HomeoRoq: Homeolog Ratio and Quantification

HomeoRoq is a pipeline to detect statistically significant changes in the homeolog-specific expression ratios under different conditions using RNA-seq data. The pipeline consists of two parts: (A) quantification of homeolog-specific expression levels and (B) statistical test for changes in homeolog-specific expression ratios under different conditions. These programs are provided as a supplemental material of the paper "Genome-wide quantification of homeolog expression ratio revealed non-stochastic gene regulation in synthetic allopolyploid *Arabidopsis*," written by Satoru Akama, Rie Shimizu-Inatsugi, Kentaro K. Shimizu and Jun Sese.

Programs

(A) Quantification of homeolog-specific expression levels

- read_classify.py

(B) Statistical test for changes in homeolog-specific expression ratios under different conditions

- calcpval_one.R, calcpval.R, calcpval_mean.R

Requirements (external programs)

- STAR (ultrafast universal RNA-seq aligner) 2.3.0 or higher
    - http://code.google.com/p/rna-star/
    - Dobin, A. et al. (2013) STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, **29**, 15-21.
- Samtools 0.1.19 or higher
- R-package, locfit 1.5-9.1 or higher (used in "calcpval_one.R")
- R-package, doMC 1.3.0 or higher (used in "calcpval_one.R")

Procedure

In this procedure, we assume the expression analysis of allopolyploid with 3 replicates originated from two parental species under two different conditions as a sample case. The quantification of homeolog-specific expression levels can be performed without multiple replicates.

Input files (sample case)

- Parental genomic sequences in FASTA format: [Parent1_genome.fas] and [Parent2_genome.fas]
- RNA-seq reads (paired-end), 6 libraries:
    (Control 1) [Allo_Ctrl_1_R1.fastq], [Allo_Ctrl_1_R2.fastq],
    (Control 2) [Allo_Ctrl_2_R1.fastq], [Allo_Ctrl_2_R2.fastq],
    (Control 3) [Allo_Ctrl_3_R1.fastq], [Allo_Ctrl_3_R2.fastq],
    (Stress 1) [Allo_Stress_1_R1.fastq], [Allo_Stress_1_R2.fastq],
    (Stress 2) [Allo_Stress_2_R1.fastq], [Allo_Stress_2_R2.fastq],
    (Stress 3) [Allo_Stress_3_R1.fastq], [Allo_Stress_3_R2.fastq].

(A) Quantification of homeolog-specific expression levels

Step 0: Prepare sequences of parental genomes

Genomic sequences of both of the parents are required: [Parent1_genome.fas] and [Parent2_genome.fas].

Step 1: Align RNA-seq reads to each parental genome separately with STAR

Step 1a: Make indexes of both parental genome sequences

```
$ star --runMode genomeGenerate --genomeDir [Parent 1 dir] --genomeFastaFiles [Parent1_genome.fas] --runThreadN [Arbitrary value]
--limitGenomeGenerateRAM [Arbitrary value]
$ star --runMode genomeGenerate --genomeDir [Parent 2 dir] --genomeFastaFiles [Parent2_genome.fas] --runThreadN [Arbitrary value]
--limitGenomeGenerateRAM [Arbitrary value]
```

Step 1b: Align RNA-seq reads to each parental genome separately

```
$ star --genomeDir [Parent 1 dir] --readFilesIn [Allo_Ctrl_1_R1.fastq] [Allo_Ctrl_1_R2.fastq] --runThreadN [arbitrary value] --genomeLoad
NoSharedMemory
$ star --genomeDir [Parent 2 dir] --readFilesIn [Allo_Ctrl_1_R1.fastq] [Allo_Ctrl_1_R2.fastq] --runThreadN [arbitrary value] --genomeLoad
NoSharedMemory
```

The same alignment procedure is performed for the other libraries (Ctrl_2, Ctrl_3, Stress_1, Stress_2, Stress_3). Here, the 12 SAM files are obtained:

[Allo_Ctrl_1_on_parent1.sam], [Allo_Ctrl_1_on_parent2.sam],
[Allo_Ctrl_2_on_parent1.sam], [Allo_Ctrl_2_on_parent2.sam],
[Allo_Ctrl_3_on_parent1.sam], [Allo_Ctrl_3_on_parent2.sam],
[Allo_Stress_1_on_parent1.sam], [Allo_Stress_1_on_parent2.sam],
[Allo_Stress_2_on_parent1.sam], [Allo_Stress_2_on_parent2.sam],
[Allo_Stress_3_on_parent1.sam], [Allo_Stress_3_on_parent2.sam].

