Joshua Conte

jconte@neontherapeutics.com

Abstract

This is an overview that takes a normal BAM file and a tumor BAM file and uses Polysolver, Sequenza, and LOHHLA to evaluate HLA loss.

LOHHLA Best Practice

Table of Contents

[1. Overview 2](#_Toc506815708)

[2. Software used: 2](#_Toc506815709)

[2.1. Dependencies 2](#_Toc506815710)

[3. Files used: 2](#_Toc506815711)

[4. Installing Software 3](#_Toc506815712)

[4.1. Polysolver 3](#_Toc506815713)

[4.2. Sequenza 3](#_Toc506815714)

[4.3. LOHHLA 3](#_Toc506815715)

[4.3.1. BEDTools 3](#_Toc506815716)

[4.3.2. SAMTools 3](#_Toc506815717)

[4.3.3. GATK 3.5 4](#_Toc506815718)

[4.3.4. Picard Tools 1.120 5](#_Toc506815719)

[4.3.5. NovoAlign 5](#_Toc506815720)

[5. Running the software 5](#_Toc506815721)

[5.1. Polysolver 5](#_Toc506815722)

[5.2. Sequenza 6](#_Toc506815723)

[5.3. LOHHLA 7](#_Toc506815724)

# Overview

This is an overview that takes a normal BAM file and a tumor BAM file and uses Polysolver, Sequenza, and LOHHLA to evaluate HLA loss.

# Software used:

* Polysolver (<http://archive.broadinstitute.org/cancer/cga/polysolver>)
* Sequenza (<http://www.cbs.dtu.dk/biotools/sequenza/>)
* LOHHLA (<https://bitbucket.org/mcgranahanlab/lohhla>)

## Dependencies

Please ensure a number of dependencies are first installed. These include:

* BEDTools (<http://bedtools.readthedocs.io/en/latest/>)
* SAMtools (<http://samtools.sourceforge.net/>)
* Novoalign (<http://www.novocraft.com/products/novoalign/>)
* Picard tools 1.120 (<https://github.com/broadinstitute/picard/releases/download/1.120/picard-tools-1.120.zip>)
* R (<https://www.r-project.org/about.html>)
* Perl 5.16 (or later version)

Within R, the following packages are required:

* seqinr (<https://CRAN.R-project.org/package=seqinr>)
* Biostrings (<http://bioconductor.org/packages/release/bioc/html/Biostrings.html>)
* beeswarm (<https://CRAN.R-project.org/package=beeswarm>)
* zoo (<https://cran.r-project.org/package=zoo>)
* Rsamtools (<http://bioconductor.org/packages/release/bioc/html/Rsamtools.html>)
* copynumber (<http://bioconductor.org/packages/release/bioc/html/copynumber.html>)

# Files used:

Files composed by Joshua Conte:

* SeqLOHHLA.pl (to produce hlas and patient.hlaFasta.fa files from the Polysolver output and hla.dat file supplied by LOHHLA)
* sequenzaLOHHLA.r (to produce solutions.txt file from the Sequenza Python file output)

Files required for Polysolver:

* Normal BAM file (produced from Neon Recon Pipeline)

Files required for Sequenza:

* Normal BAM file (produced from Neon Recon Pipeline)
* Tumor BAM file (produced from Neon Recon Pipeline)
* HG19.gencode.fasta (supplied from Reference files in the Neon Recon Pipeline)
* hg19.gc5Base.txt.gz (supplied by Sequenza downloaded from http://hgdownload-test.cse.ucsc.edu/goldenPath/hg19/gc5Base/)

Files required for LOHHLA:

* Normal BAM file (produced from Neon Recon Pipeline)
* Tumor BAM file (produced from Neon Recon Pipeline)
* Patient HLA calls file (produced by SeqLOHHLA.pl)
* HLA FASTA (produced by SeqLOHHLA.pl)
* Patient purity and ploidy output file (produced by sequenzaLOHHLA.r)
* hla.dat file (supplied by LOHHLA)

# Installing Software

Software packages and dependencies required to run LOHHLA.

