PING (Pushing Immunogenomics to the Next Generation) manual

The PING scripts were written by Paul Norman, Wesley Marin and Jill Hollenbach. If you use this program please cite the following paper:

Norman et al., Defining KIR and HLA Class I Genotypes at Highest Resolution via High-Throughput Sequencing, The American Journal of Human Genetics (2016), http://dx.doi.org/10.1016/j.ajhg.2016.06.023

Table of Contents

The PING pipeline

What is it?

The KIR region of the human genome varies in gene content. The genes encode molecules expressed by natural killer cells, which have important functions in human health. The PING pipeline consists of a series of R scripts that use Bowtie2, Samtools and MIRA to determine KIR gene presence, copy numbers and genotypes from high throughput sequencing data. The latest versions of these scripts support KIR gene presence and copy number determination for all KIR loci, and genotype calling for KIR2DL1, KIR2DL23, KIR2DL4, KIR2DL5, KIR2DS3, KIR2DS4, KIR2DS5, KIR2DP1, KIR3DL1, KIR3DS1, KIR3DL2 and KIR3DL3.

The PING pipeline is split between three scripts; PING_extractor, PING_gc_caller and PING_allele_caller. PING_extractor pulls relevant KIR reads from fastq files, PING_gc_caller determines gene presence and copy number, and PING_allele_caller determines KIR genotypes. The fourth script, PING_run_all, loads in the three previous scripts and runs through the entire pipeline using simplified parameters.

Downloading

The entire PING pipeline can be found at https://github.com/wesleymarin/PING

Dependencies

Linux (tested on Ubuntu 15.10 and CentOS 6.8)

MIRA (tested with version 4.0.2) -- link

Bowtie2 (tested with version 2.2.9) -- link

Samtools (tested with version 1.3.1) -- link

Bcftools (tested with version 1.3.1) -- link

R (tested with version 3.2.3)

R package dependencies:

- data.table
- ape
- stringr

PING_run_all.R

Overview

Once PING is downloaded and all dependencies installed, the entire pipeline can be run using the PING_run_all.R script. This script loads in the three other scripts and passes the relevant information from one script to the next.

Usage

```
ping_run_all( sample.location = "Sequences/", fastq.pattern.1 = "_1.fastq", fastq.pattern.2 = "_2.fastq",
        bowtie.threads = 4, results.directory = "" )
```

Arguments

sample.location	the location of the fastq files to analyze, a trailing slash "/" is important.
fastq.pattern.1	a character vector to identify the pair split pattern, normal and gzipped files are accepted. eg "_1.fastq" or "_1.fastq.gz".
fastq.pattern.2	a character vector to identify the pair split pattern, normal and gzipped files are accepted. eg "_2.fastq" or "_2.fastq.gz".
bowtie.threads	integer setting the number of threads to use for bowtie2.
results.directory	the location to place PING results. If left blank, a generic results directory will be created using the date and time of day.

Details

ping_run_all is meant to be a simplified way to run the entire PING pipeline. Once PING_run_all.R is loaded into R, the pipeline can be run using the ping_run_all() command (with the appropriate arguments).

ping_run_all reads in paired-end sequence files in sample.location, matching fastq.pattern. These sequence files are sent through ping_extractor, pulling out relevant KIR reads and creating new fastq (or fastq.gz) files with these reads, these new sequence files are deposited in "PING_sequences/". Once the KIR reads are extracted, they are sent through ping_gc_caller, where MIRA and custom probe searches are used to determine KIR gene presence and copy numbers, this step is interactive and will prompt the user for input to determine copy number threshold values. Once threshold values are set, ping_gc_caller produces a csv file with the gene content results. The gene content result file is then fed into ping_allele_caller, which uses bowtie2 and samtools to create VCF files for each sequence at each locus, genotypes are then called from these VCF files.

All results (except for extracted KIR reads) are stored in results.directory, which defaults to a All_results_[YEAR]_[MONTH]_[DAY]_[HOUR]_[MIN] naming format. Extracted KIR reads are stored in the “PING_sequences/” directory.

