CoNVex: Copy Number Variation from Exomes

June 25, 2013

Breakpoints *Estimate Breakpoints using Number of Reads*

Description

MAD of the log2 ratio varies within a set of regions based on the number of reads that hit the probe region. Based on this, MADs are calculated. For that purpose, probe regions are binned according to the number of reads making sure that each bin has <max_bin_size> number of probe regions. Default (1000) works fine in most cases. This function returns the max of the total reads for each bin (breakpoints using cut function).

Usage

```
Breakpoints(RDfile_RepSample,max_bin_size=1000)
```

Arguments

`RDfile_RepSample`
: Read depth file of a representative sample in the analysis batch (any sample should work)

`max_bin_size`
: Bin size for number of probe regions from which MAD is estimated (default=1000 should be enough for most cases)

Details

Breakpoints.

Note

Differences between exome samples generated with same experimental protocols/parameters should not make a huge difference between samples. Each sample can use the same breakpoints generated from a representative sample. However, you may like to select a representative sample using the Number of total mapped reads per sample. i.e. Consider a sample that has the median number of total reads across samples!

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

BreakpointsCall, BreakpointsCallCommands, cut

Examples

```
> BP = Breakpoints(RDfile_RepSample=RDfiles[1],max_bin_size=1000)
```

BreakpointsCallCommands

Command Generator for using BreakpointsCall

Description

Generates a Unix command (one per batch) to calculate Breakpoints using BreakpointsCall R program

Usage

```
BreakpointsCallCommands(RDfile_RepSample,max_bin_size=1000,\
BPfile,Rbatch_folder="")
```

Arguments

`RDfile_RepSample`
: Read depth file of a representative sample in the analysis batch (any sample should work)

`max_bin_size`
: Bin size for number of probe regions from which MAD is estimated (default=1000 should be enough for most cases)

`BPfile`
: Breakpoints file generated by BreakpointsCall

`Rbatch_folder`
: [optional] Rbatch folder bundled with CoNVex for batch execution. Rbatch_folder="" (default, works mostly) assumes that it's in R's library folder. You may copy it to somewhere else and modify this explicitly

Details

BreakpointsCallCommands.

Note

This can be easily used together with batch queuing (e.g., bsub) programs. This function usually one Unix command for all samples in the batch.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

BreakpointsCall, Breakpoints, cut

Examples

```
> gam_commands = BreakpointsCallCommands(RDfile_RepSample,\
max_bin_size=1000,BPfile,Rbatch_folder="")
> gam_commands[1]
[1] "R --vanilla --slave --args '/home/pv1/UK10K/ProbeRD_UK10K12345.dat,\
/home/pv1/UK10K/MOPD_Features.txt,\
/home/pv1/UK10K/L2R/GAM_MOPD5095427.dat,\
/home/pv1/UK10K/ProbeRD_MOPD5095427.dat,\
/home/pv1/UK10K/MOPD_Breakpoints.txt' < \
/home/pv1/R/x86_64-linux/2.11/CoNVex/Rbatch/GAMCorrectionPerSample.R"
```

CNV2BED *Create UCSC uploadable bed file from CNV calls*

Description

This functions accepts the data frame created by GetCNVCalls() function and creates a bed file in a given name. You can either upload this to UCSC genome browser for a custom track or use it for other purposes.

Usage

```
CNV2BED(CNVcalls=data.frame(),Bedfile="CoNVexCNVs_UCSC.bed", db="hg19")
```

Arguments

CNVcalls	CNV calls from all samples - output from GetAllCalls()
Bedfile	Name of the bed file to be created. Default: CoNVexCNVs_UCSC.bed
db	Genome build; Default: Assumes that the coordinates are in 'hg19' format

Details

GetCNVCalls() function returns columns in this format: "chr", "start", "end", "num_probes", "convex_score", "cnv_type", "sample_id". This format is expected. DELS and DUPS are listed separately and coloured in red and blue respectively.

Note

After creating the basic bed file, you can open it and modify few things to suit your purpose.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GetCNVCalls

Examples

```
UK10KCNVCalls = GetCNVCalls(CNVfiles=cnv_call_files)
CNV2BED(CNVcalls=UK10KCNVCalls)
```

EWScore *Calculate MAD-weighted Scores*

Description

This function calculates error-weighted scores from corrected log2 ratio and regional MADs. The calculation is partly similar to Agilent's ADM2 algorithm, but the exome log2 ratio and MAD-based calculation of weights are unrelated to ADM2 which was developed for aCGH data. Regional MADs are first calculated using number of reads and breakpoints.

Usage

```
EWScore(d,doNorm=0)
```

Arguments

`d`	Data frame having the following columns: chr, start, end, log2 ratio (corrected), MAD (regional)
`doNorm`	doNorm=0 (default) does not do any normalization (recommended). Other values normalize (Z-score) the log2 ratio. In future, this can be extended for better normalization if required.

Details

EWScore

Note

This function is internally called from GAMCorrection function. Changing parameters are recommended only for advanced users (mostly CoNVex authors)

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

Agilent G4175AA CGH Analytics 3.4 User Guide, ADM2 algorithm

See Also

GAMCorrection
Agilent user guides: http://www.chem.agilent.com/

Examples

```
> ADMrep = EWScore(d, doNorm=0)
```

`GAMCorrection` *Systematic Error Correction of log2 Ratio*

Description

GAMCorrection corrects log2 ratio for systematic errors caused by GC content, deltaG and Tm. GC content is the proportion of GC in probe regions that causes systematic variation in read depth. deltaG is the free energy of hybridization between biotinylated RNA probes and sheared genomic DNA. Tm is the melting temperature of hybridization. deltaG and Tm are thermodynamic features that estimate the systematic fluctuation of read depth caused by variation in hybridization efficiency between RNA probes and genomic DNA. This function depends on ’mgcv’ package.

Usage

```
GAMCorrection(L2Rfile,features_file,RDfile,BPfile,doNorm=0)
```

Arguments

`L2Rfile`	File containing log2 ratio of a specific sample
`features_file`	A subset of probe regions file with rowcount, Chr regions, GC, deltaG and Tm as columns
`RDfile`	Read depth file of a specific sample
`BPfile`	Breakpoints file generated by Breakpoints / BreakpointsCall
`doNorm`	doNorm=0 (default) does not do any normalization (recommended). Other values normalize (Z-score) the log2 ratio. In future, this can be extended for better normalization if required.

Details

GAMCorrection function corrects systematic errors in log2 ratio (of depth/sample-median) for each probe region. This function uses the gam() function of the ’mgcv’ package and its multivariate cubic splines model to correct the log2 ratio using GC, deltaG and Tm. deltaG and Tm are interrelated thermodynamic features So, gam() considers them as causing synergetic effect and uses s(deltaG,Tm). GAMCorrectionPerSample.R calls it for all samples. GAMCorrectionCommands generates Unix commands for batch execution.

