CompoundHetVIP-GMKF Usage Example

Dustin Miller

12/7/2020

Compound Heterozygous Variant Pipeline Example

This pipeline is designed to use gVCF data from the Gabriella Miller Kids First Data Resource Center (KFDRC). The files in the KFDRC are controlled access so no example files can be provided. However, this guide will show you the steps we used to obtain phased data that has been queried for compound heterozygous variants.

All python scripts are available at https://github.com/dmiller903/CompoundHetVIP-GMKF.

The steps of this pipeline assume that you have already installed Docker on the system where you will be processing your data.

Pull compound-het-vip Docker Image

First, it is necessary to pull the compound-het-vip image from Docker Hub. This image contains all tools needed for *CH* variant identification and will be used for all steps of the pipeline.

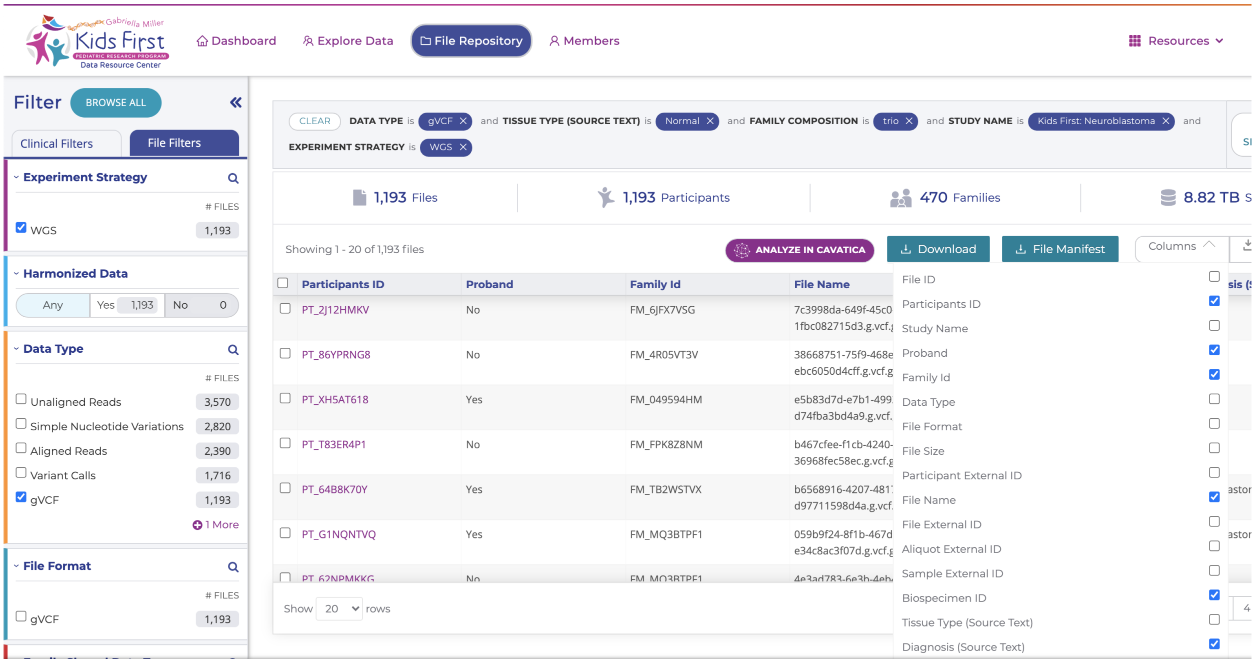
```
docker pull dmill903/compound-het-vip-gmkf:1.0
```

Consolidate table and clinical files

The table and clinical files (directions for obtaining these files are in the next paragraph) for the disease are needed for this step and can be obtained through the KFDRC. Each of these files contain information about the sample files. Using pieces from each of these files is necessary to determine family relationships, gender, and disease status. The consolidation of these files results in a tsv file with 5 columns: file_name, family_id, sample_id, proband (Yes, no), and sex (1 = male, 2 = female). This file is used repeatedly throughout the pipeline as an easy way to keep track of file names and family relationships, and is eventually used to create .fam files which are necessary for certain programs (e.g. *GEMINI*) used in the pipeline.

To obtain the table, make sure to apply the filters, and select the column boxes as seen below. Once filters are applied, and columns are selected, click "Export TSV". To obtain the clinical file, click the "Download" button and select "Clinical Data: Participants only".

