ChromHMM Version 1.20 User Manual

Jason Ernst and Manolis Kellis

Overview

ChromHMM is a Java program for the learning and characterizing chromatin states using a multivariate Hidden Markov Model that models the combinatorial and spatial patterns in data from multiple chromatin marks. ChromHMM can be run on any computer supporting Java 1.5 or later. ChromHMM is executed from the command line with a command such as:

`java -mx4000M -jar ChromHMM.jar` *`Command`* `[commandoptions] commandparameters`

Where the 4000 specifies the amount of memory given to Java and could be adjusted based on the size of the data, the model size, and the *Command* being executed. In some cases the memory flag could be omitted.

ChromHMM has eleven top level commands which then determine the required and optional set of parameters. The top level commands are briefly described here and a detailed description of each command, the required and optional parameters can be found in the remaining sections. The usage for any of these commands can be obtained at the command line by typing at the command line: `java -jar ChromHMM.jar` *`Command`*

LearnModel – takes a set of binarized data files, learns chromatin state models, and by default produces a segmentation, generates browser output with default settings, and calls OverlapEnrichment and NeighborhoodEnrichments with default settings for the specified genome assembly. A webpage is a created with links to all the files and images created.

BinarizeBed – converts a set of bed files of aligned reads into binarized data files for model learning and optionally prints the intermediate signal files

BinarizeBam – converts a set of bam files of aligned reads into binarized data files for model learning and optionally prints the intermediate signal files

BinarizeSignal – converts a set of signal files into binarized files.

MakeSegmentation – takes a learned model and binarized data and outputs a segmentation.

MakeBrowserFiles – can convert segmentation files into a browser viewable format.

OverlapEnrichment – shows the enrichment of each state of a segmentation for a set of external data

NeighborhoodEnrichment – shows the enrichment of each state relative to a set of anchor positions

CompareModels – can compare models with different numbers of states in terms of correlation in emission parameters

Reorder – allows reordering the states of the model, the columns of the emission matrix, or adding state labels

EvalSubset – can be used to evaluate the extent to which a subset of marks can recover a segmentation using the full set of marks.

ConvertGeneTable – converts gene annotations from a UCSC genome browser gene table into coordinate and anchor files for enrichment analyses

MergeBinary – merges the columns of binarized files for different marks, also can be used to split rows of a binarized file

StatePruning – can be used to prune states from a model in order to initialize models when using the non-default two pass approach

LearnModel

Description

This command takes a directory with a set of binarized data files and learns a chromatin state model. Binarized data files have '_binary' in the file name. The format for the binarized data files are that the first line contains the name of the cell separated by a tab with the name of the chromosome. The second line contains in tab delimited form the name of each mark. The remaining lines correspond to consecutive bins on the chromosome. The remaining lines in tab delimited form corresponding to each mark, with a '1' for a present call or '0' for an absent call and a '2' if the data is considered missing at that interval for the mark. As an example:

```
Cell  chr1
Mark1 Mark2 Mark3
0     0     0
0     1     0
0     0     1
```

The emission parameters in table format are printed to text, png, and svg image files named beginning with 'emissions_' then having the number of states and optionally an identifier followed by a '.txt', '.png' , and '.svg' respectively. Similarly the transition parameters are printed to text and image files prefixed with 'transitions'. All the model parameters in an internal format are printed to a file beginning with the name 'model_'. After each iteration across the genome these files are updated. At the termination of the learning process, LearnModel effectively calls MakeSegmentation to produce segmentation files, MakeBrowserFiles, OverlapEnrichments, and NeighborhoodEnrichments with all the files corresponding to the specified assembly. Also upon termination webpage is created called webpage_NUMSTATES.html (or webpage_NUMSTATES_outfileID.html if a outfileID is specified) summarizing all the files and images created, and if a browser can be found, then is automatically opened with this file. The current search progress in terms of estimated log likelihood and change after each iteration is printed to the terminal. If one terminates the search early, a segmentation can still be produced by applying MakeSegmentation to the current model file.

Usage

```
LearnModel [-b binsize][-color r,g,b][-d convergedelta]
[-e loadsmoothemission][-f inputfilelist][-gzip][-h informationsmooth]
[-holdcolumnorder][-holdroworder][-i outfileID][-init information|random|load]
[-l chromosomelengthfile][-lowmem][-m modelinitialfile][-many][-n numseq][-
noautoopen][-nobed][-nobrowser][-noenrich][-noimage][-p maxprocessors][-pseudo][-
printposterior][-printstatebyline][-r maxiterations][-s seed][-scalebeta][-
splitrows][-stateordering emission|transition][-t loadsmoothtransition] [-u
coorddir][-v anchorfiledir][-x maxseconds][-z zerotransitionpower]
inputdir outputdir numstates assembly

Note items in [] are optional
```

Required Parameters

`inputdir` – This is the directory containing the binarized input files. Only file names containing '_binary' are used by default.

`outputdir` – This is the directory where the output files are written.

numstates – This parameter specifies the number of states to include in the model.

assembly – This parameter specifies the assembly. OverlapEnrichment and NeighborhoodEnrichment will be called with default parameters using this assembly.

Optional Parameters

-b binsize – The number of base pairs in a bin determining the resolution of the model learning and segmentation. By default this parameter value is set to 200 base pairs.

-color r,g,b – This specifies the color of the heat map. r,g,b are integer values between 0 and 255 separated by commas. By default this parameter value is 0,0,255 corresponding to blue.

-d convergedelta – The threshold on the change on the estimated log likelihood that if it falls below this value, then parameter training will terminate. If this value is less than 0 then it is not used as part of the stopping criteria. The default value for this parameter is 0.001.

