Using VAAST to Identify Disease-Associated Variants in Next-Generation Sequencing Data

Brett Kennedy,[1,4] Zev Kronenberg,[1,4] Hao Hu,[2,4] Barry Moore,[1] Steven Flygare,[1] Martin G. Reese,[3] Lynn B. Jorde,[1] Mark Yandell,[1] and Chad Huff[2]

[1]Department of Human Genetics, University of Utah School of Medicine, Salt Lake City, Utah
[2]Department of Epidemiology, The University of Texas M.D. Anderson Cancer Center, Houston, Texas
[3]Omicia, Inc., Emeryville, California
[4]These authors collectively are the first authors of the unit

ABSTRACT

The VAAST pipeline is specifically designed to identify disease-associated alleles in next-generation sequencing data. In the protocols presented in this paper, we outline the best practices for variant prioritization using VAAST. Examples and test data are provided for case-control, small pedigree, and large pedigree analyses. These protocols will teach users the fundamentals of VAAST, VAAST 2.0, and pVAAST analyses. *Curr. Protoc. Hum. Genet.* 81:6.14.1-6.14.25. © 2014 by John Wiley & Sons, Inc.

Keywords: VAAST • rare-variant association test • variant classification • disease-gene identification • next-generation sequencing • genome-wide association studies • human disease • genomics • computational genomics • bioinformatics

INTRODUCTION

The Variant Annotation Analysis and Search Tool (VAAST) is a software suite for disease gene discovery from next-generation sequence (NGS) data. The central component of VAAST is a probabilistic disease-gene finder that combines amino acid substitution (AAS) and allele-frequency information to search for and prioritize genes harboring damaging alleles (Yandell et al., 2011). The latest VAAST release, 2.0, incorporates robust models of cross-species sequence conservation, which further improve the accuracy in differentiating between benign and disease-causing variation (Hu et al., 2013). The VAAST package also includes the Variant Selection Tool (VST) for filtering, combining, and manipulating large genomic datasets, and the Variant Annotation Tool (VAT) for detailed genomic variant annotation. VAAST is primarily designed for human disease studies (Rope et al., 2011; McElroy et al., 2013; Shirley et al., 2013), but has also been successfully applied to genotype-phenotype relationships in nonhuman organisms (Shapiro et al., 2013). VAAST is freely available for academic research, is actively supported through a dedicated mailing list, and is under continuous development to keep pace with the rapidly increasing complexity of next-generation sequence data analyses.

Here we present a series of VAAST workflows covering a broad range of disease-gene discovery applications. Basic Protocol 1 provides general variant-calling recommendations for NGS projects. We describe the preparation of NGS variant data for the VAAST pipeline using its Variant Annotation Tool (VAT) and Variant Selection Tool (VST). A support protocol describes how to obtain and install VAAST. In Basic Protocol 2, we provide a series of best-practice examples for using VAAST in case-control studies and small family studies. Basic Protocol 3 describes the use of VAAST with pedigrees.

Current Protocols in Human Genetics 6.14.1-6.14.25, April 2014
Published online April 2014 in Wiley Online Library (wileyonlinelibrary.com).
DOI: 10.1002/0471142905.hg0614s81
Copyright © 2014 John Wiley & Sons, Inc.

Additionally, we describe the use of pedigree-VAAST (pVAAST), which extends VAAST to support large, complex pedigrees, in Basic Protocol 4. Finally, an alternate protocol describes Omicia Opal, a commercial genomic medicine decision-support software platform (Coonrod et al., 2013), which improves the VAAST result interpretation and clinical reporting for personal patient NGS sequences by combining the powerful VAAST statistics with clinical knowledge databases. Each protocol includes command lines, example data, expected results, and interpretation of VAAST output. These protocols are designed to train users for successful real-world applications of VAAST and pVAAST.

BASIC PROTOCOL 1

VARIANT CALLING

The underlying goal of a VAAST analysis is to identify disease genes and disease-causing alleles using case and control NGS datasets. Because of the complexity and rapid development of next-generation sequencing technologies and analysis pipelines, systematic technical differences in variant calls between cases and controls can be a major source of false-positive results in any NGS case-control study. A number of steps can be performed to improve the quality of a variant-calling pipeline to reduce such technical artifacts.

Prior to Running VAAST

For instructions on downloading and installing VAAST, see Support Protocol.

Alignment

Matching the read aligner and alignment parameters for all samples (both case and control) will help to avoid introducing alignment-specific artifacts. A series of alignment polishing steps have become standard practice and should be implemented to improve variant quality and reduce technical artifacts. These steps include removal of duplicate reads, local realignment of insertions and deletions, and base-quality score recalibration (Li et al., 2009; McKenna et al., 2010; DePristo et al., 2011) .

Variant-calling

Several high-quality variant-calling tools are freely available, and a comparison of these tools has been reviewed elsewhere (Nielsen et al., 2011; Pabinger et al., 2013; O'Rawe et al., 2013). Popular tools include GATK UnifiedGenotyper, SAMtools, and Atlas2 (Li et al., 2009; McKenna et al., 2010; DePristo et al., 2011; Challis et al., 2012). All variant callers make both false-positive and false-negative errors in variant-calling. However, the error profile produced by different variant callers (or even the same variant caller with different parameters or different versions) can be quite different. If cases and controls are not called with *exactly* the same variant-calling tool (including software version and run-time parameters), these differences in systematic error can manifest as false-positive results in VAAST.

Joint calling and missing genotypes

Current versions of most variant-calling tools (Li and Homer, 2010; DePristo et al., 2011; Wei et al., 2011) allow multiple samples to be jointly called, i.e., processed simultaneously. Joint variant-calling has two significant advantages. First, the variant-calling algorithm considers the alignments of all samples simultaneously to estimate the probability that a given locus is variable in the population, resulting in more accurate variant calls for each individual sample (McKenna et al., 2010; DePristo et al., 2011). Second, joint variant callers such as GATK UnifiedGenotyper (McKenna et al., 2010; DePristo et al., 2011) provide missing genotypes (i.e., 'no calls'). By default, most variant callers will not produce a variant call for missing genotypes. Thus, homozygous reference sites are indistinguishable from sites where no genotype information is available