Note

This can be easily used together with batch queuing (e.g., bsub) programs. This function usually one Unix command per sample in the analysis.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

Wood, S.N. (2011) Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. Journal of the Royal Statistical Society (B) 73(1):3-36

See Also

GAMCorrectionCommands, GAMCorrectionPerSample
mgcv:gam - http://people.bath.ac.uk/sw283/mgcv/

GAMCorrectionCommands *Unix Command Generator for using GAMCorrectionPerSample*

Description

Generates Unix command(s) (e.g., one per batch) to calculate corrected Log2 Ratio and error-weighted Scores

Usage

```
GAMCorrectionCommands(features_file,L2Rfiles,GAMfiles,RDfiles,\
BPfile,output_folder,sample_ids,Rbatch_folder="")
```

Arguments

features_file	A subset of probe regions file with rowcount, Chr regions, GC, deltaG and Tm as columns
L2Rfiles	Vector of files containing log2 ratio generated by SampleLogRatio
GAMfiles	Vector of file names to save the corrected log2 ratio and error-weighted scores
RDfiles	Vector of read depth files
BPfile	Breakpoints file generated by BreakpointsCall
output_folder	Output folder for read depth files
sample_ids	Vector of Sample IDs
Rbatch_folder	[optional] Rbatch folder bundled with CoNVex for batch execution. Rbatch_folder="" (default, works mostly) assumes that it's in R's library folder. You may copy it to somewhere else and modify this explicitly

Details

GAMCorrection function corrects systematic errors in log2 ratio (of depth/sample-median) for each probe region. GAMCorrectionPerSample.R calls it for all samples. GAMCorrectionCommands (this function) generates Unix commands for batch execution. This function depends on 'mgcv' package.

Note

This can be easily used together with batch queuing (e.g., bsub) programs. This function usually one Unix command per sample in the analysis.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

Wood, S.N. (2011) Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. Journal of the Royal Statistical Society (B) 73(1):3-36

See Also

GAMCorrection, GAMCorrectionPerSample,
mgcv:gaml - http://people.bath.ac.uk/sw283/mgcv/

Examples

```
> gam_commands = GAMCorrectionCommands(features_file,L2Rfiles,\
GAMfiles,RDfiles,BPfile,output_folder,\
sample_ids,Rbatch_folder="");
> gam_commands[1]
[1] "R --vanilla --slave --args '/home/pv1/UK10K/ProbeRD_UK10K12345.dat,\
/home/pv1/UK10K/MOPD_Features.txt,\
/home/pv1/UK10K/L2R/GAM_MOPD5095427.dat,\
/home/pv1/UK10K/ProbeRD_MOPD5095427.dat,\
/home/pv1/UK10K/MOPD_Breakpoints.txt' < \
/home/pv1/R/x86_64-linux/2.11/CoNVex/Rbatch/GAMCorrectionPerSample.R"
```

GenomePlot *Creates a genome-wide plot of all exome bait regions in the GAMfile*

Description

Creates a genome-wide plot of all exome bait regions' ADM scores in the GAMfile. Input requires a single sample's GAMfile and its sample_id.

Usage

```
GenomePlot(GAMfile,sample_id="",output_folder)
```

Arguments

`GAMfile`	GAM file name of a sample - output from GAMCorrection
`sample_id`	sample_id; default: ” - GAMfile used in place for plotting
`output_folder`	Output folder for storing plots

Details

GenomePlot

Note

# Output contains a genomeplot in the output folder. Use this function within a for() loop for multiple samples

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GetCNVCalls

Examples

```
>#
```

`GetCNVCalls` *Returns CNV calls on all samples*

Description

Returns CNV calls on all samples. Takes CNVfiles (file names – one for each sample) as input and returns a data frame.

Usage

```
GetCNVCalls(CNVfiles)
```

Arguments

`CNVfiles`	CNV file names contains deletions and duplications – one for each sample
`names`	(default=1) Assigns column names; no names otherwise;

Details

GetCNVCalls

Note

# Data frame Columns: "chr","start","end","num_probes","convex_score","cnv_type","sample_id"

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SWCNV, SWCNVCall, SWCNVCallCommands

Examples

```
> CallsAll = GetCNVCalls(CNVfiles);
```

GetCNVStartEnd *Get Centromere Boundaries in Probe Regions*

Description

This function determines centromere boundaries within probe regions by binning them separately for each chromosome arm (before and after centromere). While using the Smith-Waterman algorithm, CNVs (change points) are not merged across the centromere. This function is used internally by SWCNV.

Usage

```
GetCNVStartEnd(d,centromeres)
```

Arguments

`d`	Data frame having the following columns: chr, start, end, log2 ratio (corrected), MAD (regional)
`centromeres`	Centromere co-ordinates for each chromosome (as in UCSC) are supplied in 3 columns: chr, start, end

Details

GetCNVStartEnd

Note

This function is internally called from SWCNV function. For different builds (hg19, hg18), use different centromere regions file to supply the co-ordinates for the 'centromere'. hg19 gaps table has been included with CoNVex package. For different organisms and chromosome builds, you can download the gaps table from the UCSC genome browser (link below). How to download the gaps table? [1] Open the URL below ('see also' section) in a web browser [2] In the 'group:' option, select 'All tables' [3] In the 'table:' option, select 'gap'. [4] In the 'output format:' option, select 'All fields from select table' [5] In the 'output file:' option, enter a file name [6] Click 'get output' button to download the file [7] Use the file for the centromere_regions_file option in SWCNV (or in SWCNVCall, SWCNVCallCommands appropriately) [8] This function is normally called by SWCNV

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

UCSC genome browser

See Also

SWCNV, SWCNVCall, SWCNVCallCommands Download UCSC gaps table from here: http://genome.ucsc.edu/cgi-bin/hgTables

Examples

```
>  Fdup_reportX = GetCNVStartEnd(d,Fdup_report);
```

GetCentroBoundaries *Get Centromere Boundaries in Probe Regions*

Description

This function determines centromere boundaries within probe regions by binning them separately for each chromosome arm (before and after centromere). While using the Smith-Waterman algorithm, CNVs (change points) are not merged across the centromere. This function is used internally by SWCNV.

Usage

```
GetCentroBoundaries(d,centromeres)
```

Arguments

d	Data frame having the following columns: chr, start, end, log2 ratio (corrected), MAD (regional)
centromeres	Centromere co-ordinates for each chromosome (as in UCSC) are supplied in 3 columns: chr, start, end

Details

GetCentroBoundaries

Note

This function is internally called from SWCNV function. For different builds (hg19, hg18), use different centromere regions file to supply the co-ordinates for the 'centromere'. hg19 gaps table has been included with CoNVex package. For different organisms and chromosome builds, you can download the gaps table from the UCSC genome browser (link below). How to download the gaps table? [1] Open the URL below ('see also' section) in a web browser [2] In the 'group:' option, select 'All tables' [3] In the 'table:' option, select 'gap'. [4] In the 'output format:' option, select 'All fields from select table' [5] In the 'output file:' option, enter a file name [6] Click 'get output' button to download the file [7] Use the file for the centromere_regions_file option in SWCNV (or in SWCNVCall, SWCNVCallCommands appropriately) [8] This function is normally called by SWCNV

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

UCSC genome browser

See Also

SWCNV, SWCNVCall, SWCNVCallCommands
Download UCSC gaps table from here:
http://genome.ucsc.edu/cgi-bin/hgTables

Examples

```
> centro_boundaries = GetCentroBoundaries(d,hg_cent);
```

GetFrequency — *Get frequency within or between Chr regions*

Description

This function accepts upto two lists (as data frames) of chromosome regions as CNV calls. Each list should have chr, start, end (chr must be 1,2,3,...,X,Y) as the first 3 columns. Additionally, sample_id must be provided in the 7th column as in the CoNVex output of CNV calls. It returns the number of times each region (including itself) is encountered in the Chr (CNV) regions list. If a region in one sample overlaps multiple (split) regions in another sample, it's counted only once. So, the count is the number of samples (including the sample to which the CNV belongs) that have at least one overlapping CNV region. If you divide the count with the number of total samples in the list, you will get a proportion. chr_list1 is returned with two additional columns at the end having the frequency as count and percentage. Refer to the details section below.