Step 2: Classify genomic origin of reads

Step 2a: Prepare header file and sort SAM file

```
$ grep "^@" [Allo_Ctrl_1_on_parent1.sam] > [Allo_Ctrl_1_on_parent1.header]
$ grep "^@" [Allo_Ctrl_1_on_parent2.sam] > [Allo_Ctrl_1_on_parent2.header]
$ grep -v "^@" [Allo_Ctrl_1_on_parent1.sam] | sort -k1 > [Allo_Ctrl_1_on_parent1.sort.sam]
$ grep -v "^@" [Allo_Ctrl_1_on_parent2.sam] | sort -k1 > [Allo_Ctrl_1_on_parent2.sort.sam]
```

The same procedure is performed for the other libraries.

Step 2b: Classify aligned read into 5 categories

The python script “read_classify.py” classifies the aliened reads into 5 categories with a specific suffix (xxx_orig, xxx_other, xxx_common, xxx_unmapped, xxx_ambi). Each category corresponds to a read mapped on H-genome in

Figure 1B (xxx_orig <=> H-origin, xxx_other <=> L-origin, xxx_common <=> Unclassified, xxx_unmapped <=> Discard). The execution is as follows:

```
$ python read_classify.py [Allo_Ctrl_1_on_parent1.sort (filename of SAM file on parent 1 without .sam)] [Allo_Ctrl_1_on_parent2.sort (filename of SAM file on parent 2 without .sam)] > [classify.log]
```

The same procedure is performed for the other libraries. In this case, the python script outputs 10 bam files (5 categories x 2 parental genomes):
[Allo_Ctrl_1_on_parent1_orig.sam], [Allo_Ctrl_1_on_parent1_other.sam], [Allo_Ctrl_1_on_parent1_common.sam], [Allo_Ctrl_1_on_parent1_unmapped.sam], [Allo_Ctrl_1_on_parent1_ambi.sam],
[Allo_Ctrl_1_on_parent2_orig.sam], [Allo_Ctrl_1_on_parent2_other.sam], [Allo_Ctrl_1_on_parent2_common.sam] , [Allo_Ctrl_1_on_parent2_unmapped.sam], [Allo_Ctrl_1_on_parent2_ambi.sam].

Step 3: Count reads on gene region

The classified reads are counted on the gene region for expression analysis, and then the count data on the same parental genome are merged into a file. An example of the column names in [count.xls] file is as follows:

```
> colnames(count)
[1] "gene"            "Ctrl_1_hal"   "Ctrl_1_lyr"   "Ctrl_1_both" "Ctrl_2_hal"
[6] "Ctrl_2_lyr"   "Ctrl_2_both" "Ctrl_3_hal"   "Ctrl_3_lyr"   "Ctrl_3_both"
[11] "Cold_1_hal"   "Cold_1_lyr"   "Cold_1_both" "Cold_2_hal"   "Cold_2_lyr"
[16] "Cold_2_both" "Cold_3_hal"   "Cold_3_lyr"   "Cold_3_both"
```

Step 4: Normalize expression level by RPKM

The RPKM normalization is performed with the count data and the cDNA lengths. An example of the column names in [rpkm.xls] file is as follows:

```
> colnames(count)
[1] "gene"                    "RPKM_hal.ctrl_rep1"   "RPKM_lyr.ctrl_rep1"
[4] "RPKM_both.ctrl_rep1" "RPKM_hal.ctrl_rep2"   "RPKM_lyr.ctrl_rep2"
[7] "RPKM_both.ctrl_rep2" "RPKM_hal.ctrl_rep3"   "RPKM_lyr.ctrl_rep3"
[10] "RPKM_both.ctrl_rep3" "RPKM_hal.cold_rep1"   "RPKM_lyr.cold_rep1"
[13] "RPKM_both.cold_rep1" "RPKM_hal.cold_rep2"   "RPKM_lyr.cold_rep2"
[16] "RPKM_both.cold_rep2" "RPKM_hal.cold_rep3"   "RPKM_lyr.cold_rep3"
[19] "RPKM_both.cold_rep3"
```

(B) Statistical test for changes in homeolog-specific expression ratios under different conditions

To identify homeolog pairs whose expression ratio are statistically significantly changed, permutation based test is performed. P-values are calculated multiple times with the R-script “calcpval_one.R”, and then the mean value of the obtained p-values is calculated with “calcpval_mean.R”.