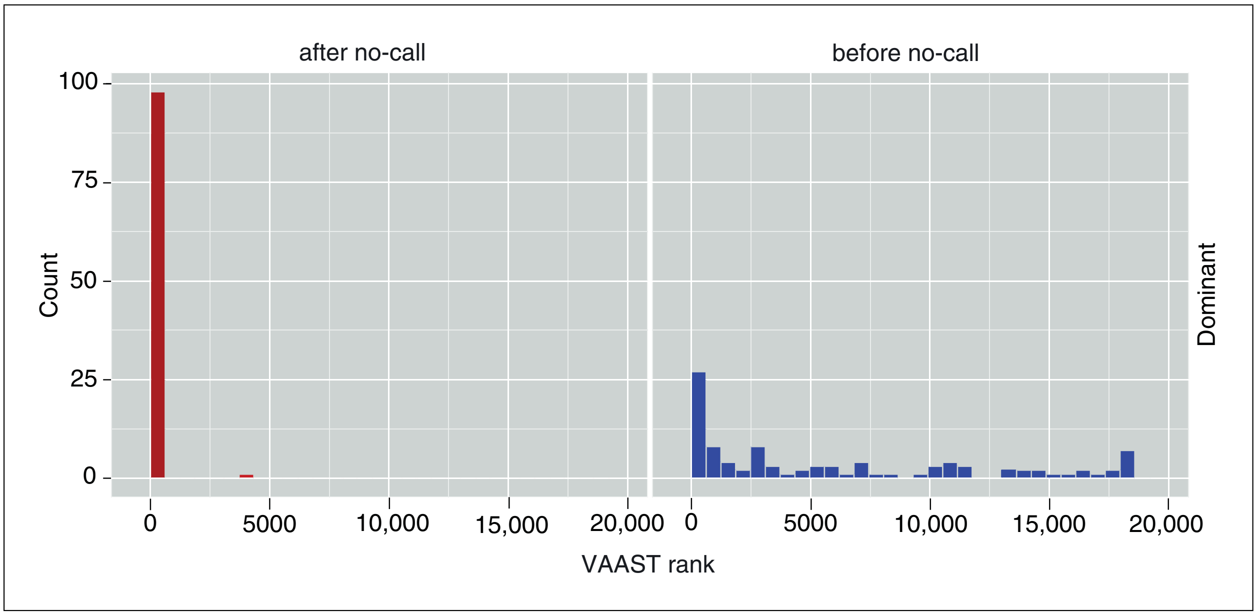


Figure 6.14.1 Accounting for missing genotypes improves the power of VAAST. The VAAST ranks of "doped" genes are shown on the *x* axis. The number of trials with a specific rank is shown on the *y* axis. The data presented were generated from "doping" experiments under a dominant inheritance pattern. Known HGMD disease-associated alleles were "doped" into three target exomes. In the "before-nocall" batch (right panel), this subset of HGMD alleles were incorrectly coded as the reference allele. This type of error can occur in empirical datasets due to missing coverage. In the "after no-call" batch (left panel), the same subset of HGMD variants were coded as no-call (^) rather than the reference allele. This simulates joint-variant-calling and demonstrates the importance of accounting for missing genotypes.

(for example, due poor sequence quality or low depth of coverage). When both cases and controls are processed through these variant-calling tools simultaneously, all variant sites in all samples are consistently called for missing genotypes. VAAST is specifically designed to make use of missing genotype information, which substantially improves the signal-to-noise ratio in disease-gene searches (see Fig. 6.14.1). A lack of missing genotype data in cases or controls can be a significant source of error for all downstream analyses and interpretations. In addition, sites with an excess of missing genotypes typically have higher error rates, and filtering sites with missing genotype rates of 10% or greater can further reduce false-positive rates. Such filtering steps can either be performed prior to VAAST or by using the variant_mask option in VAAST (see below). Note that some variant-calling algorithms can generate missing genotype calls without joint variant-calling, such as those employed by Complete Genomics and earlier versions of GATK (Drmanac et al., 2010; McKenna et al., 2010; DePristo et al., 2011).

Variant filtering and quality score recalibration

Most variant callers assign a variant quality score to each variant to indicate the probability that the variant (or genotype) was incorrectly called. Numerous strategies have been developed to mitigate false-positive variants by filtering on quality scores or other variant metrics. More recently, algorithms have been developed that create a model of true-positive variants trained on accurate variant calls (using HapMap3 and other highly validated variant sites). For example, the variant-quality method implemented in the GATK VariantRecalibrator tool can greatly increase the accuracy of disease-gene searches (Fig. 6.14.2).

GVF Conversion

The first step in a VAAST analysis is to ensure that the variant data are in Genome Variation Format (GVF; Eilbeck et al., 2005; Reese et al., 2010). For variant call data in