Usage

```
GetFrequency(chr_list1,chr_list2,tmp_folder="/tmp/")
```

Arguments

`chr_list1`	(mandatory) List of Chr regions (e.g., CNV calls)
`chr_list2`	(optional) List of Chr regions to compare against chr_list1
`sid_col1`	(optional) Sample ID column in chr_list1; Default: column 7
`sid_col2`	(optional) Sample ID column in chr_list2; Default: column 7
`ro_threshold`	(optional) Reciprocal overlap threshold in percentage; Default: > 0 (any overlap)
`tmp_folder`	(Default: '/tmp/') (optional) Temporary files folder

Details

GetFrequency calculates the number of times each region (including itself) is encountered in the Chr regions list. If chr_list1 is the only given list, GetFrequency() calculates the internal frequency within the list through ANY overlap between regions. If chr_list2 is ALSO given, this is the number of samples in chr_list2 in which each CNV/Chr region from chr_list1 is seen.

Possible use cases: (1) CNV calls' internal frequency (within chr_list1 only with sample_id) (2) CNV calls (chr_list1 with sample_id) and the number of times each call is seen in AN EXTERNAL BATCH of CNV calls (chr_list2 with sample_id) This function simplifies and conveniently encapsulates the complexity of CNV frequency estimation.

Note

Please include CoNVex.jar in classpath if it is not configured previously (you might have already done this for executing 'java ReadDepth' program).

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GetCNVCalls

Examples

```
GetFrequency(chr_list1) # or
GetFrequency(chr_list1,chr_list2)
```

`GetGAMScores` *Returns a list of all samples' log2 ratio and ADM scores*

Description

Returns a list of all samples' log2 ratio and ADM scores. Returned list contains 2 matrices - one with log2 ratio, other with ADM scores

Usage

```
GetGAMScores(GAMfiles)
```

Arguments

`GAMfiles`	GAM file names - output files of GAMCorrection

Details

GetGAMScores

Note

# Returned list contains 2 matrices - one with log2 ratio, other with ADM scores

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GAMCorrection, GAMCorrectionCommands

Examples

```
> L2R_ADM = GetGAMScores(GAMfiles);
```

`GetKnownCNVPC` *Returns known CNVs (percentage) in a data frame*

Description

For a given data frame of CNV calls, this function checks whether each CNV is known in the consensus CNV list bundled with the CoNVex package. If so, it calculates the overlap (forward overlap - percentage of the length of the call present in the Consensus CNV list). It returns a data frame with all columns in the input CallsAll data frame together with a percentage column at the end.

Usage

```
GetKnownCNVPC(CallsAll,known_dels=data.frame(),known_dups=data.frame())
```

Arguments

`CallsAll`	CNV calls in all samples - output of GetAllCalls
`known_dels`	Known deletions, if you like to use a different list of known CNVs
`known_dups`	Known duplications, if you like to use a different list of known CNVs

Details

GetKnownCNVPC uses GetOverlap() internally. The java binaries (jar files bundled with this R package) must be in classpath to execute this function.

Note

Please include CoNVex.jar and other jars in classpath if it is not configured previously (you might have already done this for executing 'java ReadDepth' program).

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GetCNVCalls, GetOverlap, getClassPath

Examples

```
> setwd("~/")
> calls = GetCNVCalls(CNVfiles);
> GetKnownCNVPC(CallsAll=calls)
```

`GetOverlap` *Get overlap between two lists of chromosome regions*

Description

This functions accepts two lists of chromosome regions. Each list should have the following columns (1-3): chr, start, end (chr must be 1,2,3,...,X,Y). It returns max percentage overlap of each region in chr_list1 with any of the regions in chr_list2. i.e. chr_list2 is returned with an additional column at the end having the percentage.

Usage

```
GetOverlap(chr_list1,chr_list2)
```

Arguments

`CNVcallsAll`	CNV calls in all samples - output of GetAllCalls
`file_prefix`	file name prefix for plots; default: CNVstats
`outlier_cutoff`	The y-axis ratio threshold marking outliers in 'deletions/duplications' plot; default=2

Details

GetOverlap is internally used by other functions. The ConVex java binary (CoNVex.jar bundled with CoNVex R package) must be in classpath to execute this function.

Note

Please include CoNVex.jar in classpath if it is not configured previously (you might have already done this for executing 'java ReadDepth' program).

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GetCNVCalls

Examples

```
GetOverlap(chr_list1,chr_list2)
```

MeansMadsCommands *Mean log2 ratio and MAD of mean log2 ratios for each CNV call*

Description

Mean log2 ratio and MAD of mean log2 ratios for each CNV call.

Usage

```
MeansMadsCommands(CNVfiles,sample_ids,cor_matrix,sample_info_file,regions_file, features_file, a
```

Arguments

`CNVfiles`	Vector of output files to store CNV calls
`sample_ids`	Vector of Sample IDs
`cor_matrix`	Correlation matrix
`sample_info_file`	Tab-separated file auto-generated by SamplePrepInfo() function - required for the whole analysis
`regions_file`	Probe regions file
`features_file`	A subset of probe regions file with rowcount, Chr regions, GC, deltaG and Tm as columns
`all_samples`	[default = 0] Consider all samples in the analysis to calculate means and mads (no need for cor_matrix file; suitable for small projects)
`Rbatch_folder`	[optional] Rbatch folder bundled with CoNVex for batch execution. Rbatch_folder="" (default, works mostly) assumes that it's in R's library folder. You may copy it to somewhere else and modify this explicitly
`output_folder`	[optional] Output folder

Details

MeansMadsCommands calls MeansMadsCall

Note

# Output file name is auto-generated (ending with: MeansMads.txt)

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

MeansMadsCall is a batch execution script

Examples

```
> #
```

`MultiPanelPlot` *Zoomed-in visualisation of each CNV calls*

Description

Plots a CNV call and other information in multiple panels.

Usage

```
MultiPanelPlot(allX,allXscore,Fd_rep,sample_id,known_rstr, gene_file="",misc_str="",mother_id=""
```

Arguments

`allX` Contains log2 ratio of sample(s) to be plotted and a set of background samples

`allXscore` Contains error-weighted scores of sample(s) to be plotted and a set of background samples

`Fd_rep` Single CNV call - a row in the data frame generated by GetCNVCalls()

`sample_id` Sample ID of this CNV call - scores of this sample must be present in allX and allXscore

`known_rstr` List of known CNV calls in this region

`gene_file` A file containing genes and transcripts in columns 5 and 6. Col1=rowcount, Col2-3:

`misc_str` Misc short string that goes in the legend

`mother_id` Mother's sample id if present in allX

`father_id` Father's sample id if present in allXscore

Details

MultiPanelPlot has 3 panels: (1) log2 ratio of CNV call; (2) Error-weighted score of CNV call; (3) Information panel

Note

# Output contains 1 plot in the current directory. You may like to use setwd("your_folder") to set the current directory. File names of the png files are determined automatically. Rows in gene_file (if present) should match the rows in regions_file used for CNV calling. Columns of the file include: rowcount, chr, start, end, gene_symbols, transcripts. (you can substituet gene symbols and transcripts with any other relevant information instead.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

PlotCNVStats

Examples

```
> #
```

PlotCNVStats   *Creates plots of (1) number of deletions and duplications, and (2) ratio of deletions/duplications in each sample*

Description

Plots CNV calls stats grom a given data frame of CNV calls.