More detailed explanations of each of these steps can be found in the relevant section of this document.

PING_extractor_v1.0.R

Overview

PING_extractor is a script that extracts KIR reads from fastq files using bowtie2, and writes new fastq files with these reads. This is the first step in the PING pipeline, and the rest of the PING scripts depend on the output of PING_extractor. The extracted reads are placed in the “PING_sequences/” folder.

Dependencies

Linux (tested on Ubuntu 15.10 and CentOS 6.8)

Bowtie2 (tested with version 2.2.9) -- link

R (tested with version 3.2.3)

Usage

```
ping_extractor( sample.location = “Sequences/”, fastq.pattern.1 = “_1.fastq”,
        fastq.pattern.2 = “_2.fastq”, bowtie.threads = 4 )
```

Arguments

sample.location	the location of the fastq files to analyze, a trailing slash “/” is important.
fastq.pattern.1	a character vector to identify the pair split pattern, normal and gzipped files are accepted. eg “_1.fastq” or “_1.fastq.gz”.
fastq.pattern.2	a character vector to identify the pair split pattern, normal and gzipped files are accepted. eg “_2.fastq” or “_2.fastq.gz”.
bowtie.threads	integer setting the number of threads to use for bowtie2.

Details

ping_extractor reads in paired-end sequence files found in the sample.location folder, matching fastq.pattern. ping_extractor uses bowtie2 calls, along with a KIR filter, to extract KIR reads from the sequence files found in sample.location, and places these extracted reads in the "PING_sequences/" folder. If the input sequences are gzipped, the output sequences in "PING_sequences/" will also be gzipped.

The results of this script are essential for the following PING scripts.

PING_gc_caller_v1.1.R

Overview

PING_gc_caller is a script that determines KIR gene presence and copy numbers from the results of PING_extractor. This script is interactive and will prompt for user input to determine copy number threshold values based on graphs. Examples of these graphs can be found at the end of this document. Because of how long the MIRA and KFF portions of this script can take, the main ping_gc_caller() function outputs a file that can be used to redo the graphing and threshold setting without having to run KFF or MIRA a second time. The function for redoing graphing is called ping_recalc().

The csv results of PING_gc_caller are essential for running PING_allele_caller.

Dependencies

Linux (tested on Ubuntu 15.10 and CentOS 6.8)

MIRA (tested with version 4.0.2) -- link

R (tested with version 3.2.3)

R package dependencies:

data.table

Usage

ping_gc_caller(run.MIRA = T, run.KFF = T, make.graphs = T, sample.location = "PING_sequences/", threshold.file = "Resources/gc_resources/defaultThresholds.txt", threshold.KFF = 0.2, read.cap = 120000, results.directory = "")

ping_recalc(mira.csv = "MIRA_count_table.csv",

threshold.file = “Resources/gc_resources/defaultThresholds.txt”, make.graphs = T, results.directory = “”)

Arguments

run.MIRA	a logical value. If TRUE, the MIRA portion of the script will be run. If FALSE, the MIRA portion of the script will be skipped.
run.KFF	a logical value. If TRUE, the KFF portion of the script will be run. If FALSE, the KFF portion of the script will be skipped.
make.graphs	a logical value. If TRUE, the graphing portion of the MIRA script will be enabled, displaying graphs of read counts for each locus and prompting the user to specify copy number cutoff thresholds. If FALSE, the graphing portion will be skipped, and the threshold values specified in threshold.file will be used.
sample.location	the location of the KIR fastq files to analyze, a trailing slash “/” is important. Defaults to the output folder of ping_extractor(), which is “PING_sequences/”.
threshold.file	a character vector of the threshold file name to read and used for the MIRA portion of the script, default is set to the default threshold file found in the Reference folder.
threshold.KFF	the threshold to be used by the KFF portion of the script. It is not recommended to change this value.
read.cap	integer setting the max number of lines to use from the sequence files. Since four lines in a fastq file correspond to one read, the default of 120000 lines corresponds to 30000 reads.
results.directory	the location to place ping_gc_caller results. If left blank, a generic results directory will be created using the date and time of day.
mira.csv	the MIRA read count file upon which recalculations will be performed. Default is set to the output MIRA read count file produced by ping_gc.