-e loadsmoothemission – This parameter is only applicable if the load option is selected for the init parameter. This parameter controls the smoothing away from 0 when loading a model. The emission value used in the model initialization is a weighted average of the value in the file and a uniform probability over the two possible emissions. The value in the file gets weight (1-loadsmoothemission) while uniform gets weight loadsmoothemission. The default value of this parameter is 0.02.

-f inputfilelist – A list of files to include in the segmentation. If this option is not provided then by default all files in inputdir with '_binary' in the name will be included in the model learning.

-gzip – If this flag is present then the segmentation files and browser files are outputted in a gzip format

-h informationsmooth – A smoothing constant away from 0 for all parameters in the information based initialization. This option is ignored if random or load are selected for the initialization method. The default value of this parameter is 0.02.

-holdcolumnorder –Including this flag suppresses the reordering of the mark columns in the emission parameter table display.

-holdroworder –Including this flag suppresses reordering the states

-i outfileID – If this option is included the string outfileID is included in the file names of all the output files.

-init information|random|load – This specifies the method for parameter initialization method. information is the default method described in (Ernst and Kellis, Nature Methods 2012). Random –randomly initializes the parameters from a uniform distribution. load loads the parameters specified in modelinitialfile and smooths them based on the value of the loadsmoothemission and loadsmoothtransition parameters. The default is information.

-l chromosomelengthfile – This file specifies the length of the chromosomes. It is a two column tab delimited file with the first column specifying the chromosome name and the second column the length. If this file is provided then no end coordinate will exceed what is specified in this file. By default BinarizeBed excludes the last partial bin along the chromosome, but if that is included in the binarized data input files then this file should be included to give a valid end coordinate for the last interval. Also by default if a file exists in the CHROMSIZES directory or the directory for the specified assembly that will be used (note that this is only applicable when using this option through LearnModel and not MakeSegmentation since no assembly is specified).

-lowmem – This flag reduces the memory usage at the cost of additional run time when there is sufficient memory without it.

-m modelinitialfile – This specifies the model file containing the initial parameters which can then be used with the load option

-many – If this flag is present, ChromHMM uses a slower, but more numerically stable procedure for evaluating the emission parameters. There is generally only reason to use this if there are many features (at least several hundred).

-noautoopen – If this flag is present, then ChromHMM does not try to automatically open a web browser with the summary page of results.

-nobed – If this flag is present, then this suppresses the printing of segmentation information in the four column format. The default is to generate a four column segmentation file

-nobrowser – If this flag is present, then genome browser files are not printed. If -nobed is requested then browserfile writing is also suppressed.

-noenrich – If this flag is present, then enrichment files are not printed. If -nobed is requested then enrichment file writing is also suppressed.

-noimage – If this flag is present, then outputting of image files is suppressed.

-n numseq – If this flag is present and the '-p' flag is present then on each iteration of training only numseq chromosome files are randomly selected to be used for training. In such cases the '-d' flag should be set to a negative number so model learning does not terminate prematurely since negative changes in the log-likelihood are expected since different chromosomes are used on each iteration. Also only numseq files are considered in the initial model initialization under the default 'information' mode. If the '-n' flag is specified without the '-p' flag a subset of chromosomes will still be used for initialization, but all chromosomes will still be used on all iterations of training.

-p maxprocessors – If this option is present then ChromHMM will attempt to train a model using multiple processors in parallel. ChromHMM will create multiple threads up to the number of processors specified by maxprocessors. If maxprocessors is set to 0, then the number of threads will just be limited by the number of processors available. Note that each additional processor used will require additional memory. If this option is specified ChromHMM will use a standard Baum-Welch training algorithm opposed to the default incremental expectation-maximization algorithm.

-pseudo – If this flag is present, pseudo counts of 1 are used in computing the model parameters to smooth away from zero values. These pseudo counts can provide numerical stability in the situation when the -n numseq is specified in training and some feature has very few present occurrences.

-printposterior – If this flag is present the posterior probabilities over state assignments are also printed in a file. These files end with '_posterior.txt' or '_posterior.txt.gz' if the '-gzip' flag is specified. One file is generated per cell type and chromosome. The first line of these files specify the chromosome and cell type, followed by a header line for each column, and then the posterior probabilities one per line. By default these files are not printed.

-printstatebyline – If this flag is present the state assignment are printed to a file one per line. These files end with '_maxstate.txt' or '_maxstate.txt.gz' if the '-gzip' flag is specified. One file is generated per cell type and chromosome. The first line specifies the cell type and chromosome and the second line says MaxState and the state ordering methods. The remaining lines have the state assignments. By default these files are not printed.

`-r maxiterations` – This option specifies the maximum number of iterations over all the input data in the training. By default this is set to 200.

`-s seed` – This allows the specification of the random seed. Randomization is used to determine the visit order of chromosomes in the incremental expectation-maximization algorithm used to train the parameters and also used to generate the initial values of the parameters if `random` is specified for the `init` method.

`-scalebeta` – This flag changes the internal numerical procedure for computing the backward variables that can prevent numerical overflow observed in specialized settings. Specifically the backward variables are rescaled based on the sum of their values at a position opposed to reusing the scaling of the forward variables. Numerical underflow is prevented by setting the beta values to 10^{-300} if they fall below that, and likewise for the forward variables with the added condition that emission must evaluate greater than 0 at the position.

`-splitrows` – This flag is used to denote that the chromosome data is split across multiple files as produced by applying BinarizeBam, BinarizeBed, BinarizeSignal, or MergeBinary with this option and that they should be combined with the proper chromosome naming and coordinates.

`-stateordering emission|transition` – This determines whether the states are ordered based on the emission or transition parameters. See (Ernst and Kellis, Nature Methods 2012) for details. Default is `emission`.

`-t loadsmoothtransition` – This parameter is only applicable if the `load` option is selected for the `init` parameter. This parameter controls the smoothing away from 0 when loading a model. The transition value used in the model initialization is a weighted average of the value in the file and a uniform probability over the transitions. The value in the file gets weight (`1-loadsmoothtransition`) while uniform gets weight `loadsmoothtransition`. The default value is 0.5.