Usage

```
PlotCNVStats(CNVcallsAll,file_prefix="CNVstats",outlier_cutoff=2)
```

Arguments

`CNVcallsAll`	CNV calls in all samples - output of GetAllCalls
`file_prefix`	file name prefix for plots; default: CNVstats
`outlier_cutoff`	The y-axis ratio threshold marking outliers in 'deletions/duplications' plot; default=2

Details

PlotCNVStats

Note

# Output contains 2 plots in the current directory. You may like to use setwd("your_folder") to set the current directory

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

GetCNVCalls

Examples

```
> setwd("~/")
> calls = GetCNVCalls(CNVfiles);
> PlotCNVStats(calls);
```

`PlotKnownCNVStats` *Known (percentage) CNV Calls in Samples*

Description

Plots the percentage of known CNVs in the CNV calls

Usage

```
PlotKnownCNVStats(CallsAll,known_dels=data.frame(),known_dups=data.frame(),mark_outliers=0,outli
```

Arguments

`CallsAll` CNV calls in all samples (data frame) - output of GetAllCalls

`known_dels` Known deletions in a data frame (default: uses the list supplied by CoNVex, if this option is not used)

`known_dups` Known duplications in a data frame (default: uses the list supplied by CoNVex, if this option is not used)

`mark_outliers` default=0 (don't marks outliers). =1 marks outliers; check the next option

`outlier_from_median`
Marks outliers that are <outlier_from_median>% away from the median. Default: N=5; (ignored if mark_outliers = 0)

`file_prefix` Prefix for the file name (default: CNVstats_Known

Details

PlotKnownCNVStats uses the GetOverlap() function internally. This function calls the java program that gets the overlap. By default, temporary files re created in /tmp folder before estimating the overlap. You can change this option in GetOverlap() function.

Note

# Output contains 1 plot in the current directory. You may like to use setwd("your_folder") to set the current directory, and an explicit file name (should you need one!)

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

PlotKnownCNVStats, GetOverlap, getClassPath

Examples

```
> setwd("~/")
> calls = GetCNVCalls(CNVfiles);
> PlotKnownCNVStats(CallsAll=calls);
```

`PlotKnownCNVStatsV2` *Number of Calls vs. Known (percentage) CNV Calls*

Description

Plots the #Calls vs. Known (percentage) for all samples and saves a call stats file

Usage

```
PlotKnownCNVStatsV2(CallsAll,known_dels=data.frame(),known_dups=data.frame(),mark_outliers=0,out
```

Arguments

`CallsAll` CNV calls in all samples (data frame) - output of GetAllCalls

`known_dels` Known deletions in a data frame (default: uses the list supplied by CoNVex, if this option is not used)

`known_dups` Known duplications in a data frame (default: uses the list supplied by CoNVex, if this option is not used)

`mark_outliers` default=0 (don't marks outliers). =1 marks outliers; check the next option

`outlier_from_median`
Marks outliers that are <outlier_from_median>% away from the median. Default: N=5; (ignored if mark_outliers = 0)

`file_prefix` Prefix for the file name (default: CNVstats_NumCalls_Known

Details

PlotKnownCNVStatsV2 uses the GetOverlap() function internally. This function calls the java program that gets the overlap. By default, temporary files re created in /tmp folder before estimating the overlap. You can change this option in GetOverlap() function.

Note

# Output contains 1 plot in the current directory. You may like to use setwd("your_folder") to set the current directory, and an explicit file name (should you avoid the default!)

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

PlotKnownCNVStatsV2, GetOverlap, getClassPath

Examples

```
> setwd("~/")
> calls = GetCNVCalls(CNVfiles);
> PlotKnownCNVStatsV2(CallsAll=calls);
```

ReadDepthCommands *Unix Command Generator for ReadDepth java program*

Description

Generates Unix commands (e.g., one per sample) to calculate read depth.

Usage

```
ReadDepthCommands(regions_file, sample_ids, bamfiles, \
bamindex_files=NULL, RDfiles, output_folder="", chr_prefix="", max_memory=2)
```

Arguments

`regions_file`	Probe regions file
`sample_ids`	Vector of Sample IDs
`bamfiles`	Vector of BAM files - rows should match sample_ids
`bamindex_files`	[optional] Vector of BAM index files - not required if they are present in the BAM file folder and use SAM/BAM format's accepted naming convention
`RDfiles`	Vector of output read depth files
`output_folder`	[optional] Output folder for read depth files
`max_memory`	[optional] default=2 (for 2GB), Use according to the size of BAM files

Details

Calculates (1) mean read depth of a probe region (any chr region should work) and (2) number of sequence reads falling in that region

Note

CoNVex is a read depth based CNV detection algorithm. This function is useful to generate batch execution commands for a large scale project. The commands execute the java program ReadDepth which is bundled with CoNVex package. CoNVex.jar, args4j-<version>.jar, and sam-<version>.jar should be in your CLASSPATH environment variable.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

Picard, args4j (links below)

See Also

http://picard.sourceforge.net/
http://java.net/projects/args4j/
http://javarevisited.blogspot.co.uk/2011/01/how-classpath-work-in-java.html
http://samtools.sourceforge.net/SAM1.pdf

Examples

```
> rdc = ReadDepthCommands(regions_file=regions_file,sample_ids=sample_ids,\
bamfiles=bamfiles,RDfiles=RDfiles,chr_prefix="Chr",output_folder=output_folder,\
max_memory=2);
> rdc[1]
[1] "java -Xmx2g ReadDepth -bam_file /home/pv1/reads.474946.recal.bam \
 -chr_prefix Chr \
 -regions_file /home/pv1/R/x86_64-linux/2.11/CoNVex/extdata/SureSelect_50Mb.txt \
 -rd_file /home/pv1/UK10K/ProbeRD_UK10K12345.dat"
 >
 > system("java ReadDepth -help")
 -bam_file VAL      : bam or cram file location of the sample
 -bamindex_file VAL : (optional) Index file of the bam file, if exists in
                      different folder or in a different file name
 -chr_prefix VAL    :  (optional) Chr prefix in bam files [if you observe
                      'Chr1', use 'Chr' (without quotes). Default: no prefix
                      (expects 1,2,..X,Y)]
 -rd_file VAL       : Output file: No. of reads and mean depths of the given
                      probe regions
 -regions_file VAL  : File with Chr regions: chr (1,2,..X,Y), start, end
Required options are missing!
Usage: java ReadDepth -bam_file /path/to/file.bam -bamindex_file /path/to/bamindexfile.bai\
 -regions_file /path/to/file.txt \
 -rd_file /path/to/output/file/region_reads_depth.dat
Type -help or -<anyjunk> to display the options :)
```

SMMCallCommand — *Call SampleMeans and SampleMADsUnix through SampleMeansMads*

Description

SMMCallCommand creates R command with arguments to call SampleMeansMads. SampleMeansMads calculated sample mean depth and MAD(log ratio) using SampleMeans and SampleMADs.