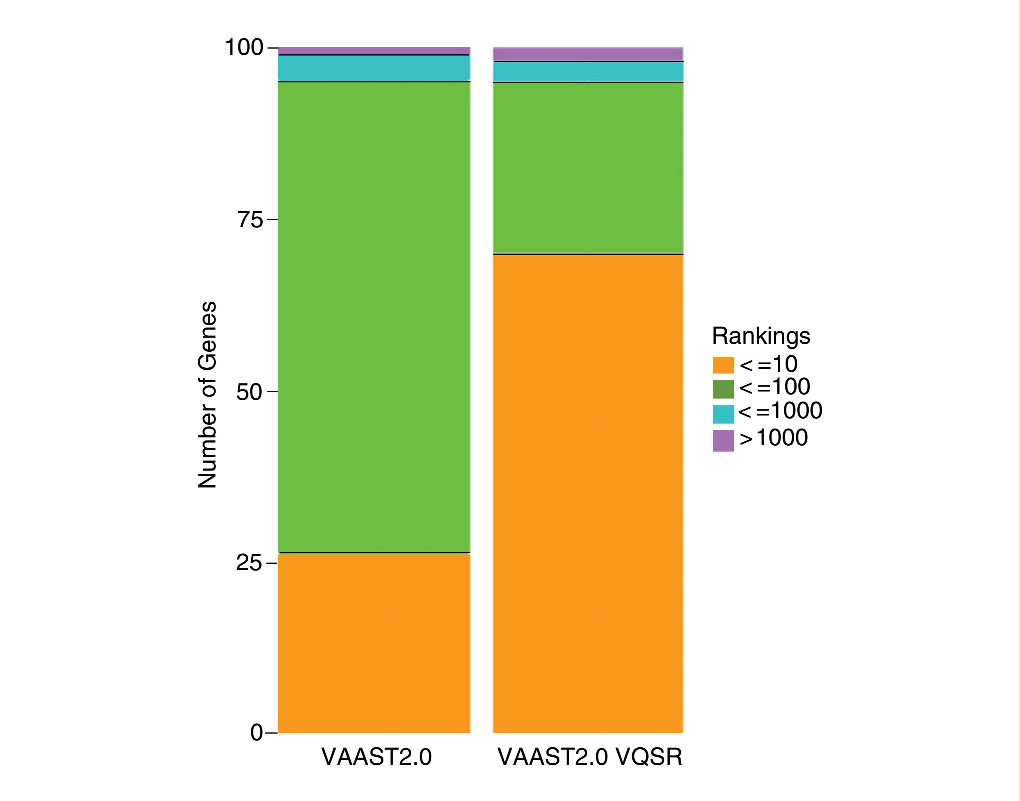


Figure 6.14.2 Filtering on VQSR scores can be used to improve VAAST's accuracy. Separate VAAST searches were performed for 100 HGMD alleles (including indels) "doped" into a target consisting of three unrelated individuals against a background of 200 individuals from the 1000 Genomes data. The graph shows the number of genes with known disease-causing variants from HGMD that were ranked by a VAAST search in one of four categories: top 10 (<=10), top 100 (<=100), top 1000 (<= 1000), or more than 1000 (>1000). The VQSR procedure dramatically increases the number of genes with variants from HGMD in the top 10 from about 30% to more than 70%. This result highlights the importance of variant call accuracy when searching personal genome data.

VCF format, this can easily be converted using the `vaast_converter` tool found in `VAAST/bin/vaast_tools/vaast_converter`. GVF is a file format developed by the Sequence Ontology group for use in describing sequence variants that provides a computationally robust ontology-controlled format for deeply annotating the effect of those variants on sequence features. The command below creates a separate GVF file for each individual in the VCF file:

```
VAAST/bin/vaast_tools/vaast_converter --build hg19
  name.vcf
```

The GVF conversion step has been completed for the example files used below.

Variant Annotation

VAT annotates the impact of variants on genomic features based on the terms and relationships described in the Sequence Ontology (SO; Eilbeck et al., 2005). VAT outputs its annotations in GVF Format (Reese et al., 2010), which is a sequence-variant annotation format maintained by the SO. The format is based on GFF3, and is compatible with other tools that parse or visualize GFF3 files.

VAT requires three files as input: (1) a GFF3 file containing sequence features (gene models and possibly other features); (2) a FASTA file containing the genome's sequence;

and (3) a GVF file containing the sequence alterations that will be annotated. VAT's command-line options allow control over extended annotation features and memory use. VAT annotates the functional impact of each sequence alteration (variant) in the input GVF file when it overlaps a sequence feature in the provided GFF3 file. These annotations are given with a Variant_effect attribute added to the final column of the GVF line. The annotated GVF line with the added Variant_effect attribute provides details of the impact of the sequence alteration on sequence feature in three parts:

sequence_alteration : Each variant in a GVF file is described in column three by the SO term sequence_alteration (SO:0001059) or one of its children. These terms describe the actual change in the genomic sequence. VAT annotates the impact of the following sequence_alterations: SNV, insertion, deletion, MNP (multi-nucleotide polymorphism), and complex_substitution.

sequence_feature: When a sequence alteration overlaps a sequence feature, the Variant_effect attribute describes the feature using the SO term sequence_feature (SO:0000110) or one of its children. These terms describe the kind of sequence feature that is impacted by the sequence alteration. VAT can annotate the impact of variants on any feature in the input GFF3 file that is a child of a sequence_feature. Common sequence_feature examples include gene, exon, splice_acceptor, splice_donor, 5_prime_UTR, and 3_prime_UTR. In addition to sequence features annotated in the GFF3 file, VAT will infer additional sequence features based on the structure of gene models such as introns, splice sites, and initiator/termination codons.

sequence_variant: When a sequence alteration impacts a sequence feature, a variant of that sequence feature is created. VAT uses the term sequence_variant (SO:0001060) or one of its children to describe the functional effect of a sequence_alteration on a sequence_feature. VAT annotates sequence alterations with terms that describe the impact of coding variants with common terms such as stop_gained, missense_variant, and splice_acceptor_variant. In addition to the specific impacts of common sequence alterations on gene models, VAT will annotate the overlap of any sequence_alteration with any sequence feature in the GVF and GFF3 files with the general term sequence_variant.

Output and error messages are written to STDOUT and STDERR, respectively. A simple example of a VAT command line with the required options is:

```
VAT --features genes.gff3 --fasta assembly.fasta
  variants.gvf > variants.vat.gvf
```

If memory is a limitation, the chunk flag will break up sequences into *N* bp segments:

```
VAT --features genes.gff3 --chunk 50000 --fasta
  assembly.fasta variants.gvf > variants.vat.gvf
```

With the --extended_gvf option, VAT will add additional information to many Variant_effect attributes that provide more detail about a particular sequence_variant, such as the reference and variant amino acids for a missense_variant. A complete description of the SO terms and GVF format are described on the SO Web site: *http://www.sequenceontology.org*.

The VAAST package includes all files necessary to annotate variants in the human genome in the context of NCBI builds 36 (hg18) and 37 (hg19). The genome build can be provided to VAT in the incoming GVF file (added by vaast_converter) or can be described by the user on the command line using the --build flag.

Variant Selection

VST condenses variant files from multiple individuals into a single condenser (CDR) file. VST can also apply set operations across the individual input GVF files. These operations include union (U), intersection (I), left relative complement (C), and symmetric difference (D). VST also includes a shared (S) operation that specifies a cut-off to an intersection style operator to exclude or include variants based upon the number of individuals that share them.

Variant filtering with VST imposes strict assumptions about the presence or absence of variants, and care should be taken to avoid errors introduced by missing data. If missing genotypes are included in the GVF files, VST will utilize this information in filtering variants. For example, the intersection of a variant where one individual has a missing genotype will be included in the output CDR file. In contrast, the symmetric difference of a variant where the complemented individual has a variant call but another individual has a missing genotype will not be included. The `--genotype` flag will constrain set operations to allow variants to match only if they share the same genotype.

Set operations are passed to VST with the `--ops (or -o)` flag in the form of a quoted string. Individual GVF files are specified within the set operation as integers based on the order of the files on the command line. The first GVF file is file id 0, the second is file 1 and so on. A simple example of a VST set operation shown below uses GVF files in the case-control example data and creates a CDR file with the union of the variants from all three individuals:

```
VST --build hg19 -o 'U(0,1,2)' HG00096.gvf HG00110.gvf
  HG00246.gvf > union.cdr
```

The resulting `union.cdr` file contains 412,700 variants. An `--ops` argument that selects only variants common to all individuals would use the ‘I’ (intersection) set operation:

```
VST --build hg19 -o 'I(0,1,2)' HG00096.gvf HG00110.gvf
  HG00246.gvf > intersection.cdr
```

The resulting intersection.cdr file contains only 138,627 variants.

