Toward a methodology for evaluating DNA variants in nuclear families: data processing and analysis

Dustin Miller

06/10/2021

VCF Processing

For each step of VCF filtering, we used a Docker image that encapsulated the software tools as well as Python scripts (available on github) to process the data. These Python scripts either used custom code or a combination of custom code and existing software. Some of the scripts used during the VCF filtering process were adapted from CompoundHetVIP.

The BAM files used as part of our analyses are available from the European Genome-Phenome Archive (accession: EGAS00001005321).

A more detailed description of each processing step is available in the manuscript.

Filter out poor quality variants for each sample and each file type

Initial files being filtered are those that were output by *Long Ranger*. *Long Ranger* output a phased SNP VCF, a phased mid-scale deletion VCF, and a phased large-scale SV VCF for each sample.

```
# Phased SNP files
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/keep_passed_variants.py \
  PedFam/*/outs/phased_variants.vcf.gz \
  phased_files/ \
  passed_variants_summary.tsv \
  > keep_passed_variants.out

# Phased dels files
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/keep_passed_variants.py \
  PedFam/*/outs/dels.vcf.gz \
  dels_files/ \
  passed_dels_summary.tsv \
  > keep_passed_dels.out

# Phased svs files
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/keep_passed_variants.py \
  PedFam/*/outs/large_svs.vcf.gz \
```

```
  svs_files/ \
  passed_svs_summary.tsv \
  > keep_passed_svs.out
```

Combine sample files

```
# Filtered SNP files
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/concat_merge_phased_vcf.py \
  phased_files/ \
  combined.vcf.gz \
  --output_fam_file combined.fam \
  --concat_files n \
  --merge_files all \
  > concat_merge_phased_vcf.out

# Filtered dels files
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/concat_merge_phased_vcf.py \
  dels_files/ \
  dels_combined.vcf.gz \
  --output_fam_file dels_combined.fam \
  --concat_files n \
  --merge_files none \
  > concat_merge_dels.out

# Filtered SVs files
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/concat_merge_phased_vcf.py \
  svs_files/ \
  svs_combined.vcf.gz \
  --output_fam_file svs_combined.fam \
  --concat_files n \
  --merge_files none \
  > concat_merge_svs.out
```

Normalized and left aligned with vt tools

*Only the combined SNP file was left aligned and normalized. The SV and del combined files were not able to be left aligned and normalized due to limitations in the reference genomes used.

```
# Combined SNP file
docker run -d -v /Data:/proj -v /Data/references:/references -w /proj \
  -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/vt_split_trim_left_align.py \
  combined.vcf.gz \
  combined_vt.vcf.gz \
  > vt_split_trim_left_align.out
```

Annotate

```
# Combined, normalized SNP file
docker run -d -v /Data:/proj \
  -v /Data/references/snpEff_data:/snpEff/./data/GRCh38.86 -w /proj \
  -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/annotate.py \
  combined_vt.vcf.gz \
  combined_annotated.vcf \
  > annotate.out

# Combined del file
docker run -d -v /Data:/proj \
  -v /Data/references/snpEff_data:/snpEff/./data/GRCh38.86 -w /proj \
  -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/annotate.py \
  dels_combined.vcf.gz \
  dels_annotated.vcf \
  > annotate_dels.out

# Combined SV file
docker run -d -v /Data:/proj \
  -v /Data/references/snpEff_data:/snpEff/./data/GRCh38.86 -w /proj \
  -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/annotate.py \
  svs_combined.vcf.gz \
  svs_annotated.vcf \
  > annotate_svs.out
```

Load files into a GEMINI Database

```
# Annotated SNP file
docker run -d -v /Data:/proj \
  -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/gemini_load.py \
  combined_annotated.vcf \
  combined.db \
  --fam_file combined.fam \
  > gemini_load.out

# Annotated del file
docker run -d -v /Data:/proj \
  -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/gemini_load.py \
  dels_annotated.vcf \
  dels_combined.db \
  --fam_file dels_combined.fam \
  > gemini_load_dels.out

# Annotated SV file
docker run -d -v /Data:/proj \
```

```
  -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/gemini_load.py \
  svs_annotated.vcf \
  svs_combined.db \
  --fam_file svs_combined.fam \
  > gemini_load_svs.out
```

Create a file that contains global gnomAD minor allele frequencies, global 1000 Genomes Project minor allele frequences, and CADD scores

This file was used to provide each variant (when possible) with minor allele frequency, and a CADD score. File was output as `gnomAD_1K_cadd_GRCh38.tsv.gz`.

```
docker run -d -v /Data:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/create_gnomAD_1K_cadd_file.py \
  > create_gnomAD_1K_cadd_file.out
```

Identify compound heterozygous variants

```
# SNP GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_CH_variants.py \
  combined.db \
  PedFam_CH_cadd20_maf01.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file combined.fam \
  --cadd 20 \
  --af 0.01 \
  > identify_CH_variants.out

# del GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_CH_variants.py \
  dels_combined.db \
  PedFam_CH_cadd20_maf01_dels.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file dels_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_CH_variants_dels.out

# SV GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_CH_variants.py \
  svs_combined.db \
  PedFam_CH_cadd20_maf01_svs.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
```

```
  --fam_file svs_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_CH_variants_svs.out
```

Identify de novo variants

```
# SNP GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_deNovo_variants.py \
  combined.db \
  PedFam_deNovo_cadd20_maf01.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  combined.fam \
  --cadd 20 \
  --af 0.01 \
  > identify_deNovo_variants.out

# del GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_deNovo_variants.py \
  dels_combined.db \
  PedFam_deNovo_cadd20_maf01_dels.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  dels_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_deNovo_variants_dels.out

# SV GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_deNovo_variants.py \
  svs_combined.db \
  PedFam_deNovo_cadd20_maf01_svs.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  svs_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_deNovo_variants_svs.out
```