Details

ping_gc_caller reads in paired-end sequence files found in the sample.location folder. These sequences are run through a KFF module that determines presence or absence of KIR loci, dependent on the threshold.KFF value, and a MIRA module that determines read counts for each locus.

If the make.graphs option is enabled, the read counts are plotted for each locus, and the user prompted to input the number of threshold values to be used as copy number cutoff points. Blue lines might appear on these graphs, these lines correspond to the threshold values that were specified in threshold.file. Possible values for this prompt are the numbers 1 – 10, or nil to keep the cutoffs represented by the blue lines. These cutoff points are usually associated with gaps in the graph, as can be seen in the examples at the end of this document. Once the number of threshold values is set, the user is prompted to click on points of the graph that correspond to each cutoff. Only the y values for these cutoff points are recorded, and any part of the graph can be clicked. Once the number of cutoff points matches the number of thresholds the user specified, red lines will appear on the graph, corresponding to the cutoff points, and the graphs legend will be updated. The user will then be prompted to confirm the cutoffs with a y/n response. If 'n', the graphing process will be restarted for that locus. If 'y', the graphing process will proceed to the next locus.

Once graphing is complete, ping_gc_caller will determine the copy number of each sequence based on the thresholds, and output a csv file with copy number results, a threshold file with the newly specified thresholds, a MIRA_count_table.csv file, and pdf's of the graphs. The MIRA_count_table.csv file can be input to ping_recalc to redo the graphing process without having analyze the sequences again.

ping_recalc is a repackaging of the MIRA module of ping_gc_caller, which allows the user to redo threshold value setting without having to repeat the process of analyzing sequences, which is by far the most time consuming part of ping_gc_caller.

Results

Results are saved in the GC_results_[year]_[month]_[day]_[hour]:[minute] directory, unless otherwise specified by results.directory. KFF results will be written to the kff_results.csv file, MIRA results will be written to the MIRA_results.csv file. If both KFF and MIRA are run, then a combined file will be made that combines the results and highlights any areas where KFF and MIRA differ. The combined results file is formatted so that inconsistent results will show up as "[KFF result]_[MIRA result]". If the make.graphs parameter was set to TRUE, pdf's of the graphs with the set cutoff points will be saved.

Two additional files, New_thresholds.txt and MIRA_count_table.csv, are both saved in the GC_results* directory and the working directory. These files are relevant for their use in running additional analyses. The New_thresholds.txt file is formatted to be used as input for the threshold.file parameter. The MIRA_count_table.csv file is the default input for ping_recalc, which will rerun copy number threshold setting without having to perform additional sequence analyzation. The files in the working directory will be overwritten in subsequent running of ping_gc_caller.

PING_allele_caller_v0.9.R

Overview

PING_allele_caller is a script that determines KIR genotypes from the results of PING_extractor and PING_gc_caller. PING_allele_caller uses a series of bowtie2 calls and Samtools to generate VCF files, which are then used to make genotype calls. For some loci, these genotype calls get further refined by using custom KFF probes.

Again, this is meant to be the last script in the pipeline, and cannot be run before PING_extractor and PING_gc_caller.