## Polysolver

The first piece of software that needs to be installed is Polysolver. This is a Docker image and can be installed in Linux by typing:

docker pull sachet/polysolver:v4

## Sequenza

Sequenza is composed of an R script and a Python script. To get the Python file run:

pip install sequenza-utils

To get the R package, open R and type:

install.packages("sequenza")

When the R package is installed, it copies a PDF file in the library folder that has great references for this program called sequenza.pdf.

## LOHHLA

To install LOHHLA, simply clone the repository:

git clone https://bitbucket.org/mcgranahanlab/lohhla.git

The dependencies are listed below, and you need to make sure all the programs are added to PATH.

### BEDTools

LOHHLA requires BEDTools and the script to install it is shown below:

#!/bin/sh

wget https://github.com/arq5x/bedtools2/releases/download/v2.26.0/bedtools-2.26.0.tar.gz

tar -vxzf bedtools-2.26.0.tar.gz && rm -rf bedtools-2.26.0.tar.gz

cd bedtools2

make

### SAMTools

LOHHLA requires SAMTools and the script to install it is shown below:

#!/bin/sh

INSTLDIR=$PWD

# install libraries

sudo apt-get update

sudo apt-get install build-essential

sudo apt-get install libncurses5-dev libncursesw5-dev

sudo apt-get install libpng-dev

sudo apt-get install zlib1g-dev

sudo apt-get install libbz2-1.0 libbz2-dev libbz2-ocaml libbz2-ocaml-dev

sudo apt-get install liblzma-dev

# Move to install directory

cd $INSTLDIR

# install samtools

wget https://github.com/samtools/samtools/releases/download/1.6/samtools-1.6.tar.bz2

bunzip2 samtools-1.6.tar.bz2

tar xvf samtools-1.6.tar && rm samtools-1.6.tar

cd samtools-1.6

./configure --prefix=$INSTLDIR/samtools-1.6

make

make install

cd $INSTLDIR

echo 'export PATH=${PATH}:'$INSTLDIR'/samtools-1.6' >> $HOME/.bashrc

# relod .bashrc

cd

exec bash

cd $INSTLDIR

### GATK 3.5

LOHHLA requires GATK and a script to install GATK 3.5 is shown below:

#!/bin/sh

# Need to manually download package at:

# https://software.broadinstitute.org/gatk/download/auth?package=GATK-archive&version=3.5-0-g36282e4

INSTLDIR=$PWD

# Install GATK 3.5

bunzip2 GenomeAnalysisTK-3.5-0-g36282e4.tar.bz2

tar xvf GenomeAnalysisTK-3.5-0-g36282e4.tar && rm GenomeAnalysisTK-3.5-0-g36282e4.tar

echo 'export gatk='$INSTLDIR'/GenomeAnalysisTK.jar' >> $HOME/.bashrc

# relod .bashrc

cd

exec bash

cd $INSTLDIR

### Picard Tools 1.120

LOHHLA requires an old version of Picard tools, I recommend Picard tools version 1.120 and can be downloaded by typing:

wget \

https://github.com/broadinstitute/picard/releases/download/1.120/picard-tools-1.120.zip

The contents of this folder need to be installed in the same folder as GATK.

When all of the files from Picard Tools is added to the GATK folder, you need to run the libIntelDeflater.so file with execstack. Execstack is a program which sets, clears, or queries executable stack flag of ELF binaries and shared libraries. An example of the commands is shown below:

Sudo apt-get execstack

execstack -c /opt/GATK-3.5/libIntelDeflater.so

### NovoAlign

LOHHLA requires the software package NovoAlign. This software package requires a key which is supplied by NovoAlign.

To install LOHHLA clone the repository:

git clone https://bitbucket.org/mcgranahanlab/lohhla.git

Once the repository is cloned, the key should be put in the same folder.

# Running the software

Steps to run LOHHLA, the software packages need to be executed in the order shown below.