VST needs to know the length of the sequences in the assembly. This information may have been included in the GVF files by vaast_converter or VAT upstream of VST; however, it can be provided by VST for the human genome using the `--build` flag. While VAAST was developed for use with the human genome, it can be applied to an annotated genome assembly from any organism. Non-human data or unique human builds can be specified to VST with a three-column (seqid, start, end) build file and passed to VST with the `--build` flag. Example entries from a build file are:

```
#seqid       start  length
Scaffold01   0      140000
Scaffold02   0      20000
Scaffold03   0      34500
```

More advanced VST operations are described in the VAAST Quick Start Guide and in “VAAST with Small Pedigrees” section below.

Direct VCF to CDR Conversion

For users who have variant call data in VCF format and want to convert it to CDR format with only union operations in VST, we also provide a wrapper script (`vcf2cdr.pl`) that takes VCF file(s) as input, performs intermediate conversion steps (vaast_converter, VAT and VST), and produces CDR file(s) as the end output. This script requires four input files: (1) the VCF file(s), specified by the `--vcf` option; (2) the genome-sequence FASTA file, specified by the `--fasta` option; (3) the genome annotation GFF3 file, specified by `--gff3` option; and (4) a three-column (individual ID, cohort name, and sex), tab-delimited "info" file describing which individuals to include in each output CDR file, specified by `--info` option. The individual IDs in this file should match the IDs specified in the VCF file. An example is:

```
INDI1   cohort1   male
INDI2   cohort2   male
INDI3   cohort1   female
```

This info file instructs the script to create two CDR files: `cohort1.cdr` and `cohort2.cdr`. The former contains two genomes: "INDI1" and "INDI3"; the latter contains one genome: "INDI2".

An example command line is:

```
VAAST/bin/vaast_tools/vcf2cdr.pl --vcf vcf2cdr_test.vcf
  --output test --build hg19 --fasta hg19_chr16.fa
  --gff3 refGene_hg19.gff3 --info info.txt --cpus 3
```

In this command, `--output` specifies the output folder name; `--build` specifies the genome build, which can be either hg18 or hg19; `--cpus` parallelize the VAT and VST steps using three CPUs. After the execution, the final output can be found in the folder named "`test-step4`, which contains one CDR file for each cohort.

SUPPORT PROTOCOL

OBTAINING AND INSTALLING VAAST

VAAST is free for academic research use (commercial licenses are also available). The software can be downloaded from the VAAST Web site: *http://www.yandell-lab.org/software/vaast.html*.

After obtaining the VAAST tarball (`VAAST_Code_Current.tar.gz`), the program can be installed in a basic development environment (Linux or OS X) with the following command lines:

```
perl Build.PL
sudo ./Build installdeps
./Build test
./Build install
```

Additional information can be found in the included `INSTALL` document. Tutorials and examples are located in the `docs` directory of the VAAST package. These documents are also available for download on the VAAST Web site. For example, the VAAST Quick Start Guide provides several detailed examples using data included in the software distribution. The VAAST User's Guide provides detailed documentation for all software in the VAAST package, including a description of file formats, command-line options, and error codes. The VAAST users mailing list provides a dedicated forum for addressing problems or questions that are not covered in the documentation. It is linked from the VAAST home page above or can be accessed with the following URL: *http://yandell-lab.org/mailman/listinfo/vaast-user_yandell-lab.org*.

BASIC PROTOCOL 2

USING VAAST WITH CASE-CONTROL DATA

The case-control test dataset used in the examples below consists of exome data from 266 individuals of Northern European ancestry from the 1000 Genomes Project, with 10 cases (target) and 256 controls (background) (Abecasis et al., 2012). Two known disease-causing variants associated with frontotemporal lobar degeneration have been “doped” (Online Mendelian Inheritance in Man, OMIM, 2013) into the cases as heterozygous variants (rs63750653 and rs63750944). Each of the 10 individuals in the cases has one of two damaging variants. Both variants cause missense variants in the gene *CHMP2B*. The distributed test dataset is in GVF format and available for download at *http://www.yandell-lab.org/software/VAAST/data/hg19/CurrentProtocols*.

The first step in this example is variant annotation requiring a feature file (GFF3) and a reference FASTA sequence. In this protocol we are using a GFF3 (`refGene_hg19_chr3.gff3`) generated from the refGene table (Karolchik et al., 2004) and a FASTA file of the hg19 genome assembly (`ucsc.hg19.fasta`) (Lander et al., 2001). Both files contain data for human chromosome 3. Below is an example command line for annotating a single GVF file from the example data:

```
VAT -f refGene_hg19_chr3.gff3 -a ucsc.hg19.chr3.fasta
  HG00096.gvf > HG00096.vat.gvf
```

The annotated GVF files are next condensed into a CDR file. This is achieved using VST as demonstrated in the ‘Variant Selection’ section above. Here we take the union of the 10 affected individuals (i.e., all variants from all individuals):

```
VST --build hg19 -o 'U(0..9)' *.vat.gvf >
  10cases_union.cdr
```

The VAAST search tool uses the CDR files generated in the prior step. Below is a basic VAAST command line that specifies the output file (`-o`), the scoring method (`-m lrt`), and the number of permutations (`--gp 1e7`). By default, VAAST will report its progress to the terminal (STDERR). Temporary files will be written while VAAST is running that allow VAAST to restart if interrupted (using the `--restart` option). When VAAST successfully completes a run, temporary files are removed and two reports are generated. These two output files have the file extensions `.simple` and `.vaast`. The inputs to this command line are the reference gff3, the background CDR and the target CDR:

```
VAAST -m lrt --gp 1e7 -o 10cases_output
  refGene_hg19_chr3.gff3 1KG_chr3_Background.cdr
  10cases_union.cdr
```

The first five lines of the resulting `10cases_output.simple` file are below:

```
RANK   Gene      p-value   p-value-ci             Score
1      CHMP2B    0.0001    2.43e-18,0.000369      68.595
2      GTF2E1    0.0011    0.00048,0.00184        14.364
3      EHHADH    0.0039    0.00269,0.00522        8.87
4      DBR1      0.0102    0.00768,0.0128         4.885
```

The six columns for the `.simple` report are: rank, gene, *p*-value, confidence interval, gene score, and a variant info field. Only the first five columns are shown here for illustration purposes. For more information about the `.simple` report, see the “Interpreting Results” section, below. *CHMP2B* is the top ranking feature in the `10cases_output.simple` output above. The confidence interval for *CHMP2B* ranges from 2.4×10^{-18} to 0.000369, indicating that further permutation is necessary to accurately estimate the *p*-value. However, the confidence intervals do not overlap between the first and second ranking gene, indicating that the rank order is stable and is not likely to change with additional permutation (see the “Specifying the number of permutations” section below).