`-u coorddir` – This parameter specifies the directory containing subdirectories where each subdirectory contains coordinate files for a specific assembly for OverlapEnrichment. By default this is the `COORDS` directory where the `ChromHMM.jar` file is.

`-v anchorfiledir` – This parameter specifies the directory containing subdirectories where each subdirectory contains anchor files for a specific assembly for NeighborhoodEnrichment. By default this is the `ANCHORFILES` directory where the `ChromHMM.jar` file is.

`-x maxseconds` – This parameter specifies the maximum number of seconds that can be spent optimizing the model parameters. If it is less than 0, then there is no limit and termination is based on maximum number of iterations or a log likelihood change criteria. The default value of this parameter is -1.

`-z zerotransitionpower` – This parameter determines the threshold at which to set extremely low transition probabilities to 0 during training. Setting extremely low transition probabilities makes model learning more efficient with essentially no impact on the final results. If a transition probability falls below 10^-`zerotransitionpower` during training it is set to 0. Making this parameter too low and thus the cutoff too high can potentially cause some numerical instability. By default this parameter is set to 8.

BinarizeBed

Description

This command converts coordinates of aligned reads into binarized data form from which a chromatin state model can be learned. The binarization is based on a poisson background model. If no control data is specified the parameter to the poisson distribution is the global average number of reads per bin. If control data is specified the global average number of reads is multiplied by the local enrichment for control reads as determined by the specified parameters. Optionally intermediate signal files can also be outputted and these signal files can later be directly converted into binary form using the BinarizeSignal command.

Usage

```
BinarizeBed [-b binsize][-c controldir][-center]
[-colfields chromosome,start,end[,strand]][-e offsetend][-f foldthresh]
[-g signalthresh][-gzip][-n shift][-o outputcontroldir][-p poissonthresh][-peaks
[-i splitindex]][-s offsetstart][-splitrows [-j numsplitbins]][-strictthresh][-t
outputsignaldir][-u pseudocountcontrol][-w flankwidthcontrol] chromosomelengthfile
inputbeddir cellmarkfiletable outputbinarydir
```

Note items in [] are optional

Required Parameters

`chromosomelengthfile` – A two column tab delimited file with the first column being the chromosome and the second being the chromosome length. For genome assemblies hg18, hg19, hg38, mm9, mm10, rn5, rn6, danRer7, danRer10, dm3, dm6, ce6, and ce10 these can be found in the directory CHROMSIZES. For other assemblies these can be obtained with the fetchChromSizes script available from the UCSC browser http://hgdownload.cse.ucsc.edu/admin/exe/ specifying the desired assembly and redirecting the output to a text file.

`inputbeddir` – The directory containing the input bed files.

`cellmarkfiletable` – A tab delimited file each row contains the cell type or other identifier for a groups of marks, then the associated mark, then the name of a bed file, and optionally a corresponding control bed file

```
cell1 mark1 cell1_mark1.bed   cell1_control.bed
cell1 mark2 cell1_mark2.bed   cell1_control.bed
cell2 mark1 cell2_mark1.bed   cell2_control.bed
cell2 mark2 cell2_mark2.bed   cell2_control.bed
```

If a mark is missing in one cell type, but not others it will receive a 2 for all entries in the binarization file and -1 in the signal file. If the same cell and mark combination appears on multiple lines, then the union of all the reads across entries is taken except for control data where each unique file is only counted once.

`outputbinarydir` – The output directory to which the binarized data files should be written. These files will be named `CELL_CHROM_binary.txt` or `CELL_CHROM_binary.txt.gz` if the `'-gzip'` flag is specified.

Optional Parameters

`-b binsize` – The number of base pairs in a bin determining the resolution of the model learning and segmentation. By default this parameter value is set to 200 base pairs.

-c controldir – A directory containing the control input files. If this is not specified then the inputbeddir is used. If no control files are specified then by default a uniform background will be used in determining the binarization thresholds.

-center – If this flag is present then the center of the interval is used to determine the bin to assign a read. This can make sense to use if the coordinates are based on already extended reads. If this option is selected, then the strand information of a read and the shift parameter are ignored. By default reads are assigned to a bin based on the position of its 5' end as determined from the strand of the read after shifting an amount determined by the -n shift option.

-colfields chromosome,start,end[,strand] – This is a comma delimited list of integers specifying the 0-based index of the columns of the chromosome, the start position, the end position, and the strand ('+' or '-') of the reads. If the option –center is present, then the strand can be omitted. By default the chromosome is found in the first column, the start in the second, and the end position in the third, and the strand in the sixth column unless there are fewer columns in which case the last column of the file is assumed to contain the strand.

-e offsetend – Specifies the amount that should be subtracted from the end coordinate of a read so that both coordinates are inclusive and 0 based. The default value is 1 corresponding to standard bed convention of the end interval being 0-based but not inclusive.

-f foldthresh – This indicates a threshold for the fold enrichment over expected that must be met or exceeded by the observed count in a bin for a present call. The expectation is determined in the same way as the mean parameter for the poission distribution in terms of being based on a uniform background unless control data is specified. This parameter can be useful when dealing with very deeply and/or unevenly sequenced data. By default this parameter value is 0 meaning effectively it is not used.

-g signalthresh – This indicates a threshold for the signal that must be met or exceeded by the observed count in a bin for a present call. This parameter can be useful when desiring to directly place a threshold on the signal. By default this parameter value is 0 meaning effectively it is not used.