Usage

```
SMMCallCommand(sample_info_file,regions_file,output_file,Rbatch_folder="")
```

Arguments

`sample_info_file`
: Sample information file created by SampleInfoPrep (as part of you analysis)

`regions_file`
: Probe regions file

`output_file` Sample means and MADs will be stored in this file

`Rbatch_folder` [optional] Rbatch folder bundled with CoNVex for batch execution. Default=”

Details

SMMCallCommand uses sample information file (especially read depth files and GAM files) to retrieve read depth and log2 ratio and stores sample means and MADs in the output_file.

Note

This is useful for SampleMeans vs. MADs plots that act as an visualisation of noise in the analysis batch.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleMADsUnix, SampleMeans, SampleMeansMads (in the Rbatch folder)

Examples

```
> SMMCallCommand(sample_info_file="path/to/sample_info_file",regions_file="path/to/regions_file",output_fil
> # execute the resulting command in the Unix prompt
```

`SWCNV` *Change point detection using Smith-Waterman algorithm*

Description

SWCNV uses pre-compiled swa_* binaries to call CNVs using the Smith-Waterman algorithm. This function acts as a wrapper for binary and extracts the calls through pipe() function. Calls are filtered as per the tdel and tdup values (t value thresholds for CoNVex score). Returns a data frame containing deletions and duplications. This function is run for each sample separately (one at a time). SWCNVCall, SWCNVCallCommands act as wrappers for batch execution.

Usage

```
SWCNV(p,tdel,tdup,dv,GAMfile,sample_id,\
centromere_regions_file,output_folder,sw_exec)
```

Arguments

`p` p value for Smith-Waterman algorithm

`tdel` t value threshold of CoNVex score (s) for the selection of deletion calls (default=5)

`tdup` t value threshold of CoNVex score (s) for the selection of duplication calls (default=5)

`dv`	convex_score/(num_probes^dv) >= t is used for selecting CNV calls - num_probes to the power of dv is used (default=0.5)
`GAMfile`	File containing corrected log2 ratio and error-weighted scores of a specific sample
`sample_id`	Sample ID
`centromere_regions_file`	Centromere regions file downloaded from the UCSC browser. Default: hg19 build's file bundled with CoNVex
`output_folder`	Output folder for storing CNV calls - CNVcalls subfolder is ideal
`sw_exec`	Smith-Waterman execution binary for change point detection - works similar to SW-array for array data

Details

SWCNV is called by SWCNVCall R program available from Rbatch folder. SWCNVCallCommands creates one Unix command per sample to call CNVs.

Note

This can be easily used together with batch queuing (e.g., bsub) programs. This function usually one Unix command per sample in the analysis.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

# Smith, Temple F.; and Waterman, Michael S. (1981). "Identification of Common Molecular Subsequences". Journal of Molecular Biology 147: 195<d0>197.

# Price, T.S., Regan, R., Mott, R., Hedman, A., Honey, B., Daniels, R.J., Smith, L., Greenfield, A., Tiganescu, A., Buckle, V., et al.(2005) SW-ARRAY: A dynamic programming solution for the identification of copy-number changes in genomic DNA using array comparative genome hybridization data. Nucleic Acids Res. 33:3455<d0>3464.

See Also

SWCNVCall SWCNVCallCommands

Examples

```
# Check the example of SWCNVCallCommands and the source code of SWCNVCall
```

SWCNVCallCommands *Unix Command Generator for using SWCNVCall R program*

Description

Generates Unix command(s) (e.g., one per batch) to call CNVs using the Smith-Waterman algorithm

Usage

```
SWCNVCallCommands(p,swt_del,swt_dup,dv,GAMfiles,CNVfiles,sample_id,\
centromere_regions_file,output_folder,sw_exec,Rbatch_folder="")
```

Arguments

p	p value for Smith-Waterman algorithm
swt_del	t value threshold CoNVex score (s) for the selection of deletion calls (default=5)
swt_dup	t value threshold CoNVex score (s) for the selection of duplication calls (default=5)
dv	convex_score/(num_probes^dv) >= t is used for selecting CNV calls - num_probes to the power of dv is used (default=0.5)
GAMfiles	Vector of files containing corrected log2 ratio and error-weighted scores
CNVfiles	Vector of output files to store CNV calls
sample_ids	Vector of Sample IDs
centromere_regions_file	Centromere regions file downloaded from the UCSC browser. Default: hg19 build's file bundled with CoNVex
output_folder	Output folder for storing CNV calls - CNVcalls subfolder is ideal
sw_exec	Smith-Waterman execution binary for change point detection - works similar to SW-array for array data
Rbatch_folder	[optional] Rbatch folder bundled with CoNVex for batch execution. Rbatch_folder="" (default, works mostly) assumes that it's in R's library folder. You may copy it to somewhere else and modify this explicitly

Details

SWCNVCallCommands calls SWCNVCall R program available from Rbatch folder. It creates one Unix command per sample to call CNVs.

Note

This can be easily used together with batch queuing (e.g., bsub) programs. This function usually one Unix command per sample in the analysis.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

# Smith, Temple F.; and Waterman, Michael S. (1981). "Identification of Common Molecular Subsequences". Journal of Molecular Biology 147: 195-197.

# Price, T.S., Regan, R., Mott, R., Hedman, A., Honey, B., Daniels, R.J., Smith, L., Greenfield, A., Tiganescu, A., Buckle, V., et al.(2005) SW-ARRAY: A dynamic programming solution for the identification of copy-number changes in genomic DNA using array comparative genome hybridization data. Nucleic Acids Res. 33:3455-3464.

See Also

SWCNVCall, SWCNV
SW-Array: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1151590/

Examples

```
> p=2; swt_del=5; swt_dup=5; dv=0.5; # Smith-Waterman algorithm / CoNVex parameters
> sw_exec = paste(getLibPath(),"/CoNVex/exec/swa_lin64",sep=""); # X86_64 linux distribution
> output_folder = "/home/pv1/UK10K/CNVcalls/"; # CNVcalls sub-folder is ideal
> sw_commands = SWCNVCallCommands(p,swt_del,swt_dup,dv,GAMfiles,sample_id,\
centromere_regions_file,output_folder,sw_exec,Rbatch_folder="");
>
> sw_commands[1]
[1] "R --vanilla --slave --args '2,5,5,0.5,/home/pv1/UK10K/L2R/GAM_MOPD5095427.dat,\
MOPD5095427,/home/pv1/R/x86_64-linux/2.11/CoNVex/extdata/gaps_table_hg19.txt,\
/home/pv1/UK10K/CNVcalls/,\
/home/pv1/R/x86_64-linux/2.11/CoNVex/exec/swa_lin64' < \
/home/pv1/R/x86_64-linux/2.11/CoNVex/Rbatch/SWCNVCall.R"
```

`SampleInfoPrep` *Input Data Validation before Starting the Analysis*

Description

This function does basic validation with the input and creates output folders/files for further CNV analysis. This is an important first step to start the analysis. The output file (sample info file) created by this function has columns/info in a defined format. Returns the same info as a data frame.