USING VAAST WITH PEDIGREES

BASIC PROTOCOL 3

VAAST provides a number of ways to explore genomic datasets from families and related individuals. We have greatly expanded this support with the addition of pVAAST, which applies probabilistic inheritance models and supports large pedigrees for both rare Mendelian and common, complex diseases (Hu et al., submitted). However, when the family size is small and complete penetrance within the pedigree can be safely assumed, VST or the VAAST `--trio` option can be used to filter variants that do not match the inheritance model. See Figure 6.14.3 for common VST set operations used in trio and quartet analyses. Three examples are included with this protocol, with recessive, dominant, and de novo mutation inheritance patterns; these files can be found at *http://www.yandell-lab.org/software/VAAST/data/hg19/CurrentProtocols*. For each example, a damaging variant in the *CHMPB2* gene has been doped according to the inheritance pattern. Each example is based on a sample trio dataset consisting of three individuals from the 1000 Genomes Project (Abecasis et al., 2012): two parents, NA12891 and NA12892, and their daughter, NA12878. The example data are organized as CDR files rather than the GVFs specified for the case-control analysis above. For the recessive example, two CDR files are provided, one for the parents and one for the affected offspring.

Recessive inheritance and the `--trio` *option*

The `--trio (-t)` option is an alternative to VST filtering that is specifically designed for recessive diseases in trios with unaffected parents and an affected offspring. This option filters variants that are found in the affected children’s genomes but not found in their parents (and thus should not be used when attempting to find de novo mutations). In addition, when complete penetrance is specified (`-pnt c`), the `--trio` option will filter out all variants for which the parents are homozygous or compound heterozygous

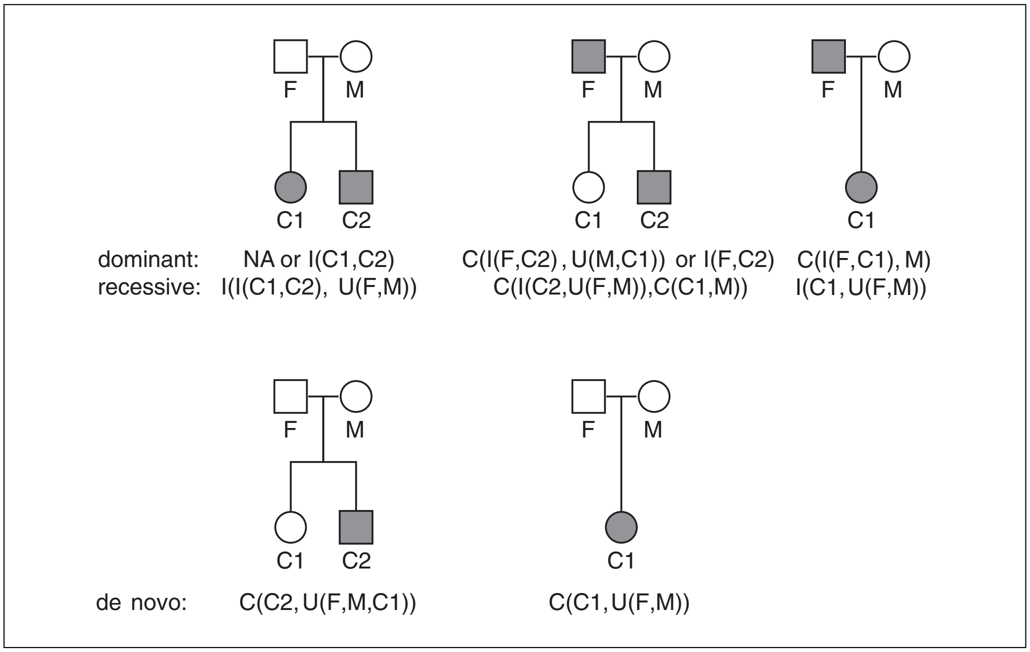


Figure 6.14.3 Example of VST set operations for small pedigrees. F,M,C represent father, mother, and child respectively.

under a recessive model. Below is an example of a complete VAAST command line using the trio option:

```
~/projects/VAAST/bin/VAAST -m lrt -iht r -pnt c --trio
  Parents_Rec.chr3.cdr -o Trio_Rec -d 1e6 -p 40
  refGene_hg19_chr3.gff3 1KG_chr3_Background.cdr
  Child_Rec_Chr3.cdr
```

The first five lines of the resulting Trio_Rec.simple file are below:

```
RANK    Gene       p-value     p-value-ci           Score
1       CHMP2B     3.89E-03    0.00389,0.00389      23.745
2       FGD5       0.0465      0.0315,0.0632        6.269
3       POLQ       0.0536      0.0374,0.0715        4.644
4       BCL6       1           1,1                  0
```

In this example, *CHMP2B* is the highest-ranking gene on chromosome 3, with a *p*-value that is an order of magnitude less than that of the second-highest ranking gene.

Small pedigree and dominant inheritance

The dominant trio example represents a scenario in which the offspring and one parent one parents are affected. The example CDR file was filtered with VST using the following command line:

```
VAAST/bin/VST -o 'C(I(0,1),2)' NA12878.vat.gvf
  NA12891.vat.gvf NA12892.vat.gvf > trio_dom.cdr
```

The resulting `Trio_IntComp.cdr` file contains only those variants that are absent in the unaffected parent and shared between the affected parent and offspring:

```
~/projects/VAAST/bin/VAAST -m lrt -iht d -o Trio_Dom -d
  1e6 -p 40 refGene_hg19_chr3.gff3
  1KG_chr3_Background.cdr trio_dom_chr3.cdr
```

Output from the `Trio_Dom.simple` file is below:

```
RANK   Gene      p-value    p-value-ci            Score
1      CHMP2B    3.12E-05   3.02e-05,3.39e-05     18.216
2      FAM43A    0.000106   7.96e-05,0.000135     15.388
3      DGKG      0.000133   8.98e-05,0.000181     15.603
4      EOMES     0.000152   0.000105,0.000205     9.859
```

Again, *CHMP2B* is the highest-ranking gene on chromosome 3, but in this case the confidence intervals and *p*-values for the top-scoring variants are somewhat close. When only a small family or a number of cases are available, VAAST will sometimes identify multiple genes with nearly equivalent statistical evidence. In these situations, the top-ranking genes can be further prioritized by combining the VAAST analysis with prior information about pathways or genes that have been associated with the disease. Tools such as Phenomizer (*http://compbio.charite.de/phenomizer/*) and the KEGG pathway database (*http://www.genome.jp/kegg/*) can be used to automate this process by creating a candidate list of genes potentially involved with the phenotype of interest.

