Manual of Target Sequencing Data Processing

General Usage

Resource File

All the data processing scripts have been integrated into one R script: TGPipeline.R. Before running the script, users need to set up the resource file first. The resource file, res.txt, is in the same directory as TGPipeline.R. It indicates the software and location used in the pipeline. So the pipeline knows where to find these softwares. An example res.txt file is as following:

```
fastx_trimmer	~/soft/fastx_toolkit/fastx_toolkit-0.0.13.2/bin/fastx_trimmer
bwa	~/soft/bwa-0.7.13/bwa
samtools		~/soft/samtools-1.3/samtools
bedtools		~/soft/bedtools2/bin/bedtools
picard	~/soft/picard-tools-2.4.1/picard.jar
gatk	~/soft/GenomeAnalysisTK.jar
annovar		~/soft/annovar/
```

The resource file lists all the software and resource files needed in the pipeline. The first 7 are the commonly used software. The last 2 are the reference file and corresponding dbSNP information file. Users need to adapt this file according to the settings in their own computer.

Script for Batch Server

After setting the resource file, users need to go to the pipeline source file directory and use Rscript to run TGpipeline.R directly. For a batch server, an example script for qsub is provided below:

```
#!/bin/bash
#PBS -q default
#PBS -l nodes=1:ppn=8,mem=32gb
#PBS -l walltime=48:00:00
#PBS -m abe -M youremail@yale.edu

module load Langs/R
module load Langs/Perl

source ~/.Renviron

cd / TGPipeline/dir/

Rscript TGPipeline.R -fq /fastq/file/dir/ -o /output/dir/ -ref /reference/sequence.fa -dbsnp
/dbsnp/variants.vcf -build hg38 -trim 20 -ampbed amplicon.bed -seqbed TargetSequence.bed
```

The first 5 lines are basic settings for batches. The script is used for batch server Ruddle in Yale, which uses module method to load software. The two "module load" lines here are used to load R and Perl environment for the pipeline. About how to load R and Perl, check the computer administrator for help. In the pipeline, we need two R packages "parallel" and "ggplot2". "ggplot2" is not installed by default in Ruddle but installed locally. We use "scource ~/.Renviron" to set local library path for R to load "ggplot2". If "ggplot2" is already in R, remove this line. Then go to the source file directory ("cd /TGPipelien/dir/") and run the script.

Parameters of The Pipeline

As shown in the last line of the script above, users need to set up some parameters for the pipeline.

fq:	fastq file directory
o:	output directory
ref:	reference genome file

dbsnp: dbsnp variants information

trim: number of base pair to trim from the reads (trim the primers)

ampbed: bed file for each amplicon. There might be overlap between each amplicon for large exons. For example:

chr1	*200*	*500*	*exon1*
chr1	*1000*	*1400*	*exon2a*
chr1	*1300*	*1650*	*exon2b*
chr1	*1600*	*2010*	*exon2c*

seqbed: bed file for target sequence. The union of region indicated by amplicon bed file. It is the target region for sequencing. The target sequence file for the amplicon bed file above is:

chr1	*200*	*500*	*exon1*
chr1	*1000*	*2010*	*exon2*

build: build version used for annovar. The default is hg38. **Make sure that the parameters of ref, dbsnp in resource file and build version set here are consistent with each other.**

Output

In the output directory set by '-o', the pipeline creates several subdirectories.

trimmed:	fastq files after trimming
alignment:	bam and sam files after alignment
qc:	quality control files and figures
gatk:	variants called by gatk
freebayes:	variants called by freebayes
annovar:	annotated variants after comparing the difference between gatk and freebayes

Steps and Sub-functions in Target Sequencing Pipeline

Reads Trimming (trim.R)

In target sequencing, at the beginning of each read, there are primers. It is better to trim off the primers in case there is any variant in the primer region. fastx_trimmer is used here to trim the primers. The trimming length can be set by parameter '-trim'.