Identify homozygous alternate variants

```
# SNP GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_homAlt_variants.py \
  combined.db \
  PedFam_homAlt_cadd20_maf01.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
```

```
  --fam_file combined.fam \
  --cadd 20 \
  --af 0.01 \
  > identify_homAlt_variants.out

# del GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_homAlt_variants.py \
  dels_combined.db \
  PedFam_homAlt_cadd20_maf01_dels.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file dels_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_homAlt_variants_dels.out

# SV GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_homAlt_variants.py \
  svs_combined.db \
  PedFam_homAlt_cadd20_maf01_svs.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file svs_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_homAlt_variants_svs.out
```

Identify heterozygous variants

```
# SNP GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_het_variants.py \
  combined.db \
  PedFam_het_cadd20_maf01.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file combined.fam \
  --cadd 20 \
  --af 0.01 \
  > identify_het_variants.out

# del GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_het_variants.py \
  dels_combined.db \
  PedFam_het_cadd20_maf01_dels.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file dels_combined.fam \
  --cadd 20 \
```

```
  --af 0.01 \
  --impact_filter_only y \
  > identify_het_variants_dels.out

# SV GEMINI database
docker run -d -v /Data/:/proj -w /proj -t dmill903/compound-het-vip:1.0 \
  python3 PedFam/scripts/identify_het_variants.py \
  svs_combined.db \
  PedFam_het_cadd20_maf01_svs.tsv \
  gnomAD_1K_cadd_GRCh38.tsv.gz \
  --fam_file svs_combined.fam \
  --cadd 20 \
  --af 0.01 \
  --impact_filter_only y \
  > identify_het_variants_svs.out
```

Pediatric Cancer Prediction Analysis with R (and a little Python)

Functions

```
# Combines dataframes, as long as there is information in the dataframes
combine_df <- function(aList) {
  newList <- list()
  for (df in aList) {
    if (nrow(df) > 0) {
      newList <- c(newList, list(df))
    }
  }
  newDf <- bind_rows(newList)
  return(newDf)
}

# Reads in a csv and removes parental samples
read_tsv_custom <- function(inputFile) {
  return(read_tsv(inputFile, col_types = cols(alt = col_character(),
                                              ref = col_character(),
                                              af_gnomAD = col_character(),
                                              cadd = col_character())) %>%
           dplyr::filter(!sample %in% c("4478_SP_0023", "4478_SP_0024",
                                        "4478_SP_0030", "4478_SP_0031",
                                        "4478_SP_0036", "4478_SP_0037")))
}

# Variation of the other custom csv script. For csv's that don't have all the
# columns that are seen in the other csv script.
read_tsv_custom_2 <- function(inputFile) {
  return(read_tsv(inputFile) %>% dplyr::filter(!sample %in% c("4478-SP-0023",
                                                              "4478-SP-0024",
                                                              "4478-SP-0030",
```

```
                                                          "4478-SP-0031",
                                                          "4478-SP-0036",
                                                          "4478-SP-
0037")))
}

# Returns only the astrocytoma samples
astrocytoma_sample_filter <- function(inputDf) {
  return(dplyr::filter(inputDf, sample %in% c("4478_SP_0025", "4478_SP_0026",
                                              "4478_SP_0027", "4478_SP_0028",
                                              "4478_SP_0029")))
}

# Returns only the wilms tumor samples
wilms_sample_filter <- function(inputDf) {
  return(dplyr::filter(inputDf, sample %in% c("4478_SP_0032", "4478_SP_0033",
                                              "4478_SP_0034",
"4478_SP_0035")))
}

# Returns only the lymphoma samples
lymphoma_sample_filter <- function(inputDf) {
  return(dplyr::filter(inputDf, sample %in% c("4478_SP_0038", "4478_SP_0039",
                                              "4478_SP_0040", "4478_SP_0041",
                                              "4478_SP_0042", "4478_SP_0043",
                                              "4478_SP_0044")))
}

# Adds a column to the input dataframe that indicates the disease type
add_disease_column <- function(inputDf) {
  astrocytoma_temp <- dplyr::filter(inputDf, sample %in% c("25", "26", "27",
                                                           "28", "29")) %>%
    mutate(disease = "low-grade astrocytoma")

  lymphoma_temp <- dplyr::filter(inputDf, sample %in% c("38", "39", "40",
"41",
                                                        "42", "43", "44"))
%>%
    mutate(disease = "Burkitt's lymphoma")

  wilms_temp <- dplyr::filter(inputDf, sample %in% c("32", "33", "34", "35"))
%>%
    mutate(disease = "Wilms tumor")

  return(bind_rows(astrocytoma_temp, lymphoma_temp, wilms_temp))
}

# Summarize the variants to the gene level
gene_level_filter <- function(df, disease) {
```

```r
  tempDf <- select(df, sample, gene)
  finalDf <- add_column(tempDf, `disease` = disease)
  return(finalDf)
}

# Summarize how many variants per sample
variant_level_filter <- function(df) {
  tempDf <- filter(df, variant_type != "Compound Heterozygous") %>%
    group_by(sample, variant_type, class) %>% summarise(variants = n())

  tempDf2 <- filter(df, variant_type == "Compound Heterozygous") %>%
    group_by(sample, variant_type, class, gene) %>%
    summarise(variants = length(unique(gene))) %>% select(!gene)

  finalDf <- bind_rows(tempDf, tempDf2)
  return(finalDf)
}

variant_level_filter_2 <- function(df) {
  finalDf <- group_by(df, sample, variant_type, class) %>% summarise(variants
= n())
  return(finalDf)
}

# Summarize shared variants
shared_variant_filter <- function(df) {
  tempDf <- group_by(df, chrom, start, ref, alt, family, sample) %>%
    summarise(shared_variants = n())

  tempDf <- group_by(tempDf, chrom, start, ref, alt, family) %>%
    summarise(shared_variants = n())
  return(tempDf)
}

# add family ID to df
add_family_ID <- function(df){
  fam1 <- filter(df, sample %in% c("4478_SP_0025", "4478_SP_0026",
                                   "4478_SP_0027", "4478_SP_0028",
                                   "4478_SP_0029", "25", "26", "27",
                                   "28", "29")) %>%
    mutate(family = "Family 1")

  fam2 <- filter(df, sample %in% c("4478_SP_0032", "4478_SP_0033",
                                   "4478_SP_0034", "4478_SP_0035",
                                   "32", "33", "34", "35")) %>%
    mutate(family = "Family 2")

  fam3 <- filter(df, sample %in% c("4478_SP_0038", "4478_SP_0039",
                                   "4478_SP_0040", "4478_SP_0041",
```

```
                                    "4478_SP_0042", "4478_SP_0043",
                                    "4478_SP_0044", "38", "39", "40",
                                    "41", "42", "43", "44")) %>%
    mutate(family = "Family 3")

  return(bind_rows(fam1, fam2, fam3))
}
```