Usage

```
SampleInfoPrep(sample_ids,gender,bamfiles,RDfiles,\
L2Rfiles,GAMfiles,sample_info_file,output_folder,overwrite=FALSE)
```

Arguments

`sample_ids`	Vector of Sample IDs
`gender`	Vector of Sample gender
`regions_file`	Probe regions file
`bamfiles`	Vector of BAM files - rows should match sample_ids
`RDfiles`	Vector of output read depth files

`L2Rfiles` Vector of file names in which uncorrected log2 ratio generated by SampleLogRatio will be stored - full path required

`GAMfiles` Vector of file names in which corrected log2 ratio and MAD-weighted scores generated by GAMCorrection will be stored - full path required

`sample_info_file` Output file name to store the all required sample information for further analysis

`output_folder` [optional] Output folder for read depth and other files

`overwrite` [optional; default=FALSE] By default, sample info file is not overwritten as a precaution. overwrite=TRUE overwrites it

Details

# Requires input info as shown in function options
# Creates output_folder if it doesn't exist
# Checks whether the BAM files exist
# Checks gender - only 'F', 'M' and 'U' are allowed
# Checks whether sample_info_file exists already - overwrite=TRUE (default: FALSE) overwrites it without warning
# Saves the sample_info_file in the specified output_folder with all required columns
# Output saved in sample_info_file is also returned as a data frame. You can keep it in an R interactive session for reference.

Note

This step is required for further analysis. Based on your expertise, you may create the sample_info_file yourself through other ways. You'll save a lot of time before the analysis by making sure all input and output files are right place by using this simple function.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

Agilent's ADM2 algorithm

See Also

CoNVex documentation

Examples

```
require(CoNVex)
baminfoRD = read.table("/path/to/SampleIDs_Gender_BAMfileLocations.txt", sep="\t")

# Required info
# regions - string / given with CoNVex package
regions_file = paste(getLibPath(),"/CoNVex/extdata/SureSelect_50Mb.txt",sep="");
sample_ids = as.character(baminfoRD[,3]);
bamfiles = as.character(baminfoRD[,4]);
gender = as.character(baminfoRD[,2]);
```

```
gender[gender==’Male’] = ’M’;
gender[gender==’Female’] = ’F’;
output_folder = ”/home/pv1/UK10K/”;
RDfiles = paste(output_folder,”/ProbeRD_”,sample_ids,”.dat”,sep=””)
L2Rfiles = paste(output_folder,”/L2R/L2R_”,sample_ids,”.dat”,sep=””)
GAMfiles = paste(output_folder,”/L2R/GAM_”,sample_ids,”.dat”,sep=””)
sample_info_file = ”/home/pv1/UK10K/UK10K_SampleInfo.txt”

# parameters of the SW algorithm and CNVfiles
p=3; swt_del=5; swt_dup=5; dv=0.5;
sw_exec = paste(getLibPath(),”/CoNVex/exec/swa_lin64”,sep=””); # SW-Array execution binary
swa_folder = paste(output_folder,”/CNVcalls_p”,p,”_t”,t,”_new”,sep=””); # CNV calls will be stored in this
CNVfiles = paste(swa_folder,”/CoNVex_”,sample_ids,”_p”,p,”_tdel”,swt_del,”_tdup”,swt_dup,”_dv”,dv,”_.txt”,s

# These files are not required for this function, but REQUIRED LATER
BPfile =  ”/home/pv1/UK10K/MOPD_Breakpoints.txt”
centromere_regions_file = paste(getLibPath(),”/CoNVex/extdata/gaps_table_hg19.txt”,sep=””)
features_file = ”/home/pv1/UK10K/MOPD_Features.txt”

# Save all vectors in a fixed format
baminfoALL = SampleInfoPrep(sample_id=sample_ids, gender=gender, bamfiles=bamfiles, RDfiles=RDfiles, L2Rfil
```

SampleInfoPrepInteractive

Input Data Validation before Starting the Analysis

Description

This function does basic validation with the input and creates output folders/files for further CNV analysis. This is an important first step to start the analysis. The output file (sample info file) created by this function has columns/info in a defined format. Returns the same info as a data frame.

Usage

```
SampleInfoPrepInteractive(sample_ids,gender,bamfiles,RDfiles,\
L2Rfiles,GAMfiles,sample_info_file,output_folder,overwrite=FALSE)
```

Arguments

`sample_ids`	Vector of Sample IDs
`gender`	Vector of Sample gender
`regions_file`	Probe regions file
`bamfiles`	Vector of BAM files - rows should match sample_ids
`RDfiles`	Vector of output read depth files
`L2Rfiles`	Vector of file names in which uncorrected log2 ratio generated by SampleLogRatio will be stored - full path required
`GAMfiles`	Vector of file names in which corrected log2 ratio and MAD-weighted scores generated by GAMCorrection will be stored - full path required
`sample_info_file`	Output file name to store the all required sample information for further analysis

`output_folder` [optional] Output folder for read depth and other files

`overwrite` [optional; default=FALSE] By default, sample info file is not overwritten as a precaution. overwrite=TRUE overwrites it

Details

# Requires input info as shown in function options
# Creates output_folder if it doesn't exist
# Checks whether the BAM files exist
# Checks gender - only 'F', 'M' and 'U' are allowed
# Checks whether sample_info_file exists already - overwrite=TRUE (default: FALSE) overwrites it without warning
# Saves the sample_info_file in the specified output_folder with all required columns
# Output saved in sample_info_file is also returned as a data frame. You can keep it in an R interactive session for reference.

Note

This step is required for further analysis. Based on your expertise, you may create the sample_info_file yourself through other ways. You'll save a lot of time before the analysis by making sure all input and output files are right place by using this simple function.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

Agilent's ADM2 algorithm

See Also

CoNVex documentation

Examples

```
require(CoNVex)
baminfoRD = read.table("/path/to/SampleIDs_Gender_BAMfileLocations.txt", sep="\t")

# Required info
# regions - string / given with CoNVex package
regions_file = paste(getLibPath(),"/CoNVex/extdata/SureSelect_50Mb.txt",sep="");
sample_ids = as.character(baminfoRD[,3]);
bamfiles = as.character(baminfoRD[,4]);
gender = as.character(baminfoRD[,2]);
gender[gender=='Male'] = 'M';
gender[gender=='Female'] = 'F';
output_folder = "/home/pv1/UK10K/";
RDfiles = paste(output_folder,"/ProbeRD_",sample_ids,".dat",sep="")
L2Rfiles = paste(output_folder,"/L2R/L2R_",sample_ids,".dat",sep="")
GAMfiles = paste(output_folder,"/L2R/GAM_",sample_ids,".dat",sep="")
sample_info_file = "/home/pv1/UK10K/UK10K_SampleInfo.txt"
```

```
# parameters of the SW algorithm and CNVfiles
p=3; swt_del=5; swt_dup=5; dv=0.5;
sw_exec = paste(getLibPath(),"/CoNVex/exec/swa_lin64",sep=""); # SW-Array execution binary
swa_folder = paste(output_folder,"/CNVcalls_p",p,"_t",t,"_new",sep=""); # CNV calls will be stored in this
CNVfiles = paste(swa_folder,"/CoNVex_",sample_ids,"_p",p,"_tdel",swt_del,"_tdup",swt_dup,"_dv",dv,"_.txt",s

# These files are not required for this function, but REQUIRED LATER
BPfile =  "/home/pv1/UK10K/MOPD_Breakpoints.txt"
centromere_regions_file = paste(getLibPath(),"/CoNVex/extdata/gaps_table_hg19.txt",sep="")
features_file = "/home/pv1/UK10K/MOPD_Features.txt"

# Save all vectors in a fixed format
baminfoALL = SampleInfoPrepInteractive(sample_id=sample_ids, gender=gender, bamfiles=bamfiles, RDfiles=RDfi
```