Execute the "kids_first_meta.py" script in the dmill903/compound-het-vip-gmkf:1.0 docker container. You must attach a volume to the docker container. This volume is where the container will look for files and be able to save files to. In this example, our directory, "/Data/KidsFirst" is were the ch-pipeline repository was cloned to and within this directory is where the gVCF files from the KFDRC were downloaded, as well as the table and clinical data files. Within the container the "/Data/KidsFirst" directory is known as "/proj" as specified by "-v" and is set as the working directory with "-w". "-t" allows the container to use a terminal and "dmill903/compound-het-vip-gmkf:1.0" is the image that was previously pulled. The python script ("kids_first_meta.py") used for this step requires 3 arguments (in order): 1) table file, 2) clinical file, 3) output file name.



Experiment Strategy

	# FILES
☑ WGS	1,193

Harmonized Data

Any | Yes 1,193 | No 0

Data Type

	# FILES
☐ Unaligned Reads	3,570
☐ Simple Nucleotide Variations	2,820
☐ Aligned Reads	2,390
☐ Variant Calls	1,716
☑ gVCF	1,193

1 More

File Format

	# FILES
☐ gVCF	1,193

CLEAR **DATA TYPE** is gVCF and **TISSUE TYPE (SOURCE TEXT)** is Normal and **FAMILY COMPOSITION** is trio and **STUDY NAME** is Kids First: Neuroblastoma and **EXPERIMENT STRATEGY** is WGS

1,193 Files | 1,193 Participants | 470 Families | 8.82 TB

Showing 1 - 20 of 1,193 files

☐	Participants ID	Proband	Family Id	File Name
☐	PT_2J12HMKV	No	FM_6JFX7VSG	7c3998da-649f-45c0 1fbc082715d3.g.vcf.g
☐	PT_86YPRNG8	No	FM_4R05VT3V	38668751-75f9-468e ebc6050d4cff.g.vcf.g
☐	PT_XH5AT618	Yes	FM_049594HM	e5b83d7d-e7b1-499 d74fba3bd4a9.g.vcf.
☐	PT_T83ER4P1	No	FM_FPK8Z8NM	b467cfee-f1cb-4240- 36968fec58ec.g.vcf.g
☐	PT_64B8K70Y	Yes	FM_TB2WSTVX	b6568916-4207-481 d97711598d4a.g.vcf.
☐	PT_G1NQNTVQ	Yes	FM_MQ3BTPF1	059b9f24-8f1b-467d e34c8ac3f07d.g.vcf.g
☐	PT_62NPMKKG	No	FM_MQ3BTPF1	4e3ad783-6e3b-4eb

Show 20 rows

Columns:

File ID	☐
Participants ID	☑
Study Name	☐
Proband	☑
Family Id	☑
Data Type	☐
File Format	☐
File Size	☐
Participant External ID	☐
File Name	☑
File External ID	☐
Aliquot External ID	☐
Sample External ID	☐
Biospecimen ID	☑
Tissue Type (Source Text)	☐
Diagnosis (Source Text)	☑

Figure 1: Accessing the table and clincial files for Neuroblastoma

```
docker run -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /kids_first_meta.py \
  neuroblastoma/neuroblastoma_table.tsv \
  neuroblastoma/participant_clinical_20200420.tsv \
  neuroblastoma/consolidated.tsv
```

Keep variant-only sites and rename files

gVCF files are different from VCF files in that they contain information for every position, including non-variant positions. Therefore, these files are large and would take a long time to process if the non-variant positions were included throughout the whole compound heterozygous pipeline.

This step uses the “consolidated.tsv” file to create a folder for each family ID and then within each of these family ID folders, it creates a folder for each sample ID. It then takes each proband file and removes all non-variant sites. It outputs the parsed file to its corresponding sample ID folder. The script then filters each parent file for sites that are congruent with the proband of that family.

The “keep_variant_sites.py” script is needed for this step and takes 3 arguments: 1) “consolidated.tsv” file, 2) path where the original files downloaded from the KFDRC were saved, 3) the max number of cores to use. The difference with the “docker run” command used for this and subsequent steps is that we use the “-d” option. This option allows the container to run in the background and will save a log where you direct it to with “>” as seen below.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /keep_variant_sites.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  40 \
  > neuroblastoma/keep_variant_sites.out
```

Combine each trio into a single file

During this step, the parsed gVCF file of each individual will be used to create a combined trio file for each family. In addition, a .fam file will be created for each trio.

The “combine_trios.py” script used below executes *GATK4* to combine gVCF files of trios. *GATK4* requires reference genome VCF files be used. Here I create a sub directory, “references”, in my “/Data/KidsFirst” path where the script will install necessary reference files.