Variant Stats of Passing and Phased variants

The passed_variants_summary.tsv, passed_dels_summary.tsv, and passed_svs_summary.tsv were generated with the keep_passed_variants.py script.

```
passed_variants <- read_tsv_custom_2("passed_variants_summary.tsv") %>%
  arrange(sample)

passed_dels <- read_tsv_custom_2("passed_dels_summary.tsv") %>%
arrange(sample)

passed_svs <- read_tsv_custom_2("passed_svs_summary.tsv") %>% arrange(sample)

passed_combined <- bind_rows(passed_variants, passed_dels, passed_svs) %>%
  group_by(sample) %>% summarise_all(sum) %>%
  mutate(passed_percentage = (num_passed/total_variants) * 100,
           passed_phased_percentage = (num_passed_phased/num_passed) * 100)
%>%
  add_disease_column()

write_tsv(passed_combined, "passed_variant_summary")

passed_tidy <- gather(passed_combined, category, "value", 2:6)

paste("Average number of total variants sequenced: ",
      mean(passed_combined$total_variants))

## [1] "Average number of total variants sequenced:  6058006.5"

paste("Average number of variants that passed quality filters: ",
      mean(passed_combined$num_passed))

## [1] "Average number of variants that passed quality filters:  4395811"
```

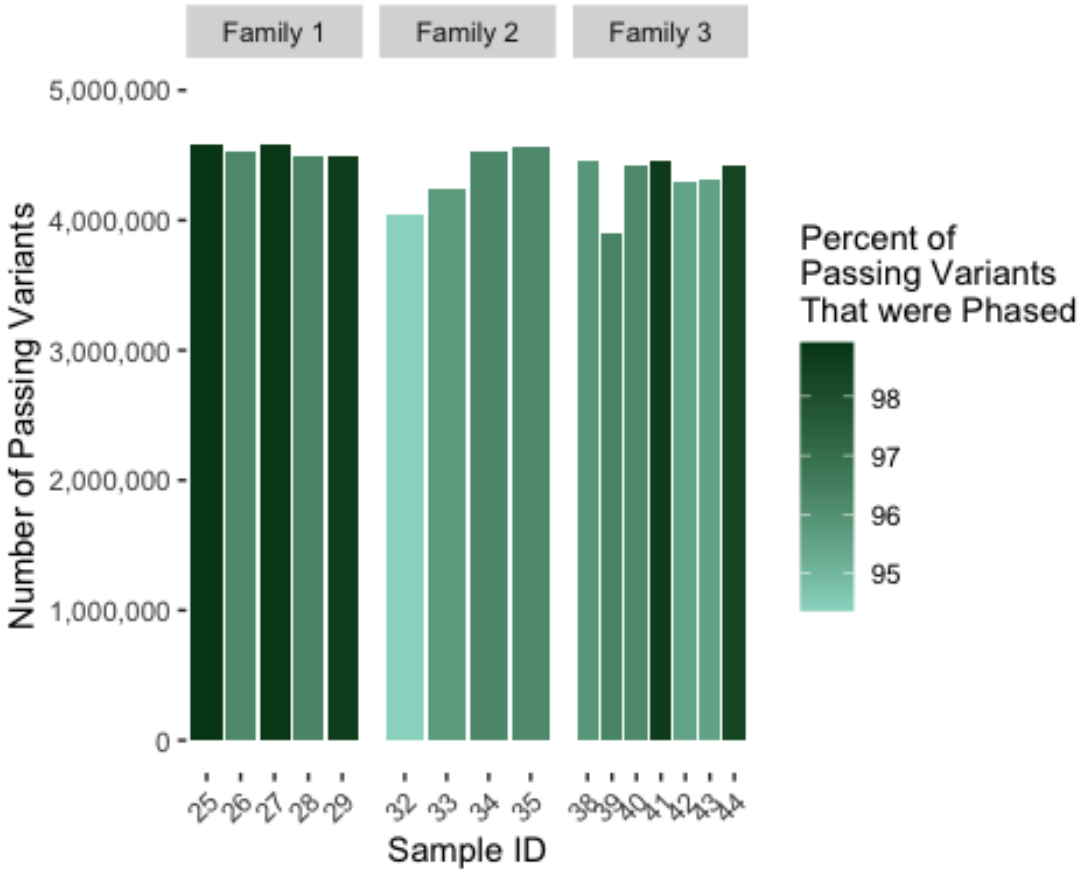
Create figure of passing/phased numbers

This code chunk was used to create Figure 2 in the manuscript.

```
passed_combined <- add_family_ID(passed_combined)

ggplot(passed_combined, aes(as.character(sample), num_passed)) +
  geom_col(aes(fill = passed_phased_percentage)) +
```

```
  theme(panel.background = element_rect(fill= "white"),
        axis.text.y = element_text(hjust = 1),
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_y_continuous(labels = scales::comma, limits = c(0, 5000000),
                     breaks = c(0, 1000000, 2000000, 3000000, 4000000,
5000000),
                     minor_breaks = waiver()) +
  scale_fill_gradient(low = "#99d8c9", high = "#00441b") +
  facet_grid(~family, scale="free_x") +
  labs(x="Sample ID", y="Number of Passing Variants",
       fill="Percent of\nPassing Variants\nThat were Phased")
```




```
ggsave("Figure_2.png", plot = last_plot())
```