Small pedigree and de novo mutations

The number of de novo mutation candidates in a parent-offspring trio depends heavily on the sequencing error rate. There are on average approximately 70 true positive new mutations in a genome relative to the parents, but the rate of Mendelian Inheritance Errors (MIEs) from NGS data can exceed the true de novo mutation rate by three orders of magnitude (Roach et al., 2010). Joint-calling (Li et al., 2009; McKenna et al., 2010; DePristo et al., 2011) can greatly reduce the rate of false-positive MIEs, but even with joint calling, the rate of MIEs typically greatly exceeds the true de novo mutation rate. VST can be used to identify MIEs that are consistent with the pattern of a de novo mutation. The example CDR was generated through the following VST command:

```
/VAAST/bin/VST --o 'C(0,U(1,2))' child.vat.gvf
  parent1.vat.gvf parent.vat.gvf > trio_denovo.cdr
```

This VST command limits the variants in the CDR file to only those that are unique to the child relative to the parents by taking the complement of the child versus the union of the parents. The corresponding VAAST run is as follows:

```
VAAST -m lrt -iht d -pnt c -o Trio_Denovo -d 1e6 -p 40
  refGene_hg19_chr3.gff3 1KG_chr3_Background.cdr
  Trio_Denovo_CI_chr3.cdr
```

Combining the complete penetrance flag with the dominant inheritance model flag will exclude any variant present in the background. This is an appropriate filter for most scenarios, but may be too stringent for very large background files or for variants with incomplete penetrance. The first five lines of the `Trio_Denovo.simple` file are below:

RANK	Gene	p-value	p-value-ci	Score
1	CHMP2B	0.00389	0.00389,0.00389	9.267
2	DVL3	0.00797	0.00687,0.00915	10.85
3	BCL6	1	1,1	0
4	CCDC71	1	1,1	0

Once more, *CHMP2B* is the highest-ranking gene on chromosome 3, and only one other gene is scored on the chromosome. This is a typical result for a parent-offspring trio with joint-calling and missing genotype information, especially when looking at only a single chromosome.

VST and VAAST with large pedigrees

VST and VAAST can also be used to analyze large pedigrees. VST can identify the intersection of affected individuals while subtracting the variants found in unaffected relatives. The resulting CDR file can then be analyzed in VAAST. This procedure can greatly reduce the number of candidate variants but is *not* robust to genotyping errors and requires perfect adherence to the specified inheritance pattern. For these reasons, we recommend pVAAST (Basic Protocol 4) for all large pedigrees.

USING pVAAST WITH PEDIGREE DATA

BASIC PROTOCOL 4

pVAAST supports a wide range of familial sequencing studies, from monogenic, Mendelian diseases in a single small family to highly polygenic, common diseases involving hundreds of large families. pVAAST retains all the functionality of VAAST, but in addition integrates linkage LOD scores into the CLRT framework of VAAST. pVAAST calculates statistical significance using a combination of permutation and gene-drop simulation (Hu et al., 2014) to account for both the family structure and the observed pattern of variation in cases and controls.

Basic Analysis

In addition to the input files used by VAAST, pVAAST requires a single control file and a pedigree file for each family. The pedigree structure is represented in PED format (Purcell et al., 2007). The individual ID in the PED file must match the `## FILE-INDEX` ID for the same individual in the corresponding CDR file, which by default is the GVF file name used by VST. All pedigree-related files and pVAAST parameters are specified in the control file. Control file templates are located in the `data/pvaast/` directory of the VAAST installation. The basic parameters are described below.

pVAAST control file arguments

`input_ped_cdr_files` : This argument specifies the input PED and CDR file pairs for each pedigree, separated by white space.

`pedigree_representatives`: This option designates an affected individual (by ID) to represent each pedigree in the case-control analysis. If more than one family is present, then the IDs are separated by white space. The designated individual must have the disease-causal variant; otherwise pVAAST will not score the causal mutation (see below).

`unknown_representatives`: In most scenarios, preselecting a pedigree representative from each family increases power by reducing the number of variants that are tested in each family. However, in some scenarios, such as a single large family with a complex genetic disease, the preferred option may be for pVAAST to score all variants in every pedigree member before selecting the highest scoring variant in each pedigree. This is done by setting the `unknown_representatives` option to “yes.”

`max_prevalence_filter`: An upper threshold for the disease prevalence in the general population. pVAAST performs a grid search over a wide range of MAF and penetrance parameters to maximize the LOD score at each variant site. This option modifies the search space to eliminate parameters that are implausible given the disease prevalence. Note that this option only affects the LOD score calculation, and it acts independently of the `--rate` option, which affects the CLRT score calculation. As a result, users need to set these two parameters separately.

`additional_cases`: In addition to pedigrees, pVAAST can also incorporate unrelated cases as part of the case-control portion of the analysis. The unrelated cases should be merged into a single CDR file with the VST union operation, and the file name should be provided here (rather than the CDR files provided in `input_ped_cdr_files` option).

`inheritance_model`: This option should be set to either “dominant” or “recessive.”

For example, to identify a disease gene in two pedigrees that fit a recessive pattern of inheritance, the control template file from the VAAST repository could be copied to a local file `recessive_test.ctl`, and then modified to include the following options:

```
input_ped_cdr_files: recessive1.ped recessive1.cdr
  recessive2.ped recessive2.cdr
pedigree_representatives: A02_1 B02_1
unknown_representatives: no
inheritance_model: recessive
```

The remaining options may be left unchanged. Here `A02_1` and `B02_2` are two affected and sequenced individuals from the two pedigrees: recesive1 and recessive2. The pVAAST command line would be:

```
VAAST -m pvaast -o recessive_test -d 1e5 -p 20
  -pv_control recessive_test.ctl -k refGene_hg19.gff3
  background.cdr
```

Note that the command parameters of pVAAST are almost identical to VAAST except for (1) -m pvaast option, which indicates the pVAAST algorithm should be used; (2) --pv_control option, which provides the path to the control file; and (3) the target CDR file(s) are specified in the control file rather than the command line.

simulate_genotyping_error: Mendelian inheritance errors in the pedigree result from either de novo mutation or genotyping errors. To account for these errors, set the genotyping_error_rate and simulate_genotype_error to "yes." If de novo mutations are of interest, no additional parameters need to be activated. The genotyping_error_rate can be estimated from the sequence data as long as at least one offspring and both parents are sequenced from a parent-offspring trio using the estimate_genotype_error_rate.pl script, which requires the pv_control file and the background CDR:

```
estimate_genotype_error_rate.pl recessive_test.ctl
  background.cdr
```

penetrance_lower_bound and penetrance_upper_bound: To restrict the analysis to a certain range of penetrance levels, penetrance boundaries can be specified.