```
mkdir /Data/KidsFirst/references
```

The “combine_trios.py” script is needed for this step and takes 3 arguments: 1) “consolidated.tsv” file, 2) path where the original files downloaded from the KFDRC were saved, 3) the max number of cores to use. Above I created the sub directory “references” in my “/Data/KidsFirst” path. In this case, another “-v” needs to be added, “-v /Data/KidsFirst/references:/references”. The portion before the colon (:) is where the files will be saved and the portion after the colon is where that path is located in the container and this portion NEEDS to be “/references” for the script to work properly. Notice in the below example, another -v has been added.

```
docker run -d -v /Data/KidsFirst:/proj -v /Data/KidsFirst/references:/references \
  -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /combine_trios.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  25 \
  > neuroblastoma/combine_trios.out
```

Liftover trio files and individual files from GRCh38 to GRCh37

The files used in the KFDRC are aligned to GRCh38. This is an issue as the phasing program (*SHAPEIT2*) and *GEMINI* need the files to be aligned to GRCh37. Therefore, this step takes all the combined trio files, genotypes them (*GATK*), and converts their positions to GRCh37 positions (*Picard*). *Picard*’s Liftover tool requires reference files; use the same reference directory as created above and the script will save the reference files to this directory when the script is executed for the first time.

The “liftover.py” script is needed for this step and takes 2 arguments: 1) “consolidated.tsv” file, 2) the path where the original files downloaded from the KFDRC were saved. In addition, another “-v” needs to be added, “-v /Data/KidsFirst/references:/references” since reference files are needed.

```
docker run -d -v /Data/KidsFirst:/proj -v /Data/KidsFirst/references:/references \
  -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /liftover.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  > neuroblastoma/liftover.out
```

Remove unplaced, and multiallelic sites and duplicate sites from lifted files

During liftover, some randomly placed sites are included in the VCF file. These randomly placed sites are those that are in GRCh38 but the exact position in GRCh37 isn’t known. Therefore, for subsequent analysis, these sites are removed. Only sites with known positions, on a known chromosome are kept. In addition, positions that are multiallelic or are duplicates are removed because programs such as *PLINK* (used next) and *SHAPEIT2* (used later) can not handle these types of sites. For trios, sites that contain missing genotype information (i.e. "./.") for both parents are removed to improve phasing accuracy.

The "remove_unplaced_multiallelic.py" script is needed for this step and takes 3 arguments: 1) "consolidated.tsv" file , 2) the path where the original files downloaded from the KFDRC were saved, 3) the max number of cores to use.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /remove_unplaced_multiallelic.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  23 \
  > neuroblastoma/remove_unplaced_multiallelic.out
```

Separate each trio and individual file into chromosome files, then generate plink files for trio's only

SHAPEIT2 requires that chromosomes be phased separately. PLINK files are also needed by *SHAPEIT2* in order to phase. Therefore, this script separates each trio into chromosome files. This step also uses *PLINK* to generate the necessary PLINK files needed for phasing.

The "separate_chr_generate_plink.py" script is needed for this step and takes 3 arguments: 1) "consolidated.tsv" file, 2) path where the original files downloaded from the KFDRC were saved, 3) the max number of cores to use.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /separate_chr_generate_plink.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  23 \
  > neuroblastoma/separate_chr_generate_plink.out
```

Phase each of the trios with a haplotype reference panel

During this step, each chromosome PLINK file of each trio is phased using *SHAPEIT2*. The parameters for phasing are set so that *SHAPEIT2* uses family relationship genotype information and also uses a haplotype reference panel. *SHAPEIT2* will integrate 1000 Genomes Project phase 3 haplotype reference panel. The files used for phasing can be found at https://mathgen.stats.ox.ac.uk/impute/1000GP_Phase3.html and ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/release/20130502/. Files from these sites will be downloaded to your local/remote machine, to the path you designate, the first time the phasing script is ran. Use the same reference directory created earlier in the pipeline.

The "phase_with_shapeit.py" script is needed for this step and takes 2 arguments: 1) "consolidated.tsv" file, 2) the path where the original files downloaded from the KFDRC were saved. In addition, another "-v" needs to be added, "-v /Data/KidsFirst/references:/references" since reference files are needed.