-gzip – If this flag is present then the binarized files are outputted in a gzip format

-i splitindex – If the -peaks option is included then this option can also be included which generates row split files for each chromosome only for positions corresponding to rows in the interval [splitindex*numsplitbins, (splitindex+1)*numsplitbins]. This option enables greater parallel processing and reduced memory usage when binarizing based on peak files to generate. If using this option BinarizeBed should be run once for each value of splitindex between 0 and ceil(max(chromosome size)/(binsize*numsplitbins))-1. When using this option output is provided for all chromosomes in chromosomelengthfile opposed to just those that contain peaks.

-j numsplitbins – If the -splitrows option is specified, then this parameter can be adjusted to change the maximum number of rows that will be placed in any split file. The default value of this parameter is 5000, which corresponds to covering a 1MB under the default setting of binsize.

-n shift – The number of bases a read should be shifted to determine a bin assignment. Bin assignment is based on the 5' end of a read shifted this amount with respect to the strand orientation. By default this value is 100.

-o outputcontroldir – This specifies the directory to which control data should be printed. The files will be named CELL_CHROM_controlsignal.txt or CELL_CHROM_controlsignal.txt.gz if the -gzip flag is specified. Control data will only be outputted if there are control bed files present and an output control directory is specified.

-p poissonthreshold – This option specifies the tail probability of the poisson distribution that the binarization threshold should correspond to. The default value of this parameter is 0.0001.

`-peaks` – This option specifies to treat the bed files as peak calls directly and give a '1' call to any bin overlapping a peak call. Not recommend for broad peak calls.

`-s offsetstart` – The amount that should be subtracted from the interval start coordinate so the interval is inclusive and 0 based. Default is 0 corresponding to the standard bed convention.

`-splitrows` – If the flag is present then the chromosome files are split across multiple files, with the maximum number of bins in any file given by `numsplitbins`. Splitting chromosome files can be desired as an approximation in large scale applications where because of time or memory constraints ChromHMM cannot handle the size of the data without splitting, and in particular when using `-n numseq` option of `LearnModel`.

`-strictthresh` – If this flag is present then the poisson threshold must be strictly greater than the tail probability, otherwise by default the largest integer count for which the tail includes the poisson threshold probability is used.

`-t outputsignaldir` – If specified signal files will be generated and outputted to the given directory. The files will be named CELL_CHROM_signal.txt. These files could later be binarized directly at different thresholds with the BinarizeSignal command. By default no output signal is written.

`-u pseudocountcontrol` – An integer pseudocount that is uniformly added to every bin in the control data in order to smooth the control data from 0. The default value is 1.

`-w flankwidthcontrol` – This determines the extent of the spatial smoothing in computing the local enrichment for control reads. The local enrichment for control signal in the x^{th} bin on the chromosome after adding `pseudocountcontrol` is computed based on the average control counts for all bins within *x-w* and *x+w*. If no `controldir` is specified, then this option is ignored. The default value is 5.

BinarizeBam

Description

This command is the same as the BinarizeBed command described above except the reads are stored in bam files instead of bed files.

Usage

```
BinarizeBam [-b binsize][-c controldir][-center][-e offsetend][-f foldthresh]
[-g signalthresh][-gzip][-n shift][-o outputcontroldir][-p poissonthresh][-
paired|[-center][-n shift][-peaks [-i splitindex]]][-s offsetstart][-splitrows [-j
numsplitbins]][-strictthresh][-t outputsignaldir][-u pseudocountcontrol][-w
flankwidthcontrol] chromosomelengthfile inputbamdir cellmarkfiletable
outputbinarydir
```

Note items in [] are optional

The parameters are used the same way as described in BinarizeBed above except there is an additional option:

`-paired` – If this option is present then reads in the BAM file are treated as pairs, and each pair is counted once with bin assignment is based on shifting half the insert size. If this option is present then the `-n shift`, `-center`, and `-peaks` options cannot be used.

BinarizeSignal

Description

This command converts data already processed into signal files into binarized data files. Signal data files have '_signal' in the file name. The format for the signal file is the first line contains the name of the cell separated by a tab with the name of the chromosome. The second line contains in tab delimited form the name of each mark. The remaining lines correspond to consecutive bins on the chromosome and contain the integer signal value for each mark. As an example:

```
Cell  chr1
Mark1 Mark2 Mark3
0     4     0
1     3     0
2     1     9
```

Control signal data are in files with '_controlsignal' in the file name and have either one column which is used for all the marks or control data specified for every mark matched to a '_signal' data file. Note the binarization from signal is designed only for signal data which represent counts of reads assigned to bins such as the optional output from the BinarizeBed command. If the signal was computed in other ways, then the binarization based on the poisson distribution may not give meaningful results. '-1' entries in a signal files are considered missing.

Usage

```
BinarizeSignal [-c controldir][-f foldthresh][-g signalthresh][-gzip][-p
poissonthresh][-splitrows [-j numsplitbins]][-strictthresh][-u
pseudocountcontrol][-w flankdwidth] signaldir outputdir
```

Note items in [] are optional

Required Parameters

`signaldir` – Directory containing the signal input files. Only files with '_signal' in the name will be read

`outputdir` – Directory to which the binarized data files should be written

Optional Parameters

`-c controldir` – If a controldir is specified then this directory has control signal files that will be used. Files containing '_controlsignal' will be considered control files and there must be matched control signal file for each signal file.

`-f foldthresh, -g signalthresh, -gzip, -j numsplitbins, -p poissonthreshold, -splitrows, -strictthresh, -u pseudocountcontrol, -w flankwidthcontrol` are same as described above for BinarizeBed

MakeSegmentation

Description

This command takes a saved model file and binarized data and outputs files with the segmentation state assignments and/or the posterior state probabilities. These files can also be outputted by LearnModel, but this command enables producing the segmentations without the need to relearn the model. The columns of the binarized data should match the column ordering in the model file, which by default is the order of columns in the binarized data used to learn the model.