lod_score_filter: Specifying "yes" will force pVAAST to evaluate only variants with positive LOD scores, i.e., variants with at least some evidence for genetic linkage.

inheritance_error_filter: Specifying "yes" will eliminate variant sites containing Mendelian inheritance errors. This will produce a cleaner result when genotyping error is frequent, but will also increase the number of false negatives, in particular with de novo mutations.

pVAAST Results and Output

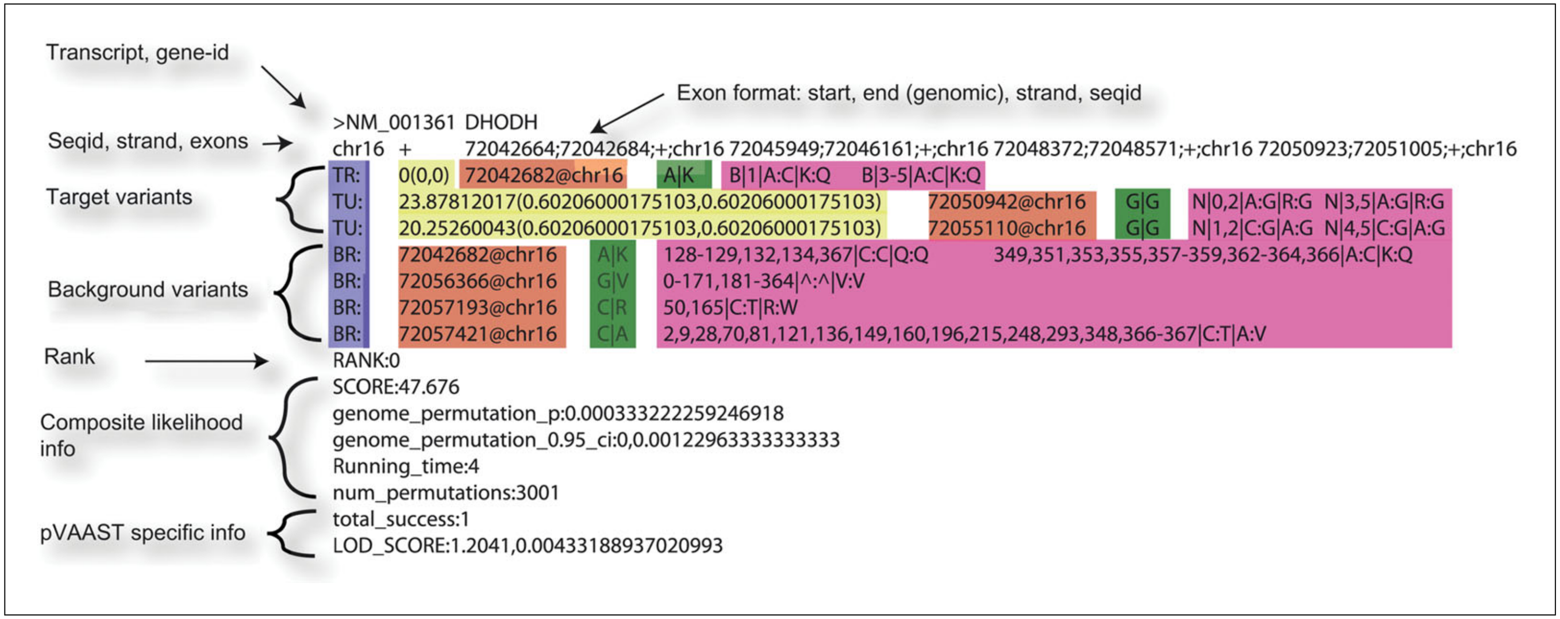
Figure 6.14.4 shows the .vaast output for pVAAST, which is very similar to a regular VAAST output but reports additional linkage information in the pedigree. The following items are added or revised in a pVAAST .vaast output.

Score and p-values

Similar to VAAST, pVAAST reports SCORE and genome_permutation_p for each gene. However, the score here is the CLRT$_p$ value for the gene, which is the sum of the CLRT score from VAAST plus 2ln(10) × LOD. Correspondingly, genome_permutation_p reports the significance of the CLRT$_p$ statistics for the current gene.

LOD scores

The LOD_SCORE item will only be seen in a pVAAST report. The traditional LOD score is the odds of the alternative hypothesis that the variant and the disease are linked versus the null model that the variant and disease are unlinked, on a log 10 scale. For example, a LOD score of 3 indicates 1000-to-1 odds in favor of linkage. The pVAAST LOD score has a similar but not identical scale to a traditional two-point parametric LOD score. In the same line, separated by comma, is the *p*-value for the reported LOD score. This is the one-tail probability of observing the reported LOD score (or higher) when there is no linkage between any variant in this gene and the disease-causing mutation. This measures the significance of the LOD score. In contrast, the overall *p*-value measures the significance of the combined VAAST score and LOD score.



```
>NM_001361  DHODH
chr16  +     72042664;72042684;+;chr16 72045949;72046161;+;chr16 72048372;72048571;+;chr16 72050923;72051005;+;chr16
TR:    0(0,0)  72042682@chr16      A|K    B|1|A:C|K:Q    B|3-5|A:C|K:Q
TU:    23.87812017(0.60206000175103,0.60206000175103)        72050942@chr16    G|G    N|0,2|A:G|R:G  N|3,5|A:G|R:G
TU:    20.25260043(0.60206000175103,0.60206000175103)        72055110@chr16    G|G    N|1,2|C:G|A:G  N|4,5|C:G|A:G
BR:    72042682@chr16    A|K    128-129,132,134,367|C:C|Q:Q        349,351,353,355,357-359,362-364,366|A:C|K:Q
BR:    72056366@chr16    G|V    0-171,181-364|^:^|V:V
BR:    72057193@chr16    C|R    50,165|C:T|R:W
BR:    72057421@chr16    C|A    2,9,28,70,81,121,136,149,160,196,215,248,293,348,366-367|C:T|A:V
RANK:0
SCORE:47.676
genome_permutation_p:0.000333222259246918
genome_permutation_0.95_ci:0,0.00122963333333333
Running_time:4
num_permutations:3001
total_success:1
LOD_SCORE:1.2041,0.00433188937020993
```