Read in CH data and separate by disease

```
# Read in CH data and generate new columns with variant type and class
combined_CH_data_vars <- read_tsv_custom("PedFam_CH_cadd20_maf01.tsv") %>%
  add_column(variant_type = "Compound Heterozygous", class = "SNP")

combined_CH_data_svs <- read_tsv_custom("PedFam_CH_cadd20_maf01_svs.tsv") %>%
  add_column(variant_type = "Compound Heterozygous", class = "SV")

combined_CH_data_dels <- read_tsv_custom("PedFam_CH_cadd20_maf01_dels.tsv")
```

```
%>%
  add_column(variant_type = "Compound Heterozygous", class = "indel")

# Combine all imported data into a single dataframe
combined_CH_all <- combine_df(list(combined_CH_data_vars,
                                   combined_CH_data_svs,
combined_CH_data_dels))

# Filter down to astrocytoma data and filter to gene level
astrocytoma_CH_data <- astrocytoma_sample_filter(combined_CH_all)
astrocytoma_CH_gene_level <- gene_level_filter(astrocytoma_CH_data,
"astrocytoma")

# Filter down to wilms data and filter to gene level
wilms_CH_data <- wilms_sample_filter(combined_CH_all)
wilms_CH_gene_level <- gene_level_filter(wilms_CH_data, "wilms")

# Filter down to lymphoma data and filter to gene level
lymphoma_CH_data <- lymphoma_sample_filter(combined_CH_all)
lymphoma_CH_gene_level <- gene_level_filter(lymphoma_CH_data, "lymphoma")
```

Read in de Novo data and separate by disease

```
# Read in de novo data and generate new columns with variant type and class
combined_deNovo_data_vars <-
read_tsv_custom("PedFam_deNovo_cadd20_maf01.tsv") %>%
  add_column(variant_type = "De Novo", class = "SNP")

combined_deNovo_data_svs <-
read_tsv_custom("PedFam_deNovo_cadd20_maf01_svs.tsv") %>%
  add_column(variant_type = "De Novo", class = "SV")

combined_deNovo_data_dels <-
read_tsv_custom("PedFam_deNovo_cadd20_maf01_dels.tsv") %>%
  add_column(variant_type = "De Novo", class = "indel")

# Combine all imported data into a single dataframe
combined_deNovo_all <- combine_df(list(combined_deNovo_data_vars,
                                       combined_deNovo_data_svs,
                                       combined_deNovo_data_dels))

# Filter down to astrocytoma data and filter to gene level
astrocytoma_deNovo_data <- astrocytoma_sample_filter(combined_deNovo_all)
astrocytoma_deNovo_gene_level <- gene_level_filter(astrocytoma_deNovo_data,
                                                   "astrocytoma")

# Filter down to wilms data and filter to gene level
wilms_deNovo_data <- wilms_sample_filter(combined_deNovo_all)
wilms_deNovo_gene_level <- gene_level_filter(wilms_deNovo_data, "wilms")
```

```
# Filter down to lymphoma data and filter to gene level
lymphoma_deNovo_data <- lymphoma_sample_filter(combined_deNovo_all)
lymphoma_deNovo_gene_level <- gene_level_filter(lymphoma_deNovo_data,
"lymphoma")
```

Read in homozygous alternate data and separate by disease

```
# Read in homozygous alternate data and generate new columns with variant
type
# and class
combined_homAlt_data_vars <-
read_tsv_custom("PedFam_homAlt_cadd20_maf01.tsv") %>%
  add_column(variant_type = "Homozygous Alternate", class = "SNP")

combined_homAlt_data_svs <-
read_tsv_custom("PedFam_homAlt_cadd20_maf01_svs.tsv") %>%
  add_column(variant_type = "Homozygous Alternate", class = "SV")

combined_homAlt_data_dels <-
read_tsv_custom("PedFam_homAlt_cadd20_maf01_dels.tsv") %>%
  add_column(variant_type = "Homozygous Alternate", class = "indel")

# Combine all imported data into a single dataframe
combined_homAlt_all <- combine_df(list(combined_homAlt_data_vars,
                                       combined_homAlt_data_svs,
                                       combined_homAlt_data_dels))

# Filter down to astrocytoma data and filter to gene level
astrocytoma_homAlt_data <- astrocytoma_sample_filter(combined_homAlt_all)
astrocytoma_homAlt_gene_level <- gene_level_filter(astrocytoma_homAlt_data,
                                                   "astrocytoma")

# Filter down to wilms data and filter to gene level
wilms_homAlt_data <- wilms_sample_filter(combined_homAlt_all)
wilms_homAlt_gene_level <- gene_level_filter(wilms_homAlt_data, "wilms")

# Filter down to lymphoma data and filter to gene level
lymphoma_homAlt_data <- lymphoma_sample_filter(combined_homAlt_all)
lymphoma_homAlt_gene_level <- gene_level_filter(lymphoma_homAlt_data,
"lymphoma")
```