Usage

```
MakeSegmentation [-b binsize][-f inputfilelist][-gzip][-i outfileID]
[-l chromosomelengthfile][-lowmem][-many][-nobed][-printposterior][-
printstatesbyline][-scalebeta][-splitrows] modelfile inputdir outputdir
```

Note items in [] are optional

Required Parameters

`modelfile` – the file with the model parameters to produce a segmentation

`inputdir` – the directory with the binarized data for which a segmentation should be produced

`outputdir` – the output directory for the segmentation results

Optional Parameters

`-b binsize, -f inputfilelist , -gzip, -i outfileID , -l chromosomelengthfile, -many, -nobed, -printposterior, -printstatesbyline, -scalebeta, -splitrows` are the same as described above in LearnModel

MakeBrowserFiles

Description

This command converts segmentation coordinates in a .bed file into two files that can be viewed as custom tracks in the UCSC genome browser (http://genome.ucsc.edu). One of the files can give a dense view of the segmentation in a single track with states being differentiated solely by color. The other file gives an expanded view showing one state per track.

Usage

```
MakeBrowserFiles [-c colormappingfile][-gzip][-m labelmappingfile][-n numstates]
segmentfile segmentationname outputfileprefix
```

Note items in [] are optional

Required Parameters

`segmentfile` – The name of a four column tab delimited segmentation file. The four columns are chromosome, start, end, and label

`segmentationname` – A name for the segmentation that will be given to the segmentation and displayed in the browser

`outputfileprefix` – The output file prefix including possibly the directory name. Appended to this prefix is the suffix '_browserexpanded.bed' and '_browserdense.bed' for the expanded and dense browser files.

Optional Parameters

`-c colormappingfile` – The specified file under this option gives a mapping from state ID to desired color. The file is a two column tab delimited file. The first column contains the state number without the state ordering prefix. The second column contains r,g,b integer values between 0 and 255 delimited by commas specifying the colors for each state. If this file is not specified different colors are automatically generated for each state such that states closer in the order have more similar colors.

`-gzip` – If this flag is present then the browser files are outputted in a gzip format.

`-m labelmappingfile` – This option can specify a file which maps state IDs to descriptive names. The descriptive names can be appended to the state IDs when viewing the states in the browser. The format of this file is a two column tab delimited file. The first column contains each state ID with the state ordering letter prefix. The second column contains a descriptive name or mnemonic. If this file is not specified, then just the state IDs are displayed in the browser.

`-n numstates` – This option specifies the number of states on which the segmentation is based. By default the maximum state number is used to indicate how many states there are. This parameter can potentially influence the default state coloring if there are locations in the genome which were not assigned to any state.

OverlapEnrichment

Description

This command can be used to compute the fold enrichment of each state of the segmentation for a set of external coordinates by default in bed format. Signal values can optionally be associated with each coordinate to weight the enrichments. The enrichment is outputted as a table in both text and image format. The directory COORDS contains the subdirectories hg18, hg19, hg38, mm9, mm10, rn5, rn6, danRer7, danRer10, dm3, dm6, ce6, and ce10. These directories include bed files for the RefSeq transcription start site, transcript end site, gene, exon, and regions within 2kb of the transcription start site of the corresponding assembly. Also CpG islands are included for hg18, hg19, hg38, mm9, mm10, rn5, and rn6. For hg18 and hg19 a set of coordinates on NuclearLamin domains from (Guelen et, Nature 2008) are also provided. Coordinates were obtained from the UCSC genome browser. A user can also provide their own coordinates.

By default the fold enrichment calculation is as follows, let:
A - be the number of bases in the state
B - be the number of bases in the external annotation
C - be the number of bases in the state and the external annotation
D - be the number of bases in the genome
The fold enrichment is then defined as (C/A)/(B/D).

Usage

```
OverlapEnrichment [-a cell][-b binsize][-binres][-color r,g,b][-center]
[-colfields chromosome,start,end[,signal]][-e offsetend][-f coordlistfile][-
labels][-lowmem][-m labelmappingfile][-multicount][-noimage][-posterior][-s
offsetstart][-signal][-t title][-uniformscale] inputsegment inputcoorddir
outfileprefix
```

Note items in [] are optional

Required Parameters

`inputsegment` – either a segmentation bed file or if the `-posterior` option is selected the directory containing the posterior files

`inputcoorddir` – a directory containing the external coordinates for enrichment analysis. If inputcoorddir is a file instead of a directory then enrichments for just the file are computed.

`outfileprefix` – the prefix of the text and image files to which the fold enrichments should be written. The enrichment text file has a '.txt' extension added and a heat map image has a '.png' and '.svg' extension added.

Optional Parameters

`-a cell` – If the `-posterior` flag is specified then this option can be used to specify the cell type for which to compute the enrichment. By default the posterior enrichment is computed in aggregate over all cell types.

`-b binsize` – The number of base pairs in a bin determining the resolution of the model learning and segmentation. By default this parameter value is set to 200 base pairs.

`-binres` – If this flag is present, then this flag indicates enrichments should be computed at the bin resolution just requiring single base overlap of a coordinate for a bin to fully count the bin opposed to the default of base resolution.

-center – Use the center base for computing enrichments instead of the entire interval

-colfields chromosome,start,end[,signal] – This is a comma delimited list of integers which specify the 0-based index of the column fields contains the chromosome, start, end, and if signal is being used the the signal data. By default the values of these parameters are 0,1,2,3.

-color r,g,b – This specifies the color of heat map. r,g,b are integer values between 0 and 255 separated by commas. By default this parameter value is 0,0,255 corresponding to blue.

-e offsetend – Specifies the amount that should be subtracted from the end coordinate so the coordinate is inclusive and 0 based. The default value is 1 corresponding to standard bed convention of the end interval being 0-based and exclusive.

-f coordlistfile – This option specifies a file which lists one per line the coordinate files from the inputcoorddir to compute enrichment for and the order in which they should be displayed in the enrichment table. By default enrichments are computed for all files in inputcoorddir.