Dependencies

Linux (tested on Ubuntu 15.10 and CentOS 6.8)

Bowtie2 (tested with version 2.2.9) -- link

Samtools (tested with version 1.3.1) -- link

Bcftools (tested with version 1.3.1) -- link

R (tested with version 3.2.3)

R package dependencies:

data.table
ape
stringr

Usage

```
ping_allele_caller( sample.location = “PING_sequences/”, fastq.pattern.1 = “_1.fastq”,
    fastq.pattern.2 = “_2.fastq”, bowtie.threads = 4, supported.loci = c("2DL1", "2DL23", "2DL4",
    "2DL5", "2DS3", "2DS4", "2DS5", "2DP1", "3DL1", "3DS1", "3DL2", "3DL3"),
    ping.gc.output = “Combined_results.csv”, results.directory = “” )
```

Arguments

sample.location	the location of the KIR fastq files to analyze, a trailing slash “/” is important. Defaults to the output folder of ping_extractor(), which is “PING_sequences/”.

fastq.pattern.1	a character vector to identify the pair split pattern, normal and gzipped files are accepted. eg "_1.fastq" or "_1.fastq.gz".
fastq.pattern.2	a character vector to identify the pair split pattern, normal and gzipped files are accepted. eg "_2.fastq" or "_2.fastq.gz".
bowtie.threads	integer setting the number of threads to use for bowtie2.
supported.loci	a character vector containing all the loci ping_allele_caller supports. The default vector can be shortened to run less loci, but not added to.
ping.gc.output	the location of the ping_gc_caller csv output file. The easiest way to make this work is to copy the Combined_results.csv file from the GC_results directory into the working directory. ping_allele_caller requires this file to run. Also works with MIRA_results.csv and kff_results.csv.
results.directory	the location to place ping_allele_caller results. If left blank, a generic results directory will be created using the date and time of day.

Details

ping_allele_caller reads in paired-end sequence files in sample.location, matching fastq.pattern. These sequence files are truncated to anywhere between 120,000 – 280,000 lines to reduce noise and improve genotype calls, the truncation depends on which locus is being analyzed. These sequences are run through multiple bowtie2 calls using custom filters specific for each locus to create SAM files. The SAM files are fed into Samtools, again using custom filters, to generate VCF files for each sample and each locus. The genotype calling portion of the script compares these VCF files to known KIR allele alignments to generate genotype and snp calls.

Results

Results are saved in the Caller_results_[year]_[month]_[day]_[hour]:[minute] directory if no results.directory is specified. In this directory there will be the subdirectories Fastq, KIRcaller and Vcf. The Fastq folder holds KIR locus specific consensus sequence output by Bowtie2, the KIRcaller folder contains the output files from the genotype calling portion of the script, and the Vcf folder contains VCF files output from Samtools.

The completed genotype calls are written to the genos_[locus].txt file found in the KIRcaller folder. Other files in this folder show additional alignment information. All_[locus].txt summarizes all alleles that were considered in the genotype call, bad_[locus].txt shows any bad input files, indels_[locus].txt is a dump of any indels found in the VCF files, KIR_[locus]_alleles.txt is a list of SNPs and positions that make each allele unique, newalleles_[locus].txt shows any new alleles found, newsnps_[locus].txt shows any new snps found, and snps_[locus].txt shows nucleotide alignment for each sample.

There are some known errors and warnings that ping_allele_caller can work through, these errors are written to the Error.log file in the “Caller_results*/” folder.

Troubleshooting

If the script stops while MIRA is running, please look at MIRA_log.txt.

One issue that we have found is that the read pair identifier needs to be "/1" and "/2". If the read pairs are identified with ".1" and ".2", MIRA will say it cannot find any pairs.

Any fread errors are a result of files not having enough lines. These normally appear in PING_allele_caller and the default behavior of this script is to run the files through anyway.

These scripts are still under active development, if any bugs are encountered please email Wesley.Marin@ucsf.edu.

Recommendations

Batches of 90-100 samples originating from the same experiment will give the clearest clustering. For 2 X 300 bp paired end reads we recommend limiting the total number of pairs to 30,000.