Read in heterozygous data and separate by disease

```
# Read in heterozygous data and generate new columns with variant type and
class
combined_het_data_vars <- read_tsv_custom("PedFam_het_cadd20_maf01.tsv") %>%
  add_column(variant_type = "Heterozygous", class = "SNP")

combined_het_data_svs <- read_tsv_custom("PedFam_het_cadd20_maf01_svs.tsv")
%>%
  add_column(variant_type = "Heterozygous", class = "SV")
```

```
combined_het_data_dels <- read_tsv_custom("PedFam_het_cadd20_maf01_dels.tsv")
%>%
  add_column(variant_type = "Heterozygous", class = "indel")

# Combine all imported data into a single dataframe
combined_het_all <- combine_df(list(combined_het_data_vars,
                                    combined_het_data_svs,
combined_het_data_dels))

# Filter down to astrocytoma data and filter to gene level
astrocytoma_het_data <- astrocytoma_sample_filter(combined_het_all)
astrocytoma_het_gene_level <- gene_level_filter(astrocytoma_het_data,
"astrocytoma")

# Filter down to wilms data and filter to gene level
wilms_het_data <- wilms_sample_filter(combined_het_all)
wilms_het_gene_level <- gene_level_filter(wilms_het_data, "wilms")

# Filter down to lymphoma data and filter to gene level
lymphoma_het_data <- lymphoma_sample_filter(combined_het_all)
lymphoma_het_gene_level <- gene_level_filter(lymphoma_het_data, "lymphoma")
```

Combine all data regardless of data type, and summarise at gene and variant levels

This code chunk was used to generate data for Table 1 in the manuscript and Supplementary Table 1.

```
#Combine all raw data
combined_data_all <- combine_df(list(combined_homAlt_all,
combined_deNovo_all,
                                     combined_het_all, combined_CH_all))
write_tsv(combined_data_all, "combined_data_all.tsv")
# Combine all data at gene level
gene_level_all <- combine_df(list(astrocytoma_CH_gene_level,
                                  astrocytoma_deNovo_gene_level,
                                  astrocytoma_homAlt_gene_level,
                                  astrocytoma_het_gene_level,
                                  lymphoma_CH_gene_level,
                                  lymphoma_deNovo_gene_level,
                                  lymphoma_het_gene_level,
                                  lymphoma_homAlt_gene_level,
                                  wilms_CH_gene_level,
wilms_deNovo_gene_level,
                                  wilms_het_gene_level,
wilms_homAlt_gene_level))

# Gene level summary (how many samples each potentially damaged gene has)
```

```r
gene_level_summary_all <- group_by(gene_level_all, disease, gene) %>%
  summarise(num_samples_with_damage = length(unique(sample)))

write_tsv(gene_level_summary_all, "gene_level_summary.tsv")

gene_level_summary_all
```

```
## # A tibble: 84 x 3
## # Groups:   disease [3]
##    disease     gene     num_samples_with_damage
##    <chr>       <chr>                      <int>
##  1 astrocytoma ACADS                          1
##  2 astrocytoma ACOXL                          4
##  3 astrocytoma ALG1L2                         1
##  4 astrocytoma BST1                           5
##  5 astrocytoma CBWD5                          3
##  6 astrocytoma CRNKL1                         3
##  7 astrocytoma DAPL1                          3
##  8 astrocytoma DUOXA1                         3
##  9 astrocytoma FAM8A1                         2
## 10 astrocytoma GOLGA6L1                       1
## # … with 74 more rows
```

```r
# Gene and sample level summary (how many genes have x number of samples with
a
# variant in that gene)
shared_genes_summary <- group_by(gene_level_summary_all, disease,
                                 num_samples_with_damage) %>%
  summarise(genes_with_num_damaged_genes = length(unique(gene)))

shared_genes_summary
```

```
## # A tibble: 14 x 3
## # Groups:   disease [3]
##    disease     num_samples_with_damage genes_with_num_damaged_genes
##    <chr>                         <int>                        <int>
##  1 astrocytoma                       1                           10
##  2 astrocytoma                       2                            5
##  3 astrocytoma                       3                            6
##  4 astrocytoma                       4                            6
##  5 astrocytoma                       5                            3
##  6 lymphoma                          1                           14
##  7 lymphoma                          2                            8
##  8 lymphoma                          3                            4
##  9 lymphoma                          4                            4
## 10 lymphoma                          5                            2
## 11 wilms                             1                           11
## 12 wilms                             2                            5
## 13 wilms                             3                            4
## 14 wilms                             4                            2
```

```r
# Find any shared variants
shared_variants <- shared_variant_filter(add_family_ID(combined_data_all))

# Summarize shared variants
shared_variants_summary <- group_by(shared_variants, family, shared_variants)
%>%
  summarise(total = n())

shared_variants_summary
```

```
## # A tibble: 14 x 3
## # Groups:   family [3]
##    family   shared_variants total
##    <chr>              <int> <int>
##  1 Family 1               1    12
##  2 Family 1               2     5
##  3 Family 1               3     7
##  4 Family 1               4     6
##  5 Family 1               5     3
##  6 Family 2               1    12
##  7 Family 2               2     5
##  8 Family 2               3     4
##  9 Family 2               4     2
## 10 Family 3               1    15
## 11 Family 3               2     8
## 12 Family 3               3     5
## 13 Family 3               4     5
## 14 Family 3               5     1
```