-labels – If this flag is present and the input is a segmentation bed file, then the four column should be treated as having state labels instead of state IDs or state numbers. If the fourth column has a state ID or state number before a '_' and then followed by a label, the states will state be ordered by the state ID or number, otherwise the state ordering in the output may differ from the original state ordering.

-lowmem – This flag reduces the memory usage at the cost of additional run time when there is sufficient memory without it.

-m labelmappingfile – This option can specify a file which maps state IDs to descriptive names. The descriptive names can be appended to the state IDs when viewing the overlap enrichments. The format of this file is a two column tab delimited file. The first column contains each state ID with the state ordering letter prefix. The second column contains a descriptive name or mnemonic. If this file is not specified, then just the state IDs are displayed.

-multicount – This flag indicates to count overlaps multiple times when the -signal flag is not present. If the -signal flag is present overlaps are always counted multiple times. By default without the -signal overlaps are only counted once. Overlaps are defined to either be at the base resolution or the bin resolution based on the -binres flag. If the input coordinate data for enrichments is known to be non-overlapping then including this flag can speed up the enrichment calculation without effecting the final results.

-noimage – If this flag is present, then outputting of image files is suppressed.

-posterior – Indicates that the full posterior should be used for the enrichment opposed to the default of the maximum probability state assignments. If this flag is present then inputsegment should be a directory with the posterior files.

-s offsetstart – This parameter specifies the value that should be subtracted from the interval start coordinate so the interval is inclusive and 0 based. Default is 0 corresponding to the standard .bed convention

-signal – If this flag is given, then signal information will be used in the enrichments if available otherwise all overlaps are given equal weight. If signal is used coordinates without signal given will be assumed to have 0 signal in the enrichment calculation

-t title – This option specifies a title for the heat map image. A default title of "Fold Enrichments" is used.

`-uniformscale` – If this flag is given, then in coloring the heat map all columns will be on the same color scale. The default is to have a column specific color scale which subtracts the minimum value in the column and then divides by the maximum column value.

NeighborhoodEnrichment

Description

This command given a set of anchor positions determines the fold enrichment for each state at fixed positions relative to the anchor positions. Signal values can optionally be associated with each coordinate to weight the enrichments. Strand information can also be optionally used to compute the positional enrichments in a strand aware manner. The enrichments are outputted both as a text file and in image format. The directory ANCHORFILES has subdirectories hg18, hg19, hg38 mm9, mm10, rn5, rn6, danRer7, danRer10, dm3, dm6, ce6, and ce10 that includes containing anchor position files for transcription start sites (TSS) and transcript end sites (TES) for the corresponding assemblies. Fold enrichments by default are calculated analogously as specified in OverlapEnrichment.

Usage

```
usage NeighborhoodEnrichment [-a cell][-b binsize][-color r,g,b]
[-colfields chromosome,position[,optionalcol1|,optionalcol1,optionalcol2]]
[-l numleftintervals][-labels][-lowmem][-m labelmappingfile ][-noimage][-
nostrand][-o anchoroffset][-posterior][-r numrightintervals][-s spacing][-
signal][-t title]
inputsegment anchorpositions outfileprefix
```

Note items in [] are optional

Required Parameters

`inputsegment` – Either a segmentation bed file or if the `-posterior` option is selected the directory containing the posterior file

`anchorpositions` – Specifies a file containing the coordinates of the anchor positions around which state enrichments will be determined. Positions are specified by a chromosome and coordinate.

`outfileprefix` – The prefix of the text and image files to which the fold enrichments should be written. The enrichment text file has a '.txt' extension added and a heat map images have a '.png' or '.svg' extensions added.

Optional Parameters

`-colfields chromosome,position[,optionalcol1|,optionalcol1,optionalcol2]` – This is a comma delimited list of integers which specify the 0-based index of the column fields contains the chromosome, position. If the `-nostrand` option is not present `optionalcol1` is the strand and if `-signal` is also specified `optionalcol2` is the signal. If `-nostrand is present and -signal` is selected `optionalcol1` has the signal column. By default the values of these parameters are 0,1,2,3.

`-l numleftintervals` – The number of enrichment columns to the left of the anchor position to display. By default this value is 10.

`-labels` – If this flag is present and the input is a segmentation bed file, then the four column should be treated as having state labels instead of state IDs or state numbers. If the fourth column has a state ID or state number before a '_' and then followed by a label, the states will state be ordered by the state ID or number, otherwise the state ordering in the output may differ from the original state ordering.

`-lowmem` – This flag reduces the memory usage at the cost of additional run time when there is sufficient memory without it.

`-m labelmappingfile` – This option can specify a file which maps state IDs to descriptive names. The descriptive names can be appended to the state IDs in the neighborhood enrichment output files. The format of this file is a two column tab delimited file. The first column contains each state ID with the state ordering letter prefix. The second column contains a descriptive name or mnemonic. If this file is not specified, then just the state IDs are displayed.

`-noimage` – If this flag is present, then outputting of image files is suppressed.

`-o anchoroffset` – The value that should be subtracted from the anchor coordinate so it is 0 based. By the default this value is 0.

`-r numrightintervals` – The number of enrichment columns to the right of the anchor position to display. By default this value is 10.

`-s spacing` – The spacing in base pairs at which column enrichments should be displayed. Default is `binsize`.

`-nostrand` – If this flag is present then no strand information is used in the enrichment calculations

`-a cell, -b binsize, -color r,g,b, -posterior, -signal, -t title` are the same as described for OverlapEnrichment

CompareModels

Description

This command allows the comparison of the emission parameters of a selected model to a set of models in terms of correlation. The output is a table in both heat map and text file form showing the maximum correlation of each state in the selected model with its best matching state in each other model.