```
docker run -d -v /Data/KidsFirst:/proj -v /Data/KidsFirst/references:/references \
  -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /phase_with_shapeit.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  > neuroblastoma/phase_with_shapeit.out
```

Revert REF/ALT to be congruent with reference panel

The phased results can have the REF and ALT alleles switched as compared to the reference genome. We are unsure exactly why this occurs. It may be that since each trio file only has 3 samples, the ALT allele is more common in the trio and becomes the REF. This step ensures that the REF/ALT alleles of the phased VCF files are congruent with the REF/ALT of the reference genome. In addition, sites with Mendel errors are removed.

The "alt_ref_revert.py" script is needed for this step and takes 2 arguments: 1) "consolidated.tsv" file, 2) the path where the original files downloaded from the KFDRC were saved. In addition, another "-v" needs to be added, "-v /Data/KidsFirst/references:/references" since reference files are needed.

```
docker run -d -v /Data/KidsFirst:/proj -v /Data/KidsFirst/references:/references \
  -w /proj -t compound-het-vip \
  python3 /alt_ref_revert.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  > neuroblastoma/alt_ref_revert.out
```

Concat and merge phased trio chromosome files into one VCF file

To make subsequent analysis of the phased files easier, this step concatenates all phased chromosomes for each trio into a single trio file using *bcftools*. Then each concatenated trio file is merged into a single file (using *bcftools*) that contains all trios. In addition, a .fam file is created for this single, merged file.

The "concat_merge_phased_vcf.py" script is needed for this step and takes 3 arguments: 1) "consolidated.tsv" file, 2) path where the original files downloaded from the KFDRC were saved, 3) the max number of cores to use.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /concat_merge_phased_vcf.py \
  neuroblastoma/consolidated.tsv \
  neuroblastoma/gVCF \
  22 \
  > neuroblastoma/concat_merge_phased_vcf.out
```

Trim and normalize VCF file

Prior to annotation and loading a database into *GEMINI*, *GEMINI* recommends left trimming and normalizing VCF files. Files that are not left trimmed and normalized can be annotated incorrectly and may be incorrectly handled by GEMINI. This step uses *vt tools* to trim and normalize the phased VCF file.

The "vt_split_trim_left_align.py" script is needed for this step and requires 1 argument: 1) the path where the original files downloaded from the KFDRC were saved. In addition, another "-v" needs to be added, "-v /Data/KidsFirst/references:/references" since reference files are needed.

```
docker run -d -v /Data/KidsFirst:/proj \
  -v /Data/KidsFirst/references:/references \
  -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /vt_split_trim_left_align.py \
  neuroblastoma/gVCF \
  > neuroblastoma/vt.out
```

Annotate with snpEff

snpEff is used to annotate. Annotation provides information on the effects of variants on known genes. *GEMINI* allows users to explore genetic variation based on the annotations of the genome.

snpEff requires reference files. Using the same path set up earlier for references, add a sub directory to this path, such as "snpEff_data". This will help keep files organized but isn't necessary.

```
mkdir /Data/KidsFirst/references/snpEff_data
```

The "annotate.py" script is needed for this step and requires 1 arguments: 1) the path where the original files downloaded from the KFDRC were saved. snpEff requires annotation reference files in order to annotate. These will be downloaded when the "annotate.py" script is executed for the first time. In the docker command, another "-v" needs to be set to where you want the reference files saved. Above I created the sub directory "snpEff_data" in my "/Data/KidsFirst/references" path. In this case, another "-v" needs to be added, "-v /Data/KidsFirst/references/snpEff_data:/snpEff/./data/GRCh37.75". The portion before the colon (:) is where the files will be saved and the portion after the colon is where that path is located in the container and this portion NEEDS to be "/snpEff/./data/GRCh37.75" for the script to work properly.

```
docker run -d -v /Data/KidsFirst:/proj \
  -v /Data/KidsFirst/references/snpEff_data:/snpEff/./data/GRCh37.75 \
  -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /annotate.py \
  neuroblastoma/gVCF \
  > neuroblastoma/annotate.out
```

Load VCF as GEMINI database

GEMINI is used to query the VCF file for *CH* variants. GEMINI requires that the input VCF file be loaded as a GEMINI database. Once in a GEMINI database, various queries can be used to analyze variant data. Our scripts are tailored to identifying *CH* and *de novo* variants.