“calcpval_one.R” requires a RPKM data and a label information. For example, when the RPKM data mentioned in Step 4 of (A) is used, the label information file should contain the following content:

```
"RPKM_hal.ctrl_rep1","RPKM_lyr.ctrl_rep1","RPKM_hal.cold_rep1","RPKM_lyr.cold_rep1"
"RPKM_hal.ctrl_rep2","RPKM_lyr.ctrl_rep2","RPKM_hal.cold_rep2","RPKM_lyr.cold_rep2"
"RPKM_hal.ctrl_rep3","RPKM_lyr.ctrl_rep3","RPKM_hal.cold_rep3","RPKM_lyr.cold_rep3"
```

The columns represent the four RPKM values of parent 1-origin at control condition, parent 2-origin at control condition, parent 1-origin at stress condition, and parent 2-origin at stress condition, and the rows represent the replicates. Each label has to correspond to the column names in the RPKM file and be written in comma separated value (CSV) format. “calcpval.R” has to be located in a working directory (“calcpval_one.R” uses “calcpval.R” in the calculation). “calcpval_one.R” is executed as follows:

```
$ R --vanilla --slave --args [pval_run_1.xls (name of output file)] [RPKM.xls] [label.txt] < calcpval_one.R
$ R --vanilla --slave --args [pval_run_2.xls (name of output file)] [RPKM.xls] [label.txt] < calcpval_one.R
…
$ R --vanilla --slave --args [pval_run_N.xls (name of output file)] [RPKM.xls] [label.txt] < calcpval_one.R
```

Then, the mean value of the p-values is calculated with the outputs files ([pval_run_1.xls], [pval_run_2.xls],…, [pval_run_N.xls]) and “calcpval_mean.R”. The function “pvalmean” in “calcpval_mean.R” requires two inputs: a sequence of the input file names and a name of the output file.

```
> source("calcpval_mean.R")
> head(filenames)
[1] "pval_run_1.xls" "pval_run_2.xls" "pval_run_3.xls" "pval_run_4.xls"
…
> pvalmean(filenames, "pval_mean.xls")
```

The column names in the output are as follows:

```
> colnames(pval_mean)
[1] "gene"          "pval"          "padj"          "ctrlFirst"    "ctrlSecond"
[6] "objFirst"     "objSecond"    "ctrlRatio"    "objRatio"     "ratioSD"
[11] "fisher"       "fisher.adj" "chisqr"       "chisqr.adj"
```

The meanings of each column are as follows:

[1] gene: Gene ID [2] pval: P-value by proposed test

[3] padj: Adjusted p-value of proposed test with Benjamini-Hochberg (BH) multiple test correction

[4] ctrlFirst: Sum of parent 1-origin expression at control condition (e.g. RPKM of H-homeolog at Ctrl)

[5] ctrlSecond: Sum of parent 2-origin expression at control condition (e.g. RPKM of L-homeolog at Ctrl)

[6] objFirst: Sum of parent 1-origin expression at stress condition (e.g. RPKM of H-homeolog at Stress)

[7] objSecond: Sum of parent 2-origin expression at stress condition (e.g. RPKM of L-homeolog at Stress)

[8] ctrlRatio: Ratio of parent 1 at control condition (e.g. Ratio of H-homeolog at Ctrl)

[9] objRatio: Ratio of parent 1 at stress condition (e.g. Ratio of H-homeolog at Stress)

[10] ratioSD: Standard deviation of all of the Ratios (e.g. 6 libraries)

[11] fisher: P-value by Fisher’s exact test [12] fisher.adj: Adjusted p-value of Fisher’s exact test with BH

[13] chisqr: P-value by chi-square test [14] chisqr.adj: Adjusted p-value of chi-square test with BH