Use python to retain pertinent information from the 'combined_data_all.tsv' file and combine the chromosome and position columns

```python
sample_dict = {}
with open('combined_data_all.tsv') as input_file:
  header = input_file.readline()
  header_list = header.rstrip("\n").split("\t")
  variant_type_index = header_list.index("variant_type")
  sample_index = header_list.index("sample")
  start_index = header_list.index("start")
  class_index = header_list.index("class")
  chrom_index = header_list.index("chrom")
  for line in input_file:
    line_list = line.rstrip("\n").split("\t")
    variant_type = line_list[variant_type_index]
    if variant_type == "Heterozygous":
      variant_type = "Simple Heterozygous"
    sample = line_list[sample_index]
    start = line_list[start_index]
    chrom = line_list[chrom_index]
```

```
    variant_class = line_list[class_index]
    pos = chrom + ": " + start
    # Summarize variant class and type for each variant for each sample
    if sample not in sample_dict:
      sample_dict[sample] = {pos: {variant_class: [variant_type]}}
    elif sample in sample_dict and pos not in sample_dict[sample]:
      sample_dict[sample][pos] = {variant_class :[variant_type]}
    elif sample in sample_dict and pos in sample_dict[sample] \
      and variant_class not in sample_dict[sample][pos]:
      sample_dict[sample][pos][variant_class] = [variant_type]
    elif sample in sample_dict and pos in sample_dict[sample] \
      and variant_class in sample_dict[sample][pos] and variant_type \
      not in sample_dict[sample][pos][variant_class]:
      sample_dict[sample][pos][variant_class].append(variant_type)

with open('combined_data_updated.tsv', 'wt') as output_file:
  output_file.write("sample\tpos\tclass\tvariant_type\n")
  for sample, pos_dict in sample_dict.items():
    for pos, variant_class_dict in pos_dict.items():
      for variant_class, variant_list in variant_class_dict.items():
        variant_list.sort()
        # Retains the variant type, removes "de novo" from the type if found
        variant_string = variant_list[-1]
        # Retains "compound heterozygous" only, does not include more
specific
        # variant types contributing to the compound heterozygous variant
        if "Compound Heterozygous" in variant_string:
          variant_string = variant_list[0]
        output_file.write("{}\t{}\t{}\t{}\n".format(sample, pos,
variant_class, variant_string))
```

Generate a Figure showing a variant level summary that shows variant type and class

This code chunk was used to generate Figure 3 for the manuscript.

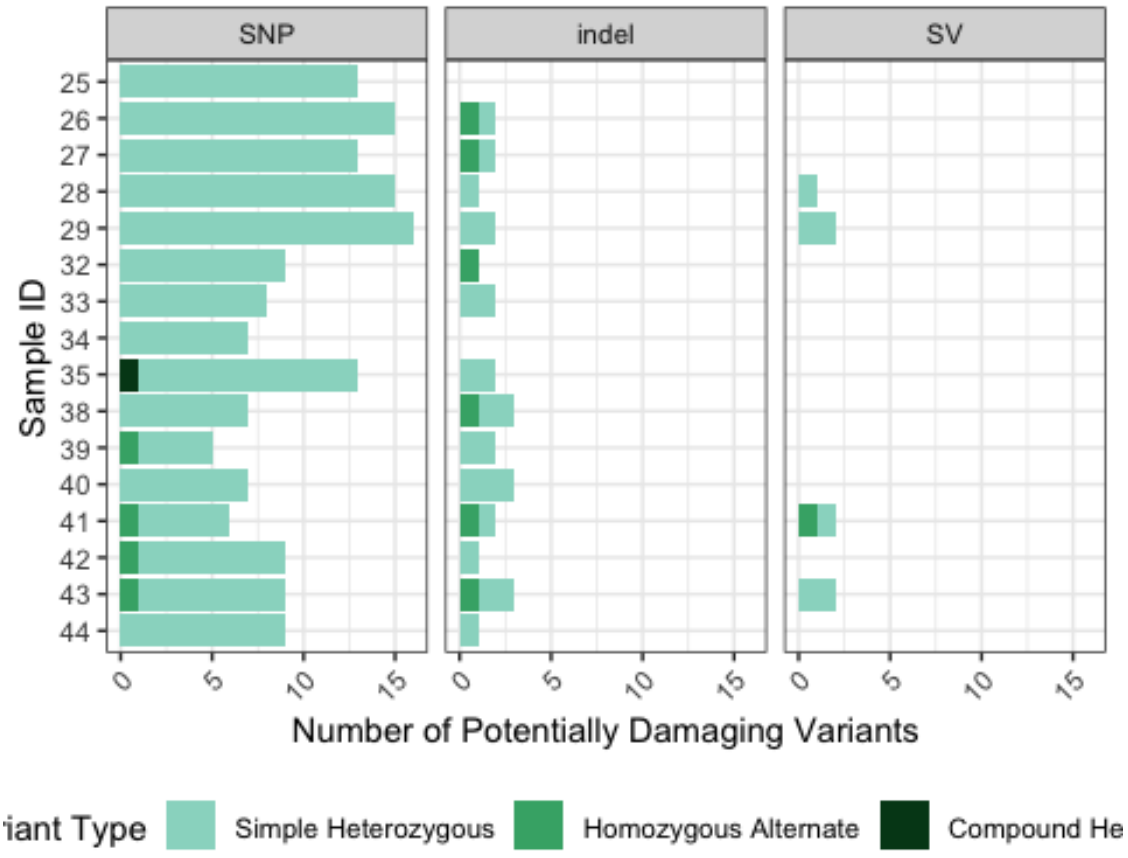
```
# Update sample names and add disease column
combined_data_updated <- read_tsv("combined_data_updated.tsv")
combined_data_updated <- mutate(combined_data_updated,
                                sample = str_replace(sample, "4478_SP_00",
""))
combined_data_updated <- add_disease_column(combined_data_updated)

# variant level summary (how many of each type of variant each sample has)
variant_level_summary_all <- variant_level_filter_2(combined_data_updated)
variant_level_summary_all <- add_disease_column(variant_level_summary_all)

# Change order that classes appear in graph
variant_level_summary_all$class <- factor(variant_level_summary_all$class,
```

```
levels = c("SNP", "indel", "SV"))
variant_level_summary_all$variant_type <-
factor(variant_level_summary_all$variant_type, levels = c("Simple
Heterozygous", "Homozygous Alternate", "Compound Heterozygous"))

# Generate figure
ggplot(variant_level_summary_all, aes(variants, sample, variant_type, class))
+
  geom_col(aes(variants, sample, fill = variant_type)) +
  facet_wrap(~class, scales = "fixed") +
  theme_bw() +
  theme(panel.background = element_rect(fill= "white"),
        axis.text.y = element_text(hjust = 1),
        axis.text.x = element_text(angle = 45, hjust = 1), legend.position =
"bottom") +
  labs(y="Sample ID", x="Number of Potentially Damaging Variants",
       fill="Variant Type") +
  scale_fill_manual(values = rev(c("#00441b", "#41ae76", "#99d8c9"))) +
  scale_y_discrete(limits = rev)
```




```
# Save figure
ggsave("Figure_3.png", plot = last_plot())
```