Usage

```
CompareModels [-color r,g,b][-noimage] referencemodel comparedir outputprefix
```

Note items in [] are optional

Required Parameters

`referencemodel` – the file, including the path, to the reference set of emission parameters. This file must start with 'emissions_' and end with '.txt' or '.txt.gz'

`comparedir` – the directory containing all the emission parameters to be compared to. Only files that start with 'emissions_' and end with '.txt' or '.txt.gz' are compared

`outputprefix` – The prefix for the output files containing parameter correlation files in both .txt, .png, and .svg form. The columns are ordered by the number of states in the models and each row corresponds to a state in the reference model. The values correspond to the best correlation in emission parameters of any state in the comparison model for that state in the reference model.

Optional Parameters

`-color r,g,b` – This determines specifies the color of heatmap. r,g,b are integer values between 0 and 255 separated by commas. By default this parameter value is 0,0,255 corresponding to blue.

`-noimage` – If this flag is present, then outputting of image files is suppressed.

Reorder

Description

This command allows reordering the states of a model without relearning the model, and outputs a model file and emission and transition tables with the states reordered. The states can be reordered based on the emission or transition parameters or based on a user provided state ordering. Also the columns of the emission parameters can be ordered based on their original order in the data file, based on the correlation of the columns, or a user specified order. If a user provided state ordering is given a user has the option to also specify a bed file that should be reorder based on the provided ordering. Note that this option only reorders a single bed file at a time. If multiple bed files are desired to be reordered, this command can be run multiple times or alternatively MakeSegmentation and/or MakeBrowserFiles can be re-run with the updated model and/or segmentation files. Also note that this command does not reorder the states in any existing analysis file. Reordered analysis files can be obtained by rerunning OverlapEnrichment, NeighborhoodEnrichment, and/or CompareModels on reordered segmentation file(s). In the segmentation files the state number is prefixed by an 'E' if the ordering is based on emissions, 'T' if it was based on the transition parameters, and 'U' if it was user provided. The command can also be used to add descriptive state labels to the parameter heatmaps with the current ordering.

Usage

```
usage: Reorder [-color r,g,b][-f columnorderingfile][-holdcolumnorder]
[-i outfileID][-m labelmappingfile][-noimage][-o stateorderingfile [-r bedfilein
bedfileout]][-reordercolsmodelfile][-stateordering emission|transition] inputmodel
outputdir
```

Note items in [] are optional

Required Parameters

`inputmodel` – The file for the model that should be reordered

`outputdir` – The directory to which the reordered model, emission, and transition files should be written

Optional Parameters

`-f columnorderingfile` – A text file which gives the ordering for the names of the columns one per line

`-m labelmappingfile` – This option can specify a file which maps state IDs to descriptive names. The descriptive names can be appended to the state IDs in the emission and transition files. The format of this file is a two column tab delimited file. The first column contains each state ID with the state ordering letter prefix. The second column contains a descriptive name or mnemonic. If this file is not specified, then just the state IDs are displayed. If reordering the states when applying this option the state number and prefix should match the reordered model. So if specifying a `stateorderingfile`, then the prefix should be U and the second column should have the new states. It can be simpler to relabel after reordering the states.

`-noimage` – If this flag is present, then outputting of image files is suppressed.

`-o stateorderingfile` – The contents of this file gives a user supplied ordering of states. Giving such a file will override the `stateordering` option if that is also specified. The format of a stateordering file is a two column tab delimited file. The first column is the old state number and the second column is the new state number in order. The state prefix should not be given.

`-r bedfilein bedfileout` – If a state ordering file is specified with the `-o stateorderingfile`, then this option can also be given where `bedfilein` is a bed file produced from LearnModel, MakeSegmentation, or MakeBrowserFiles `and bedfilein` is the output file in which the contents of `bedfilein` should be written after relabeling the state assignments according to the updated ordering in the `stateorderingfile` file.

`-reordercolsmodelfile` – If this flag is present the columns in the model file are reordered, which should be done if the model is being applied with binarized data with the columns in a different order than the order when used to learn the model and the new desired ordering can be specified by using the `-f columnorderingfile` option.

`-stateordering emission|transition,-color r,g,b, -holdcolumnorder,`
`-i outfileID,` have the same meaning as in LearnModel

EvalSubset

Description

This command evaluates the extent to which an existing segmentation can be recovered using only a subset of marks. The method takes an input the model used to generate an existing segmentation, the data used to generate the segmentation, the existing segmentation, an output file prefix, and which subset of marks to include. The output is a confusion matrix comparing the state assignments based on the full set of marks and those based on using the model learned on the full data but with the state assignments generated based only on a subset of inputs. Each row of the confusion matrix indicates the proportion of state assignments based on the full marks assigned to each state using a subset of marks (columns).

Usage

```
usage: EvalSubset [-append][-b binsize][-color r,g,b][-f inputfilelist][-i
outfileID][-lowmem][-many][-noimage][-readposterior|-readstatesbyline][-scalebeta]
inputmodel inputdir segmentdir outconfusionfileprefix includemarks
```

Note items in [] are optional

Required Parameters

`inputmodel` – The file containing the model parameters

`inputdir` – This is the directory containing the binarized input files. Only file names containing '_binary' are used by default.

`segmentdir` – The directory contains the segmentation bed corresponding to the input data in `inputdir`. If –readposterior or –readstatesbyline is specified then this directory is the directory containing the POSTERIOR and STATEBYLINE directory.

`outconfusionfileprefix` – The prefix for the output files containing the confusion matrix in .txt, .png, and .svg form. The '.txt' file output also lists the name of the marks included in the subset.

`includemarks` – A binary bit string indicating for each mark if it should be included in the subset ('1') or excluded ('0'). The order of the bits should correspond to the order of the marks in the `inputmodel` and the data files in inputdir. For instance if the marks in a data file in order are (H3K4me1,H3K4me3,H3K27me3) the bit string 101 would correspond to including H3K4me1 and H3K27me3, but excluding H3K4me3.