## Polysolver

Polysolver runs from a Docker image and it is really straightforward. You will need to make a script that defines where the normal BAM file directory, the name of the BAM file, and the name of the Docker project (which I usually define as the patient). This program usually takes about an hour or two to run, depending on the size of the data.

It’s important to note that the BAM files need to be in the same directory and the file names need to be short. I recommend a naming convention like this:

SR-16-665\_normal\_sorted.bam

SR-16-665\_normal\_sorted.bam.bai

SR-16-665\_tumor\_sorted.bam

SR-16-665\_tumor\_sorted.bam.bai

An example script is shown below:

#!/bin/sh

# Where BAM file is located

DIR=/home/jconte/reference

# Name of BAM file

BAM=" SR-16-665\_normal\_sorted.bam"

# Name of project

NAME="SR-16-680"

# Run polysolver

sudo docker run -d -P --name $NAME -v $DIR:/home/docker sachet/polysolver:latest bash /home/polysolver/scripts/shell\_call\_hla\_type /home/docker/$BAM Unknown 1 hg19 STDFQ 0 /home/docker

When the script is completed, Polysolver will produce a file called winners.hla.txt in the BAM file directory.

Next, you will need to use a Perl script that will prepare the winners.hla.txt file and format it so it can be used as an input for LOHHLA. The script requires three inputs, the winners.hla.txt file produced from Polysolver, a reference hla.dat file supplied from LOHHLA, and the patient name that will be used to name the files. The Perl script takes about 10 seconds to run. An example of how the script runs is also shown below:

# For help type ./SeqLOHHLA.pl --help

./SeqLOHHLA.pl \

--winner\_in /home/jconte/reference/winners.hla.txt \

--HLA\_DAT /home/jconte/installs/lohhla/data/hla.dat

Below is the script (double click to open):



This will produce two files for LOHHLA, the first is a hlas file that lists the patients HLA calls and the second is a HLA FASTA file that has the sequences for the HLA calls.

## Sequenza

The first step to running Sequenza is using the Python file. This takes the BAM files, fasta reference file, and the GC content from a genome FASTA (hg19.gc5Base.txt.gz) and makes seq file that contains genotype information, alleles and mutation frequency, and other features. Sequenza Python takes about 6 - 10 hours to run. An example script is shown below to run Sequenza:

#!/bin/sh

## Input files ##

# BAM files

normalBam=$HOME/reference/ SR-16-665\_normal\_sorted.bam

tumorBam=$HOME/reference/ SR-16-665\_tumor\_sorted.bam

# Fasta reference file

fasta\_ref=$HOME/reference/HG19.gencode.fasta

# gc fasta file

gcFasta=$HOME/reference/hg19.gc5Base.txt.gz

# Run Script

# install sequenza

pip install sequenza-utils

# Process BAM and Wiggle files to produce a seqz file:

sequenza-utils bam2seqz -n $normalBam -t $tumorBam --fasta $fasta\_ref \

-gc $gcFasta -o $seqName.out.seqz.gz

# Post-process by binning the original seqz file:

sequenza-utils seqz\_binning \

-w 50 \

-s $seqName.out.seqz.gz | gzip > $seqName.out.small.seqz.gz

This script will produce an out.small.seqz.gz file that will be used with the R sequenza package.

I made an R script, sequenzaLOHHLA.r, which will run Sequenza, install all of the required libraries (if needed), and make a properly formatted patient purity and ploidy output file (ends with solutions.txt) to run with LOHHLA. If the file is executed for the first time, sudo is recommended so all the required packages get properly installed. Sequenza R takes about 5 minutes to run. The user will need to specify the sequenza file and patient name, as shown below in the example:

# For help type Rscript sequenzaLOHHLA.r --help

sudo Rscript sequenzaLOHHLA.r --seqzFile /home/jconte/Neon/SR-16-680.out.small.seqz.gz --patient SR-16-680

Below is the script (double click to open):