Authors and citations

The PING scripts were written by Paul Norman, Wesley Marin and Jill Hollenbach. If you use this program please cite the following paper:

Norman et al., Defining KIR and HLA Class I Genotypes at Highest Resolution via High-Throughput Sequencing, The American Journal of Human Genetics (2016), http://dx.doi.org/10.1016/j.ajhg.2016.06.023

PING_gc_caller graphing example

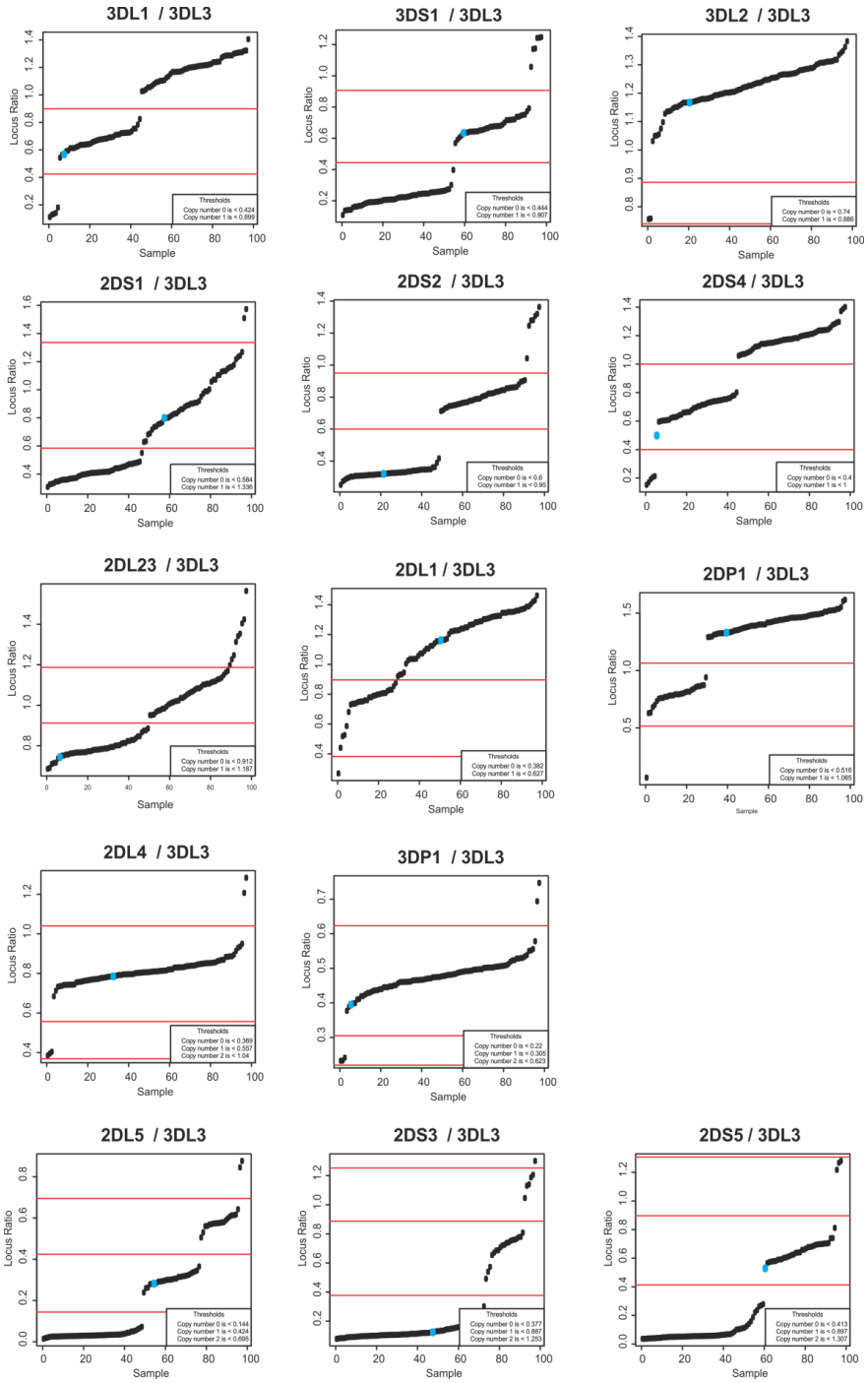

Figure 1. Examples for MIRA threshold values set using 96 samples and 2x300 bp MiSeq reads. Blue dots are the values obtained for the COX control cell line.