SampleLogRatio *SampleLogRatio - log2 ratio for each sample*

Description

This calculates log2 ratio for each sample using a median reference estimated from all samples

Usage

```
SampleLogRatio(RDfiles, sample_ids, sample_gender, regions_file, chrX=0)
```

Arguments

`RDfiles`	Vector of output read depth files
`sample_ids`	Vector of Sample IDs
`sample_gender`	Gender of the samples - 'M', 'F' and 'U' are allowed
`regions_file`	Probe regions file
`chrX`	[optional] cX=0 (default) excludes Chr X, cX=1 includes it

Details

SampleLogRatio function calculates log2 ratio (of depth/sample-median) for each probe region. chrX=0 (default) excludes Chr X. chrX=1 includes it. If Chr X=1, all samples must have 'M' or 'F' (male or female) so that Chr X is compared within males and females separately. While using chrX=0, you can use 'U' (unknown) for the sample_gender. ChrY is fully excluded as it has very few probe regions, and the number of regions are significantly decreasing in the newer versions of off-the-shelf exome libraries.

Note

SampleLogRatioCall batch script calls it for all samples. SampleLogRatioCallCommands generates Unix commands for batch execution.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleLogRatioCallCommands, SampleLogRatioCall

Examples

```
# Check examples of SampleLogRatioCallCommands
```

SampleLogRatioCallCommands *Unix Command Generator for using SampleLogRatioCall*

Description

Generates Unix command(s) (e.g., one per batch) to calculate Sample Log2 Ratio

Usage

```
SampleLogRatioCallCommands(sample_info_file,regions_file,\
features_file,cX=0,Rbatch_folder="")
```

Arguments

`sample_info_file`
: Tab-separated file auto-generated by SamplePrepInfo() function - required for the whole analysis

`regions_file`
: Probe regions file

`features_file`
: A subset of probe regions file with rowcount, Chr regions, GC, deltaG and Tm as columns

`cX`
: [optional] cX=0 (default) excludes Chr X, cX=1 includes it

`Rbatch_folder`
: [optional] Rbatch folder bundled with CoNVex for batch execution. Rbatch_folder=” (default, works mostly) assumes that it’s in R’s library folder. You may copy it to somewhere else and modify this explicitly

Details

SampleLogRatio function calculates log2 ratio (of depth/sample-median) for each probe region. SampleLogRatioCall calls it for all samples. SampleLogRatioCallCommands (this function) generates Unix commands for batch execution.

Note

This can be easily used together with batch queuing (e.g., bsub) programs. This function usually generates a single Unix command for all samples in the analysis batch.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleLogRatio, SampleLogRatioCall

Examples

```
> slr_command = SampleLogRatioCallCommands(sample_info_file,regions_file,\
features_file,cX=0,Rbatch_folder="");
> slr_command[1]
[1] "R --vanilla --slave --args '/home/pv1/MOPD_SampleInfo.txt, \
/home/pv1/R/x86_64-linux/2.11/CoNVex/extdata/SureSelect_50Mb.txt, \
/home/pv1/UK10K/UK10K_Features.txt,1' < \
/home/pv1/R/x86_64-linux/2.11/CoNVex/Rbatch/SampleLogRatioCall.R"
```

`SampleLogRatioV2` *SampleLogRatioV2 - log2 ratio using a correlated subset of samples*

Description

This SampleLogRatioV2 (version=2) calculates log2 ratio using a median reference estimated from within a highly correlated subset of samples (as opposed to all samples in the analysis batch - like SampleLogRatio). This is beneficial if you use a heterogenous mixture of samples in the analysis batch. e.g., using same exome library, but the upstream data analysis (assembly, filtering, etc.) differs.

Usage

```
SampleLogRatioV2(RDfiles, sample_ids, sample_gender, regions_file, chrX=0, min_samples=25, RPKM=
```

Arguments

`RDfiles` Vector of output read depth files

`sample_ids` Vector of Sample IDs

`sample_gender` Gender of the samples - 'M', 'F' and 'U' are allowed

`regions_file` Probe regions file

`chrX` [optional] cX=0 (default) excludes Chr X, cX=1 includes it

`min_samples=25` Minimum #samples expected in the subset. If chrX=1, both #males and #females should independently have this many number of samples

`RPKM=0` [optional] RPKM=1 Use RPKM (aka FPKM) to correlate the samples. RPKM=0 (default) Use read depth instead.

Details

SampleLogRatioV2 function calculates log2 ratio (of depth/sample-median) for each probe region. chrX=0 (default) excludes Chr X. chrX=1 includes it. If ChrX=1, all samples must have 'M' or 'F' (male or female) so that Chr X is compared within males and females separately. While using chrX=0, you can use 'U' (unknown) for the sample_gender. ChrY is fully excluded as it has very few probe regions, and the number of regions are significantly decreasing in the newer versions of off-the-shelf exome libraries. SampleLogRatioV2 differs from SampleLogRatio by estimating median reference only from a correlated subset of samples.

Note

SampleLogRatioCall batch script calls it for all samples (use 'version=2' to use SampleLogRatioV2). SampleLogRatioCallCommands generates Unix commands for batch execution. Don't use this function if you sample size is too small.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleLogRatioCallCommands, SampleLogRatioCall

Examples

```
# Check examples of SampleLogRatioCallCommands
```

`SampleLogRatioV3` *SampleLogRatioV3 - Optimised version of SampleLogRatioV2 using Unix's cut command (depends *nix style environment)*

Description

SampleLogRatioV3 (version=3) is similar to SampleLogRatioV2. i.e. It calculates log2 ratio using a median reference estimated from within a highly correlated subset of samples (as opposed to all samples in the analysis batch - like SampleLogRatio). This is beneficial if you use a heterogenous mixture of samples in the analysis batch. e.g., using same exome library, but the upstream data analysis (assembly, filtering, etc.) differs.

Usage

```
SampleLogRatioV3(RDfiles, sample_ids, sample_gender, regions_file, chrX=0, min_samples=25, RPKM=
```

Arguments

`RDfiles` Vector of output read depth files

`sample_ids` Vector of Sample IDs

`sample_gender` Gender of the samples - 'M', 'F' and 'U' are allowed

`regions_file` Probe regions file

`chrX` [optional] cX=0 (default) excludes Chr X, cX=1 includes it

`min_samples=25` Minimum #samples expected in the subset. If chrX=1, both #males and #females should independently have this many number of samples

`RPKM=0` [optional] RPKM=1 Use RPKM (aka FPKM) to correlate the samples. RPKM=0 (default) Use read depth instead.