Use python to retain de novo variants from the 'combined_data_all.tsv' file and combine the chromosome and position columns

```
sample_dict = {}
with open('combined_data_all.tsv') as input_file:
  header = input_file.readline()
  header_list = header.rstrip("\n").split("\t")
  variant_type_index = header_list.index("variant_type")
  sample_index = header_list.index("sample")
  start_index = header_list.index("start")
  class_index = header_list.index("class")
  chrom_index = header_list.index("chrom")
  for line in input_file:
    line_list = line.rstrip("\n").split("\t")
    variant_type = line_list[variant_type_index]
    if variant_type == "Heterozygous":
      variant_type = "Simple Heterozygous"
    sample = line_list[sample_index]
    start = line_list[start_index]
    chrom = line_list[chrom_index]
    variant_class = line_list[class_index]
    pos = chrom + ": " + start
    # Summarize variant class and type for each variant for each sample
    if sample not in sample_dict:
      sample_dict[sample] = {pos: {variant_class: [variant_type]}}
    elif sample in sample_dict and pos not in sample_dict[sample]:
      sample_dict[sample][pos] = {variant_class :[variant_type]}
    elif sample in sample_dict and pos in sample_dict[sample] \
      and variant_class not in sample_dict[sample][pos]:
      sample_dict[sample][pos][variant_class] = [variant_type]
    elif sample in sample_dict and pos in sample_dict[sample] \
      and variant_class in sample_dict[sample][pos] and variant_type \
      not in sample_dict[sample][pos][variant_class]:
      sample_dict[sample][pos][variant_class].append(variant_type)

with open('combined_data_de_novo_updated.tsv', 'wt') as output_file:
  output_file.write("sample\tpos\tclass\tvariant_type\n")
  for sample, pos_dict in sample_dict.items():
    for pos, variant_class_dict in pos_dict.items():
      for variant_class, variant_list in variant_class_dict.items():
        variant_list.sort()
        variant_string = " ".join(variant_list)
        if "Compound Heterozygous" in variant_string:
          variant_string = variant_string.replace("Compound Heterozygous",
"")
        # outputs de novo variants. For compound heterozygous variants, only
the
        # variant types that contribute to the compound heterozygous variant
```

```
are output.
        if variant_string != "Compound Heterozygous;" and "De Novo" in
variant_string:
          output_file.write("{}\t{}\t{}\t{}\n".format(sample, pos,
variant_class, variant_string))
```

Generate a Figure showing a variant level summary that shows de novo variant type and class

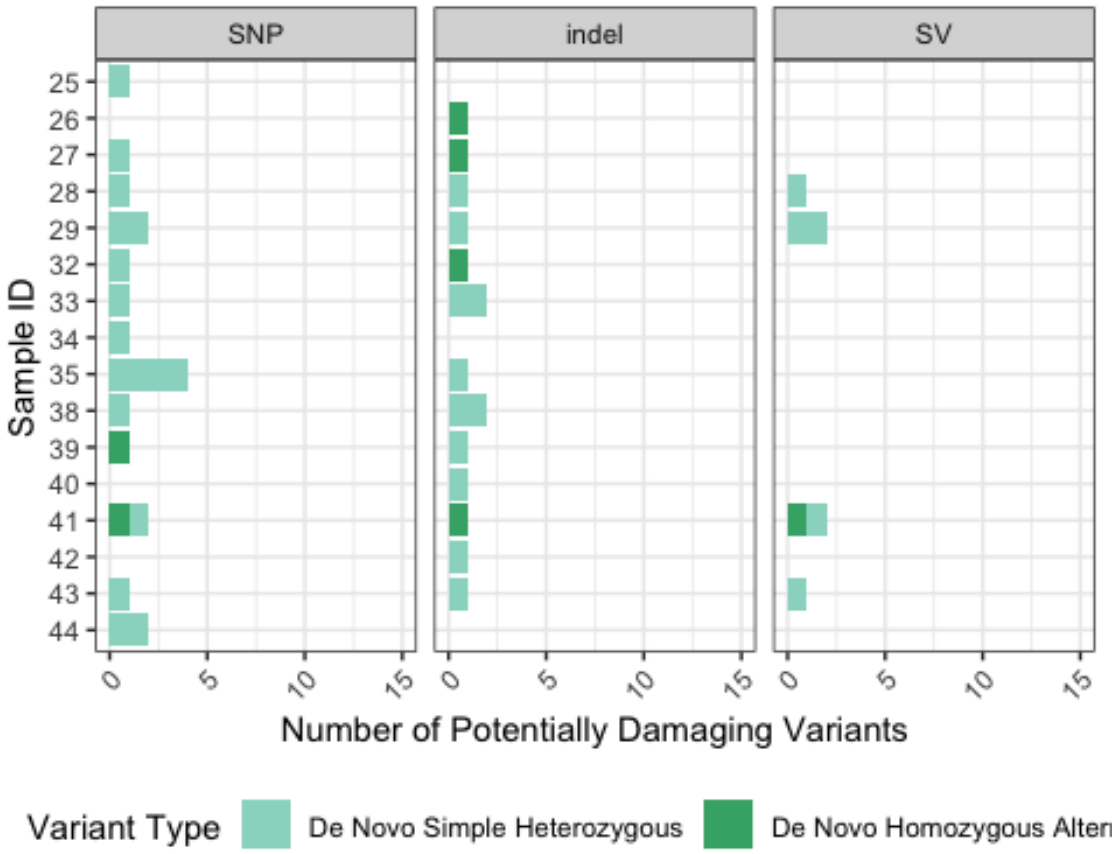
This code chunk was used to generate Figure 4 for the manuscript.

```
# Update sample names and add disease column
combined_data_de_novo_updated <-
read_tsv("combined_data_de_novo_updated.tsv")
combined_data_de_novo_updated <- mutate(combined_data_de_novo_updated,
                                sample = str_replace(sample, "4478_SP_00",
""))
combined_data_de_novo_updated <-
add_disease_column(combined_data_de_novo_updated)

# de novo variant level summary (how many of each type of variant each sample
has)
variant_level_summary_de_novo <-
variant_level_filter_2(combined_data_de_novo_updated)
variant_level_summary_de_novo <-
add_disease_column(variant_level_summary_de_novo)

# Change order that classes appear in graph
variant_level_summary_de_novo$class <-
factor(variant_level_summary_de_novo$class, levels = c("SNP", "indel", "SV"))
variant_level_summary_de_novo$variant_type <-
factor(variant_level_summary_de_novo$variant_type, levels = c("De Novo Simple
Heterozygous", "De Novo Homozygous Alternate"))

# Generate figure
ggplot(variant_level_summary_de_novo, aes(variants, sample, variant_type,
class)) +
  geom_col(aes(variants, sample, fill = variant_type)) +
  facet_wrap(~class, scales = "fixed") +
  theme_bw() +
  theme(panel.background = element_rect(fill= "white"),
        axis.text.y = element_text(hjust = 1),
        axis.text.x = element_text(angle = 45, hjust = 1), legend.position =
"bottom") +
  labs(y="Sample ID", x="Number of Potentially Damaging Variants",
       fill="Variant Type") +
  scale_fill_manual(values = rev(c("#41ae76", "#99d8c9"))) +
  scale_y_discrete(limits = rev) +
  scale_x_continuous(limits = c(0, 15))
```




```
# Save figure
ggsave("Figure_4.png", plot = last_plot())
```

