PeakRescue pipeline: steps

A. PeakRescue: extract unique reads using HTSeq and disambiguate/rescue ambiguously mapped reads.

Please download the HTSeq package hosted at https://github.com/rnaseq/peakrescue, which contains the original version of count.py (as of HTSeq version ***) and the extended version of count.py (*i.e.* count_peakRescue.py) which implements the disambiguation and rescue steps of peakrescue pipeline.

The input BAM file must be produced with a splice-aware aligner, must contain "NH:i:" tags (see SAM format specifications) and should be sorted by read names prior to running the peakRescue pipeline.

1. Extract the non-ambiguous uniquely mapped reads

Run HTSeq on the input BAM file with the following mandatory options: --*samout* as this permits to generate an output SAM file that is used in the next step of the pipeline; and the two additional options: --*type*=exon and --*idattr*=gene_id, to specify that peakRescue performs read summarization and rescue of ambiguous reads at the gene level.

Run as follows:

```
samtools view accepted_hits.bam | \
python \
count.py \
--mode=union \
--stranded=no \
--samout=accepted_hits_htseq.sam \
--type=exon \
--idattr=gene_id \
- \
genome.gff \
>htseq_count.out
```

Output files:
accepted_hits_htseq.sam :
SAM output file containing reads with an added *XF:Z* tag showing status of a given read in one of the three categories:
XF:Z:<gene_name> - read assigned to a single gene.
XF:Z:ambiguous[<gene_name_1>...<gene_name_n>] – Uniquely mapped ambiguous read overlapping more than one gene.
XF:Z:alignment_not_unique – multi mapped reads mapping at more than one location.

htseq_count.out: Tab separated file containing per gene fragment count data for: Unique, Ambiguous [multi-mapped + ambiguous unique] and the total read count.

2. Disambiguate ambiguous unique reads and store the remaining ambiguously mapped reads (unique and multimapped)

The reads flagged with either " *XF:Z:ambiguous*" or "*XF:Z:alignment_not_unique*" in the SAM file (output generated in the previous step –see A.1 section) are used as input to the extended version of the HTSeq count.py script (count_peakRescue.py).

Run as follows:

```
grep -P "ambiguous|alignment_not_unique" accepeted_hits_htseq.sam | \
python \
count_peakRescue.py \
--mode=union \
--stranded=no \
--samout=disambiguated.sam \
--type=exon \
--idattr=gene_id \
- \
genome.gff \
multimapped_readname_gene_name.out \
ambiguous_readname_gene_name.out \
>rescued_count.out \
```

Output files:

disambiguated.sam: SAM file containing unique disambiguated reads assigned to a single gene labelled in the *XF:Z:<gene_name>* tag.

multimapped_readname_gene_name.out: This is an intermediate tab separated file, which provides read names-to-gene names mappings – with two columns:

Column 1: Read name

Column 2: A list of gene names (Ensembl IDs) on which the given multimapped read maps.

ambiguous_readname_gene_name.out: This is an intermediate tab separated file, which provides read names-to-gene names mappings – with two columns:

Column 1: Read name

Column 2: A list of gene names (Ensembl IDs) on which the given ambiguous read overlaps.

rescued_count.out: Tab separated file containing per gene fragment count data with the following columns:

Column 1: Ensembl ID

Column 2: Unique disambiguated,

Column 3: Ambiguous [multimapped + ambiguous unique]

Column 4: The total read count for the given gene. Fraction of non-unique read will be assigned to underlying genes listed in *multimapped_readname_gene_name.out* and *ambiguous_readname_gene_name.out* files based on expression proxy defined by maximum unique peak[*Gene expression proxy* in Methods].

B. Pre-process GTF file:

Run as follows:

```
processGTF.pl -gtf genome.gtf.gz
```

The input GTF file may be downloaded from Ensembl and should contain transcripts and gene information for a given species.
Process the input GTF file in order to get the "global transcript" for each gene described in the input GTF (*i.e.* merged exons over all transcripts of a gene). [see Fig.1 A – paper under submission]. This GTF pre-processing steps creates the following files to be used in the later steps of PeakRescue:

1. *global_transcript.bed*: Exon interval for a single transcript created after merging all known transcripts for a gene.
2. *unique_regions.bed*: unique and non overlapping intervals per gene.
3. *geneboundaries.bed*: start and stop of a global transcript defined for respective genes.
4. *global_transcript_gene_length.tab:* gene length based on all non overlapping exons listed in global_transcript.bed file.
5. *unique_segment_gene_length.tab*: gene length based on unique regions listed in unique_regions.bed file.

C. Calculate peak [see methods]

Input BAM file is a merged BAM containing the non-ambiguous uniquely mapped and the disambiguated uniquely mapped reads.

```
getPeak.pl -bed geneboundaries.bed -bam merged.bam -g genome.fa
```

Output file:

peak.tab: Tab separated file containing gene name and peak value

D. Run probabilistic assignment of ambiguous reads to underlying genes.

Run as follows:

```
python peakRescue_readToGeneAssignment.py \
-p PEAK_FILENAME \
-m MAPPINGS_READS2GENES_FILENAME \
-l GENE_LENGTH_FILENAME \
-t READTYPE
```

Output file:

results_peakrescue_readtype_[RT]_all_genes.out is produced with the tag RT replaced by the read type *e.g.* "ambiguous_unique" or "multimappers" used in the command line (-t option).
Tab separated output file contains the following columns:
Column 1: Ensembl ID
Column 2: Proportion of [readType] assigned to gene

E. **Combine results (count data):**
non-ambiguous unique, disambiguated unique and ambiguous rescued.
@todo: command line to be updated with latest update.

peakRescueFinalCount.out : This file contains per gene count values that user can input to any differential analysis algorithms. FinalCount column contains sum of Uniqu*e,* Unique disambiguated and peakRescue contributions (*i.e.* sum of all rescued ambiguous reads' contributions).
In addition to the FinalCount column this file also contains the following additional columns:
Gene: gene name as specified in original input gtf file
uniqueCount: Unique read count
nonUniqueCount: Non unique read count includes only multi-mapped reads
disambUniqueCount: Disambiguated unique read count
allNonUniqueCount: Non unique read count includes both multi-mapped and ambiguous reads
peakContributionCount: read contribution proportion based on relative peak value
finalCount: contains sum of uniqueCount*,* disambUniqueCount and peakContributionCount