Using the same path set up earlier for references, add a sub directory to this path, such as "gemini_data". This will help keep files organized but isn't necessary.

```
mkdir /Data/KidsFirst/references/gemini_data
```

The "gemini_load.py" script is needed for this step and requires 2 arguments: 1) the path where the original files downloaded from the KFDRC were saved, 2) Number of computer cores to use. GEMINI databases can take a while to be created, but can be significantly sped up with the number of cores that are available for use. In order to load a GEMINI database, annotation files are necessary. In addition, CADD scores are required by us, not GEMINI, because many of our queries involve CADD scores. These files will be downloaded the first time the "gemini_load.py" script is executed first time. In the docker command, another "-v" needs to be set to where you want the reference files saved. Above I created the sub directory "gemini_data" in my "/Data/KidsFirst/references" path. In this case, another "-v" needs to be added, "-v /Data/KidsFirst/references/gemini_data:/usr/local/share/gemini/gemini_data". The portion before the colon (:) is where the files will be saved and the portion after the colon is where that path is located in the container and this portion NEEDS to be "/usr/local/share/gemini/gemini_data" for the script to work properly.

```
docker run -d -v /Data/KidsFirst:/proj \
  -v /Data/KidsFirst/references/gemini_data:/usr/local/share/gemini/gemini_data \
  -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /gemini_load.py \
  neuroblastoma/gVCF \
  43 \
  > neuroblastoma/gemini_load.out
```

Identify *CH* or homozygous alternate variants

To perform a query for *CH* variants, use "identify_CH_variants.py". This script has 3 arguments: 1) the path where the original files downloaded from the KFDRC were saved, 2) the minimum CADD score, 3) the maximum minor allele frequency. In the below example, during CH variant identification, variants that are of HIGH impact or are loss of function and have a minor allele frequency <= 1% are kept. If variants are of MED impact, only those that have a minor allele frequency <= 1% and have a CADD score >= 15 are kept. If either parent is homozygous alternate at a position in the CH variant positions of the child, the position is not considered. Also, if either parent has CH variants at the same positions as the child, these CH variants are not included in the final output file.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /identify_CH_variants.py \
  neuroblastoma/gVCF \
  15 \
  0.05 \
  > neuroblastoma/identify_CH_variants_gnomAD.out
```

To perform a query for homozygous alternate variants, use "identify_homAlt_variants.py". This script has 3 arguments: 1) the path where the original files downloaded from the KFDRC were saved, 2) the minimum CADD score, 3) the maximum minor allele frequency. In the below example, during variant identification, variants that are of HIGH impact or are loss of function and have a minor allele frequency <= 1% are kept. If variants are of MED impact, only those that have a minor allele frequency <= 1% and have a CADD score >= 15 are kept. Homozygous alternate variants where the parent also has a HA variant at the same position as the child are not included in the final output file.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /identify_homAlt_variants.py \
  neuroblastoma/gVCF \
  15 \
  0.05 \
  > neuroblastoma/identify_homAlt_variants_gnomAD.out
```

To perform a query for de novo variants, use "identify_deNovo_variants.py". This script has 3 arguments: 1) the path where the original files downloaded from the KFDRC were saved, 2) the minimum CADD score, 3) the maximum minor allele frequency. In the below example, during variant identification, variants that are of HIGH impact or are loss of function and have a minor allele frequency <= 1% are kept. If variants are of MED impact, only those that have a minor allele frequency <= 1% and have a CADD score >= 15 are kept.

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /identify_deNovo_variants.py \
  neuroblastoma/gVCF \
  15 \
  0.01 \
  > neuroblastoma/identify_deNovo_variants.out
```

(Optional) Add Gene Damage Index Scores and Gene lengths to CH and/or homozygous alternate files

The Gene Damage Index (GDI) is a way to rank genes based on their likelihood of harboring disease-causing variants. The “add_GDI_and_gene_lengths.py” script adds GDI scores to the CH called and/or homozygous alternate file and also adds CDS and total gene length information. The “add_GDI_and_gene_lengths.py” script requires 2 arguments: 1) the input file (CH file in this case), 2) the output file. In addition, there are 3 optional arguments: 1) another input file (the homozygous alternate file in this case), 2) another output file, 3) whether or not the sample names are to be anonymized (default=y).

```
docker run -d -v /Data/KidsFirst:/proj -w /proj -t dmill903/compound-het-vip-gmkf:1.0 \
  python3 /add_GDI_and_gene_lengths.py \
  neuroblastoma/gVCF/neuroblastoma_CH_cadd15_maf05_gnomAD.tsv \
  neuroblastoma/gVCF/neuroblastoma_CH_cadd15_maf05_GDI.tsv \
  --input_file_2 neuroblastoma/gVCF/neuroblastoma_homAlt_cadd15_maf05_gnomAD.tsv \
  --output_file_2 neuroblastoma/gVCF/neuroblastoma_homAlt_cadd15_maf05_GDI.tsv \
  --anonymize y \
  > neuroblastoma/add_GDI_and_gene_lengths_gnomAD.out
```