Find genes that have 2 or less siblings with a variant

This code chunk provided a list of genes for each family that was used as input into *VarElect*.

```
# Astrocytoma  data
paste("astrocytoma: ", str_c(filter(gene_level_summary_all,
                                    disease == "astrocytoma" &
                                      num_samples_with_damage <= 2)$gene,
                             collapse = ","))

## [1] "astrocytoma:  ACADS,ALG1L2,FAM8A1,GOLGA6L1,GOLGA6L2,GUCA1C,HLA-
DRB1,KRT76,RP11-294C11.1,SIRPB1,SPAG11B,SPATA31A6,TTBK2,XIRP2,Y_RNA"

paste("Number of genes to input to VarEelct for astrocytoma: ",
      length(filter(gene_level_summary_all, disease == "astrocytoma" &
                      num_samples_with_damage <= 2)$gene))

## [1] "Number of genes to input to VarEelct for astrocytoma:  15"
```

```r
# Lymphoma data
paste("lymphoma: ", str_c(filter(gene_level_summary_all, disease ==
"lymphoma" &
                                    num_samples_with_damage <= 2)$gene,
                          collapse = ","))
```

```
## [1] "lymphoma:
ADGRA2,C1orf50,C2orf16,CCDC179,CSNK1A1,DCHS1,DNTTIP2,ENPEP,GLYATL1,ITGB4,KRTA
P10-12,LILRA2,RGPD3,RNU7-
167P,SIRPB1,SPAG11B,TNNT3,TRMT1,USP17L15,Y_RNA,ZNF826P,ZNF99"
```

```r
paste("Number of genes to input to VarEelct for lymphoma: ",
      length(filter(gene_level_summary_all, disease == "lymphoma" &
                      num_samples_with_damage <= 2)$gene))
```

```
## [1] "Number of genes to input to VarEelct for lymphoma:  22"
```

```r
# Wilms data
paste("wilms: ", str_c(filter(gene_level_summary_all, disease == "wilms" &
                                num_samples_with_damage <= 2)$gene,
                       collapse = ","))
```

```
## [1] "wilms:  ANKRD36C,CYP2A13,DPY19L3,FAM8A1,GOSR2,HLA-DRB1,HLA-
DRB5,PAH,PCK1,PCK2,PNPLA7,RP1L1,SIPA1L3,TRPM3,Y_RNA,ZAN"
```

```r
paste("Number of genes to input to VarEelct for wilms: ",
      length(filter(gene_level_summary_all, disease == "wilms" &
                      num_samples_with_damage <= 2)$gene))
```

```
## [1] "Number of genes to input to VarEelct for wilms:  16"
```

Figure how how many unique genes there were before and after sibling filtering

```r
#unique genes
paste("Across all families, there were ",
      length(unique(filter(gene_level_summary_all,
                           num_samples_with_damage <= 2)$gene)),
      " unique genes that had a potentially damaing variant in up to 2
siblings.")
```

```
## [1] "Across all families, there were  47  unique genes that had a
potentially damaing variant in up to 2 siblings."
```

```r
paste("Across all famlies, there were ",
      length(unique(gene_level_summary_all$gene)),
      " unique genes with potentially damaging variants.")
```

```
## [1] "Across all famlies, there were  72  unique genes with potentially
damaging variants."
```

Create a vector of known cancer genes using COSMIC

```
cosmic <- read_tsv("known_genes/Census_allThu Apr 16 21_44_49 2020.tsv")
cosmic_all <- cosmic$`Gene Symbol`
```

Check to see if any of the children had a potentially damaing variant in a known cancer gene

```
cancer_genes <- c()
for(gene in combined_data_all$gene){
  if (gene %in% cosmic_all & !gene %in% cancer_genes) {
    cancer_genes <- c(cancer_genes, gene)
  }
}
paste("The following genes are known cancer genes and were identified in at
least on child across all families: ",
      str_c(cancer_genes, collapse = ", "))
```

```
## [1] "The following genes are known cancer genes and were identified in at
least on child across all families:  RGPD3, CRNKL1"
```