Function definition:

trimfastq <- function(filename, fastq.dir, out.dir, trim.length, res.info)

filename:	file name for fastq file
fastq.dir:	directory of fastq file
out.dir:	output directory for trimmed fastq file
trim.length:	number of bps to trim off
res.info:	resource information

Alignment (alignment.R)

In the pipeline, bwa is used here for reads alignment.

Function Definition:

alignment <- function(X, ref, fastq.dir, out.dir, res.info)

X:	string vector of sample name, paired fastq file name 1, and paired fastq file name 2
ref:	reference sequence
fastq.dir:	directory of fastq files
out.dir:	output directory of sam files
res.info:	resource information

Sort Reads, Create Bam File and Make Index (sortedBam.R)

In the pipeline, picard is used to sort the reads according to coordinate and them samtools is used to index the bam file.

Function Definition:

sortedbam <- function(sam.file, sam.dir, out.dir, res.info)

sam.file:	file name of sam file
sam.dir:	directory of sam file
out.dir:	output directory or sorted bam file
res.info:	resource information

Quality Control (QualityControl.R)

In this step, 3 tables and figures are generated to show the total number of reads, mapping quality and uniformity.

Function Definition:

basecoverage <- function(bam.dir, seq.bed, out.dir, sample.name, res.info)

bam.dir:	bam file directory
seq.bed:	bed file of target sequence region
out.dir:	output directory
sample.name:	name of sequenced samples
res.info:	resource information

mapsummary <- function(bam.dir, out.dir, sample.name, res.info)

bam.dir:	bam file directory
out.dir:	output directory
sample.name:	sample.name: name of sequenced samples

res.info:	resource information

uniformity <- function(bam.dir, amplicon.bed, out.dir, sample.name, res.info)

bam.dir:	bam file directory
amplicon.bed:	bed file of amplicon
out.dir:	output directory
sample.name:	sample.name: name of sequenced samples
res.info:	resource information

Variant Call with GATK (gatk.R)

Use GATK to call variants from bam files. GATK first create g.vcf file for each sample and then combine all g.vcf files together to one vcf file.

Function Definition:

Create g.vcf file for each bam file

gatk <- function(bamfile, ref, bam.dir, out.dir, res.info, dbsnp = NULL, bedfile = NULL)

bamfile:	bam file name
ref:	reference file
bam.dir:	bam file directory
out.dir:	output directory
res.info:	resource information
dbsnp:	dssnp file
bedfile:	bed file for target sequence region

parallelize the gatk function defined above and then combine the g.vcf files together.

gatkParallel <- function(bam.dir, ref, out.dir, res.info, dbsnp = NULL, bedfile = NULL)

bam.dir:	bam file directory
ref:	reference file
out.dir:	output directory
res.info:	resorce information
dbsnp:	dbsnp file
bedfile:	bed file for target seqeunce region

Variant Call with freebayes

Use freebayes to call variant with default of 20 DP at least.

Function Definition

freebayes <- function(bamfile, ref, bam.dir, out.dir, res.info, bedfile = NULL)

bamfile:	bam file name
ref:	reference file
bam.dir:	bam file directory
out.dir:	output directory
res.info:	resource information
bedfile:	bed file for target sequence region

ANNOVAR

Create annovar input file from vcf file generated by GATK and freebayes and then use annovar to annotate the variants. Before using ANNOVAR, make sure that the database files have been downloaded to humandb sub-directory. Check http://annovar.openbioinformatics.org/en/latest/user-guide/startup/ for details.

Function Definition

annovar <- function(gatk.vcf.dir, freebayes.vcf.dir, out.dir, res.info, out.label, builder,gatk.pattern=".vcf$", freebayes.pattern = ".vcf$")

gatk.vcf.dir:	directory for vcf files generated by gatk
freebayes.vcf.dir:	directory for vcf files generated by freebayes
out.dir:	output directory
res.info:	resource information
out.label:	label for output filename
builder:	annovar database builder version (hg18, hg19, hg38,…)
gatk.pattern:	pattern of vcf file name generated by gatk
freebayes.pattern:	pattern of vcf file name generated by freebayes