Optional Parameters

`-append` – If this flag is present then the output is appended to the end of outconfusionfileprefix.txt , by default it is overwritten.

`-readposterior|-readstatesbyline` – By default the input are four column segmentation files, but a posterior probability (`-readposterior`) or single states per line (`-readstatesbyline`) as generated by `-printstatebyline` files could alternatively be used. If the posterior is used the confusion matrix is based on soft assignments as described in Ernst and Kellis, Nature Biotech 2010.

`-b binsize, -color r,g,b, -f inputfilelist, -i outfileID, -lowmem, -many, -noimage, -scalebeta` have the same meaning as in LearnModel

ConvertGeneTable

Description

This command takes as input a gene table in a format provided by the UCSC genome table browser and generates external annotation files in the COORDS and ANCHORFILES directories. The input format is a tab delimited format with the first set of columns in order being: bin, name, chrom, strand, txStart, txEnd, cdsStart, cdsEnd, exonCount, exonStarts, exonEnds. All coordinates are 0-based. The command generates in the COORDS directory files with transcription start sites (TSS), transcription end sites (TES), gene, exon, and promoter regions annotations. Promoter region annotations are defined based on a window around the TSS which defaults to +/-2000bp. The command also generates in the ANCHORFILES directory position files for the TSS with strand information.

Usage

```
usage: ConvertGeneTable [-gzip][-l chromosomelengthfile][-u coorddir][-v
anchordir][-w promoterwindow] inputgenetable prefix assembly
```

Note items in [] are optional

Required Parameters

inputgenetable – The file containing the gene annotation information.

prefix – This parameter specifies the prefix in the file name for all external annotation files generated.

assembly – This parameter specifies the assembly of the annotation files.

Optional Parameters

-gzip – If this flag is present then the external annotation files are outputted in a gzip format.

-l chromosomelengthfile – This parameter specifies the file containing the length of the chromosomes. It is a two column tab delimited file with the first column specifying the chromosome name and the second column the length. The information in this file is used to prevent the window around the TSS from going over the end of the chromosome. By default this is the file assembly.txt in the CHROMSIZES directory where the ChromHMM.jar file is and assembly is the specified assembly. Note that if the parameter is not specified the file still must exist in the default directory.

-u coorddir – This parameter specifies the directory of subdirectories where each subdirectory contains coordinate files for a specific assembly for OverlapEnrichment. By default this is the COORDS directory where the ChromHMM.jar file is.

-v anchorfiledir – This parameter specifies the directory of subdirectories where each subdirectory contains anchor files for a specific assembly for NeighborhoodEnrichment. By default this is the ANCHORFILES directory where the ChromHMM.jar file is.

-w promoterwindow – This parameter specifies in base pairs the window around the TSS to include in one of the annotation files. Bases +/- promoterwindow from the TSS are included. By default this parameter is set to 2000.

MergeBinary

Description

The command can do a columnwise merge of binarized data for different mark subsets split across files in multiple subdirectories. Files that are for the same cell type and chromosome are merged. If a chromosome for a cell type is found in one subdirectory, but not another, then a warning message is printed and not present values are used for that chromosome. This command can also be used to do a row split on the input files, including if only one subdirectory is provided and no merging is done.

Usage

```
usage MergeBinary [-f dirlistfile][-gzip][-splitrows [-j numsplitbins]] inputdir outputdir
```

Note items in [] are optional

Required Parameters

`inputdir` – A directory containing subdirectories where each subdirectory has binarized data.

`outputdir` – The directory to which the output binarized files should be written.

Optional Parameters

`-f dirlistfile` – If this flag is present, then only the subdirectories of `inputdir` listed in `dirlistfile` will be used, otherwise all subdirectories in `dirlistfile` will be used

`-gzip` – If this flag is present then the external annotation files are outputted in a gzip format.

`-splitrows` – If the flag is present then the chromosome files are split across multiple files, with the maximum number of bins in any file given by numsplitbins. Splitting chromosome files can be desired in large scale applications where because of time or memory constraints ChromHMM cannot handle the size of the data without splitting, and in particular when using `-n numseq` option of LearnModel.

`-j numsplitbins` – If the `-splitrows` option is specified, then this parameter can be adjusted to change the maximum number of rows that will be placed in any split file. The default value of this parameter is 5000, which corresponds to covering a 1MB under the default setting of `binsize`.

StatePruning

Description

This command takes as input a set of learned models generally learned across different initializations and then generates a nested set of models by pruning states from the highest scored model. The highest score model is the state with the highest estimated log-likelihood. States are greedily pruned from the highest scoring model. The criteria to a prune a state is that it causes the least decrease on the total distance for all the states in other models to their nearest state in the pruned model. If there is only one model, then distance to other states in that model are considered. This can be used as one heuristic to initialize model parameters to get a roughly comparable set of models across different number of states while biasing the learning procedure to avoid redundant or non-representative states. By default ChromHMM uses a simpler single pass method for producing nested initializations.

Usage

```
StatePruning [-correlation] inputdir outputdir
```

Note items in [] are optional

Required Parameters

`inputdir` – The name of the directory containing model files to eliminate. Only files with the prefix 'model_' in the directory are used.

`outputdir` –The name of the output directory where the nested set of model files should go. Each model file is prefixed with the name 'elim_'.

Optional Parameters

`-correlation` – An optional flag indicating to use 1-correlation coefficient to define distance instead of the default of the euclidean distance

---

Acknowledgements

The software uses the open source jheatchart library (slightly modified), the batik library, and HTSJDK. RefSeq coordinates and CpG island coordinates obtained from the UCSC genome browser, and nuclear lamina data from (Guelen et al., Nature 2008) obtained through the UCSC genome browser are also included.