100    1.00E+000  5.39E-002  5.39
                  ...                          ...                         ...
1.64   7.39E-006  3         1.02E-003  415     2.68E-004  6.67E-005  24.85  1.00E+000  1.00E+000  100
1.66   4.92E-006  2         9.31E-004  378     2.68E-004  3.81E-005  14.22  1.00E+000  1.00E+000  100
1.69   9.85E-006  4         4.33E-004  176     2.68E-004  1.21E-005  4.52   1.00E+000  1.00E+000  100
1.72   4.92E-006  2         0.00E+000  0       2.68E-004  0.00E+000  0      1.00E+000  1.00E+000  100
1.75   7.39E-006  3         0.00E+000  0       2.68E-004  0.00E+000  0      1.00E+000  1.00E+000  100
                  ...                          ...                         ...
12.53  1.13E-004  46        0.00E+000  0       1.09E-005  0.00E+000  0      1.00E+000  1.00E+000  100
12.56  2.56E-004  104       0.00E+000  0       7.77E-006  0.00E+000  0      1.00E+000  1.00E+000  100
12.59  2.22E-005  9         0.00E+000  0       6.19E-007  0.00E+000  0      1.00E+000  1.00E+000  100
```

## 9.5 Autocorrelation `track.auto`

When the parameter **`AutoCorr=1`** the autocorrelation for tracks will be written:

**Example:**

```
-500000   -0.01006
-499800   -0.01045
      ...
   -200    0.20834
      0    1.00000
    200    0.20834
        ...
 499600   -0.00998
 499800   -0.01045
```

## 9.6 Spectrum density

When the parameter **`outSpectr=1`** the spectrum for tracks will be written to the file **`<track1>∼<track2>.spect`**:

**Example:**

```
# Track1 Track2
Wave_Length  Spectrum1  Spectrum2
1000000.00     6.277      2.053
 500000.00     6.234      2.042
 ...
```

### 9.7 Chromosome statistics `<track1>~<track2>.chrom`

The file contains:

| | |
|---|---|
| Chrom | Chromosome name |
| av1 | Average signal in track1 |
| av2 | Average signal in track2 |
| cc | Average correlation |
| count | Number of non-zero windows |

**Example:**

```
Track1
Track2
 chrom   av1    av2    cc  count
  chr1  758.94  611.39   0.58   232
  chr2  660.22  562.58   0.61   242
  chr3  640.48  558.50   0.60   197
```

## 10 Statistics.tsv – total statistics for all runs

This is a cumulative file with results. The file name can be changed using parameter **`statistics`**.

The file contains a table with columns:

| | |
|---|---|
| **`id`** | Run ID |
| **`Date`** | Date/Time of the end of the run in format: 13.05.17 15:13:59 |
| **`version`** | Program version |
| **`name1`** | Name of the first track |
| **`name2`** | Name of the second track |
| **`nFgr`** | Number of foreground observations |
| **`nBkg`** | Number of background observations |
| **`Fg.Corr`** | Average correlation for coherent windows |
| **`FgCorr_sd`** | Standard deviation of the correlation for coherent windows |
| **`Bg.Corr`** | Average correlation for shuffled windows |
| **`BgCorr_sd`** | Standard deviation of the correlation for shuffled windows |
| **`Mann-Z`** | Man-Whitney Z-score |
| **`p-value`** | Man-Whitney p-value |
| **`PDF_report`** | Filename of the pdf report |
| **`HTML_report`** | Filename of the html report |
| **`kernelSigma`** | Kernel Width |
| **`wSize`** | Window size |

Example:

| id | Date | version | name1 | name2 | nFgr | nBkg | Fg. Corr | FgCorr_sd | Bg. Corr |
|---|---|---|---|---|---|---|---|---|---|
| 6578d779 | 25.07.24 19:51:16 | 2.44 | SMARCA5_100 | SMARCB1_100 | 2907 | 2907 | 0.606 | 0.300 | 0.0122 |
| 6580808f | 25.07.24 19:59:38 | 2.44 | SMARCA5_100 | SMARCB1_100 | 2907 | 2907 | 0.606 | 0.300 | 0.0122 |

Continue of the table

| BgCorr_sd | Mann-Z | p-value | PDF_report | HTML_report | kernelSigma | wSize |
|---|---|---|---|---|---|---|
| 0.375 | 50.412 | 0.00e+00 | SMARCA5_100~SMARCB1_100.pdf |  | 30000 | 1000000 |
| 0.375 | 50.412 | 0.00e+00 |  |  | 30000 | 1000000 |

## 10.1 File Parameters.tsv

This file contains all the parameters (including the default values) for the *StereoGene* run. The correspondence of rows in this table with **Statistics.tsv** can be determined by the run id.

| id | version | bin | profPath | trackPath | resPath | report | chrom | bpType | NA |
|---|---|---|---|---|---|---|---|---|---|
| 6578d779 | 2.50 | 200 | ../Files/profiles/ | ../Files/Tracks/ | ../Files/res/ | report/ | chr_len_hg38.tab | SIGNAL | 0 |
| 6580808f | 2.50 | 200 | ../Files/profiles/ | ../Files/Tracks/ | ../Files/res/ | report/ | chr_len_hg38.tab | SIGNAL | 0 |

Continue of the table

| threshold | kernelType | customKern | kernelSigma | kernelShift | wSize | wStep | flankSize | maxNA | maxZero |
|---|---|---|---|---|---|---|---|---|---|
| 0 | NORMAL | NONE | 30000 | 0 | 1000000 | 1000000 | 0 | 1e+02 | 1e+02 |
| 0 | NORMAL | NONE | 30000 | 0 | 1000000 | 1000000 | 0 | 1e+02 | 1e+02 |

Continue of the table

| nShuffle | noiseLevel | complFg | localShuffle | outSpectr | outChrom | writeDistr | plotType | Cross | outLC | L_LC |
|---|---|---|---|---|---|---|---|---|---|---|
| 2000 | 0 | IGNORE_ | 1 | 0 | 1 | DETAIL | PDF | 1 | NA | -20 |
| 2000 | 0 | IGNORE_ | 1 | 0 | 1 | DETAIL | PDF | 1 | NA | -20 |

Continue of the table

| R_LC | AutoCorr | aliases | Rscript | plotH | plotW |
|---|---|---|---|---|---|
| 1 | 0 | aliases | C:/Program Files/R/R-3.6.0/bin/x64/Rscript.exe | 3 | 7 |
| 1 | 0 | aliases | C:/Program Files/R/R-3.6.0/bin/x64/Rscript.exe | 3 | 7 |