Details

SampleLogRatioV3 function calculates log2 ratio (of depth/sample-median) for each probe region. chrX=0 (default) excludes Chr X. chrX=1 includes it. If ChrX=1, all samples must have 'M' or 'F' (male or female) so that Chr X is compared within males and females separately. While using chrX=0, you can use 'U' (unknown) for the sample_gender. ChrY is fully excluded as it has very few probe regions, and the number of regions are significantly decreasing in the newer versions of off-the-shelf exome libraries. SampleLogRatioV3 differs from SampleLogRatio by estimating median reference only from a correlated subset of samples. This function depends on Unix-like environment and cut command that comes pre-installed in those systems (including OSX).

Note

SampleLogRatioCall batch script calls it for all samples (use 'version=3' to use SampleLogRatioV3). SampleLogRatioCallCommands generates Unix commands for batch execution. Don't use this function if you sample size is too small.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleLogRatioCallCommands, SampleLogRatioCall

Examples

```
# Check examples of SampleLogRatioCallCommands
```

SampleMADsUnix *Estimate median absolute deviation of log2 ratio*

Description

SampleMADsUnix calculates MAD (median absolute deviation) of the corrected log2 ratio for each sample. This is used as a measure of noise for each sample and used for plotting and visualisation.

Usage

```
SampleMADsUnix(sample_ids, GAMfiles, regions_file)
```

Arguments

`sample_ids`	Vector of Sample IDs
`GAMfiles`	GAM file names - output files of GAMCorrection
`regions_file`	Probe regions file

Details

SampleMADsUnix uses GAMfiles to get the corrected log2 ratio (4th column) and calculates MAD.

Note

This can be used together with SampleMeans() function that calculates mean depth to plot Sample Means vs. MADs plots. This would explain sample noise in the analysis batch. This function can be called using SampleMeansMads bundled in the Rbatch folder. (MAD = Median Absolute Deviation)

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleMeans, SampleMADsUnixMads

Examples

```
> SampleMADsUnix(sample_ids=sids, GAMfiles=GAMcorrectionFiles,regions_file="path/to/regions_file")
```

SampleMeans *Estimate mean depth across probe regions*

Description

SampleMeans calculates mean depth across probe regions within each sample in the analysis batch. It returns a vector (equal to the length of 'sample_ids') of mean depths. This can be used for further plotting and visualisation.

Usage

```
SampleMeans(sample_ids, RDfiles, regions_file)
```

Arguments

sample_ids	Vector of Sample IDs
RDfiles	Vector of output read depth files
regions_file	Probe regions file

Details

SampleMeans uses RDfiles (read depth files) generated using the java ReadDepth program for each sample. It simply calculates the mean of the read depths of probe regions within the sample.

Note

This can be used together with SampleMADsUnix() function that calculates MAD(log2 ratio) to plot Sample Means vs. MADs plots. This would explain sample noise in the analysis batch. This function can be called using SampleMeansMads bundled in the Rbatch folder. (MAD = Median Absolute Deviation)

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleMADsUnix, SampleMeansMads

Examples

```
> SampleMeans(sample_ids=sids, RDfiles=ReadDepthFiles,regions_file="path/to/regions_file")
```

`SampleMedian` *SampleMedian - Median Depth across Samples*

Description

This function calculates median depth of each probe region across samples and returns them in two columns for males and females separately. Depth of autosomal probe regions are calculated across all samples (so, these are exactly same in two columns for ease of use), while those of chromosome X is calculated within males and females separately. If ChrX=0, only autosomal probe regions' medians are returned. Returned data frame includes the following columns: rowcount, chr, start, end, GC, dG, Tm, Median_Male, Median_Female

Usage

```
SampleMedian(RDfiles, sample_ids, sample_gender, regions_file, chrX=0)
```

Arguments

`RDfiles`	Vector of output read depth files
`sample_ids`	Vector of Sample IDs
`sample_gender`	Gender of the samples - 'M', 'F' and 'U' are allowed
`regions_file`	Probe regions file
`chrX`	[optional] cX=0 (default) excludes Chr X, cX=1 includes it

Details

SampleMedian function calculates median depth of each probe region across samples. chrX=0 (default) excludes Chr X. chrX=1 includes it. If Chr X=1, all samples must have 'M' or 'F' (male or female) so that Chr X is compared within males and females separately. While using chrX=0, you can use 'U' (unknown) for the sample_gender. ChrY is fully excluded as it has very few probe regions, and the number of regions are significantly decreasing in the newer versions of off-the-shelf exome libraries.

Note

SampleMedian is not required for CNV detection, as it's also a part of SampleLogRatio internally. Batch scripts are not available for this function.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

See Also

SampleMedianCallCommands SampleMedianCall

Examples

```
# sample_median = SampleMedian(RDfiles, sample_ids, sample_gender, regions_file, chrX=0)
```

`getClassPath` *Helps you setup the $CLASSPATH environment variable*

Description

Automatically searches for jar files in the installation path and shows $CLASSPATH environment variable settings.

Usage

```
getClassPath(convex_path="")
```

Arguments

`convex_path`	CoNVex R package installation path (default: convex_path = "" - this tries to automatically retrieve it from your default R lib path) .

Details

getClassPath() helps you run Java programs bundled with CoNVex. Please use convex_path = "/path/to/ConvexRpackage" explicitly, if getLibPath() does not find the CoNVex installation path!

Note

getClassPath() internally uses getLibPath() which iteratively searches through all folders in .libPaths() for a functional CoNVex installation, assuming you installed CoNVex in one of these locations.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

R Library path

See Also

.libPaths(), getLibPath()

Examples

```
> getClassPath()
```

`getLibPath` *Get CoNVex installation Path*

Description

Search and return CoNVex installation path (R library path) using this function.

Usage

```
getLibPath()
```

Details

getLibPath() iteratively searches through all folders in .libPaths() for a functional CoNVex installation, assuming you installed CoNVex in one of these locations. If it fails, it returns 'NA'. Please use convex_path = "/path/to/ConvexRpackage" explicitly for getClassPath() or other functions.

Note

This path can be used as CoNVex's installation folder for locating bundled external data and binaries. If folders and files are used from a custom location, this function is not required.

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

R Library path

See Also

.libPaths(), getClassPath()

Examples

```
> convex_path = getLibPath()
> print(convex_path)
```

`getMADMLR` *MAD of mean log2 ratio or error-weighted score of a CNV region*

Description

After CNV detection, MAD is calculated from mean log2 ratio (across probe regions) and mean error-weighted scores of a CNV region. This is used to distinguish rare and common CNVs and potential false positives.

Usage

```
getMADMLR(Fd_rep,aX,aXscore)
```

Arguments

`Fd_rep`	Data frame containing CNV regions with the following columns: chr, start, end, num_probes, convex_score, cnv_type, sample_id
`aX`	Data frame containing log2 ratio: rowcount, chr, start, end, GC, dG, Tm, [log2_ratio of all samples - one sample per column]
`aXscore`	Data frame containing MAD-weighted scores: rowcount, chr, start, end, GC, dG, Tm, [MAD-weighted scores of all samples - one sample per column].

Details

getMADMLR returns following columns: mean_ratio_sample, mad_mean_ratio, mad_mean_admscore for each CNV.

Note

MAD-weighted scores are also called ADM scores or error-weighted scores in some places

Author(s)

Parthiban Vijayarangakannan
Wellcome Trust Sanger Institute
Cambridge, UK

References

R Library path

See Also

SWCNV

Examples

```
> convex_path = getMADMLR()
```

Index