## LOHHLA

You can run LOHHLA once all the required software and files have been produced. LOHHLA takes about 1 to 2 hours to run and the BAM file names need to be short, as referenced in section 5.1, an example shell script is shown below:

#!/bin/sh

Rscript /location/of/lohhla/repository/lohhla/LOHHLAscript.R \

# patient ID

--patientId example \

# location of output directory

--outputDir /out/example-out/ \

# normal BAM file

--normalBAMfile /location/of/lohhla/repository/lohhla/example-file/bam/example\_BS\_GL\_sorted.bam \

# location of all BAMs to test

--BAMDir /location/of/lohhla/repository/lohhla/example-file/bam/ \

# location to patient HLA calls

--hlaPath /location/of/lohhla/repository/lohhla/example-file/hlas \

#location of HLA FASTA [default=~/lohhla/data/hla\_all.fasta]

--HLAfastaLoc /location/of/lohhla/repository/data/example.patient.hlaFasta.fa \

#location to patient purity and ploidy output

--CopyNumLoc /location/of/lohhla/repository/lohhla/example-file/solutions.txt \

# does mapping to HLA alleles need to be done [default= TRUE]

--mappingStep TRUE \

# minimum coverage at mismatch site [default= 30]

--minCoverageFilter 10 \

# if mapping is performed, also look for fished reads matching kmers of size kmerSize

--fishingStep TRUE \

# remove temporary files [default= TRUE]

--cleanUp FALSE \

# path to GATK executable

--gatkDir /your/GATK/bin/ \

# path to novoalign executable

--novoDir /your/novoalign/bin/

# HLA exon boundaries for plotting [default=~/lohhla/data/hla.dat]

--HLAexonLoc /your/lohhla/data/hla.dat

The output files will be stored in the defined outputDir ready for review.

Below are two scripts that will automate the process. The files need to be run sequentially, and the second file can run when the polysolver docker container is finished.



Understanding LOHHLA output

LOHHLA will out put an Excel file called

### What is the output of LOHHLA?

LOHHLA produces multiple different files (see correct-example-out for an example). To determine HLA LOH in a given sample, the most relevant output is the file which ends '.HLAlossPrediction CI.xls'. The most relevant columns are:

|  |  |
| --- | --- |
| HLA\_A\_type1 | the identity of allele 1 |
| HLA\_A\_type2 | the identity of allele 2 |
| Pval\_unique | this is a p-value relating to allelic imbalance |
| LossAllele | this corresponds to the HLA allele that is subject to loss |
| KeptAllele | this corresponds to the HLA allele that is not subject to loss |
| HLA\_type1copyNum\_withBAFBin | the estimated raw copy number of HLA (allele 1) |
| HLA\_type2copyNum\_withBAFBin | the estimated raw copy number of HLA (allele 2) |

For a full definition of the columns, see below, in each case whether the column should be used [use], or can be ignored [legacy]is indicated:

|  |  |
| --- | --- |
| region | the region or tumor sample [use] |
| HLA\_A\_type1 | the identity of allele 1 [use] |
| HLA\_A\_type2 | the identity of allele 2 [use] |
| HLAtype1Log2MedianCoverage | the median LogR coverage across allele 1 [use] |
| HLAtype2Log2MedianCoverage | the median LogR coverage across allele 2 [use] |
| HLAtype1Log2MedianCoverageAtSites | the median LogR coverage across allele 1, restricted to mismatch sites [use] |
| HLAtype2Log2MedianCoverageAtSites | the median LogR coverage across allele 2, restricted to mismatch sites [use] |
| HLA\_type1copyNum\_withoutBAF | estimated copy number of allele 1, without using BAF [legacy] |
| HLA\_type1copyNum\_withoutBAF\_lower | lower 95% confidence interval of estimated copy number of allele 1, without using BAF [legacy] |
| HLA\_type1copyNum\_withoutBAF\_upper | upper 95% confidence interval of estimated copy number of allele 1, without using BAF [legacy] |
| HLA\_type1copyNum\_withBAF | estimated copy number of allele 1 using BAF, without binning sites [legacy] |
| HLA\_type1copyNum\_withBAF\_lower | lower 95% confidence interval of estimated copy number of allele 1 using BAF, without binning sites [legacy] |
| HLA\_type1copyNum\_withBAF\_upper | upper 95% confidence interval of estimated copy number of allele 1 using BAF, without binning sites [legacy] |
| HLA\_type2copyNum\_withoutBAF | estimated copy number of allele 2 without using BAF [legacy] |
| HLA\_type2copyNum\_withoutBAF\_lower | lower 95% confidence interval of estimated copy number of allele 2, without using BAF [legacy] |
| HLA\_type2copyNum\_withoutBAF\_upper | upper 95% confidence interval of estimated copy number of allele 2, without using BAF [legacy] |
| HLA\_type2copyNum\_withBAF | estimated copy number of allele 2 using BAF, without binning sites [legacy] |
| HLA\_type2copyNum\_withBAF\_lower | lower 95% confidence interval of estimated copy number of allele 1 using BAF, without binning sites [legacy] |
| HLA\_type2copyNum\_withBAF\_upper | upper 95% confidence interval of estimated copy number of allele 1 using BAF, without binning sites [legacy] |
| HLA\_type1copyNum\_withoutBAFBin | estimated copy number of allele 1 using binning, but without BAF [legacy] |
| HLA\_type1copyNum\_withoutBAFBin\_lower | lower 95% confidence interval of estimated copy number of allele 1 using binning, but without BAF [legacy] |
| HLA\_type1copyNum\_withoutBAFBin\_upper | upper 95% confidence interval of estimated copy number of allele 1 using binning, but without BAF [legacy] |
| HLA\_type1copyNum\_withBAFBin | estimated copy number of allele 1 using binning and BAF [use] |
| HLA\_type1copyNum\_withBAFBin\_lower | lower 95% confidence interval of estimated copy number of allele 1 using binning and BAF [use] |
| HLA\_type1copyNum\_withBAFBin\_upper | upper 95% confidence interval of estimated copy number of allele 1 using binning and BAF [use] |
| HLA\_type2copyNum\_withoutBAFBin | estimated copy number of allele 2 using binning, but without BAF [legacy] |
| HLA\_type2copyNum\_withoutBAFBin\_lower | lower 95% confidence interval of estimated copy number of allele 2 using binning, but without BAF [legacy] |
| HLA\_type2copyNum\_withoutBAFBin\_upper | upper 95% confidence interval of estimated copy number of allele 2 using BAF, without binning sites [legacy] |
| HLA\_type2copyNum\_withBAFBin | estimated copy number of allele 2 using binning and BAF [use] |
| HLA\_type2copyNum\_withBAFBin\_lower | lower 95% confidence interval of estimated copy number of allele 2 using binning and BAF [use] |
| HLA\_type2copyNum\_withBAFBin\_upper | upper 95% confidence interval of estimated copy number of allele 2 using binning and BAF [use |
| PVal | p-value relating to difference in logR between allele 1 and allele 2 (paired t-test)[legacy] |
| UnPairedPval | p-value relating to difference in logR between allele 1 and allele 2 (unpaired t-test)[legacy] |
| PVal\_unique | p-value relating to difference in logR between allele 1 and allele 2, ensuring each read only contributes once (paired t-test) [use] |
| UnPairedPval\_unique | p-value relating to difference in logR between allele 1 and allele 2, ensuring each read only contributes once (unpaired t-test) [use] |
| LossAllele | HLA allele that is present at lower frequency (potentially subject to loss) [use] |
| KeptAllele | HLA allele that is present at higher frequency (potentially not subject to loss) [use] |
| numMisMatchSitesCov | number of mismatch sites with sufficient coverage [use] |
| propSupportiveSites | proportion of missmatch sites that are consistent with loss or allelic imbalance [use] |