Figure 6.14.4 pVAAST report for the *DHODH* gene. The report is identical to a regular VAAST report except that: (1) In the target variant lines, pVAAST reports the variant LOD score for each family in parentheses after the variant VAAST score. The format of the LOD score report is: the 0-based pedigree IDs, a pipe sign(|), and then the LOD score. If more than one LOD score value exists, they are separated by semicolons. (2) In the target variant lines, the numerical individual IDs before the genotypes are cumulative. For example, if there are two families and the first family has three sequenced individuals (0,1, and 2), then the numerical IDs of the second family start from 3. (3) A new line `LOD_SCORE` is added, which contains the gene LOD score and its statistical significance, separated by commas.

pVAAST Simple Output

Like VAAST, pVAAST outputs a `.simple` file to provide a concise summary of the results. Below is an example of the first five lines and six columns of a pVAAST `.simple` file:

```
RANK   Gene      p-value    p-value-ci            Score    LOD
1      DHODH     0.000333   0,0.00123             47.676   1.204
2      CNTNAP4   0.000666   8.44e-06,0.00186      13.354   0
3      ZFHX3     0.001      8.07e-05,0.00241      16.118   0
4      SBK1      0.002      0.000541,0.00389      16.461   0.602
```

From left to right, the columns represent: rank, gene name, *p*-value of the gene, confidence interval of the *p*-value, $\mathrm{CLRT_p}$ score, LOD score, and variant information (the variant column here is removed for illustration purposes). In this result, *DHODH* is ranked as the first genome-wide with a LOD score of 1.204 (sum of LOD scores from the two families). Within each family, two compound heterozygous variants are found in affected individuals, consistent with a recessive inheritance pattern.

ACCESSING VAAST THROUGH OPAL

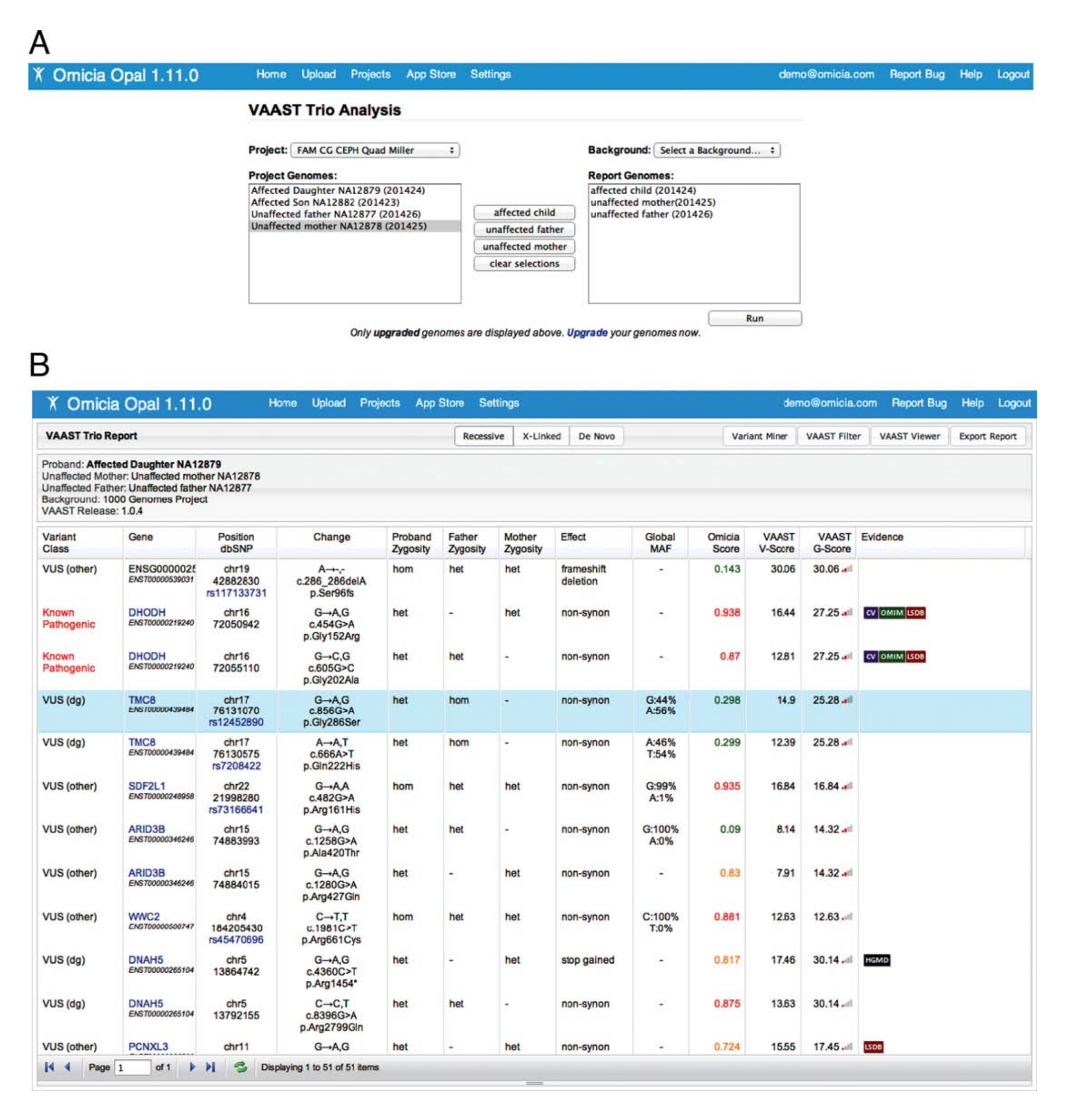
ALTERNATE PROTOCOL

In addition to the stand-alone VAAST software package, we have implemented relevant clinical workflows using VAAST in a rich genome interpretation and reporting platform called Omicia Opal (*http://app.omicia.com*). Omicia Opal is a very user-friendly Web-accessible, software-as-a-service informatics platform that allows researchers to collaboratively analyze genomes by prioritizing disease-causing variants and genes, reducing the noise and distilling data to clinically relevant findings. It has pre-loaded a comprehensive set of clinical variant databases and the most popular interpretation algorithms to provide a comprehensive but easy to use environment for the clinical researcher. The details of Opal have been described elsewhere (Coonrod et al., 2013). Part of the workflows implemented in Opal is a VAAST-based workflow for family analyses that implement VAAST small-pedigree analyses for trios and quartets.

VAAST is seamlessly integrated with Opal via an easy-to-use graphical user interface that makes this powerful algorithm accessible to a wide range of clinical users without the overhead of the infrastructure and staff necessary to manage large computationally intensive environments. Opal embeds VAAST results within a comprehensive interpretation environment wherein candidate genes and variants are presented in the context of numerous additional analyses and linked to a rich set of genome and clinical annotations from well-established pathogenic variant databases.

Figure 6.14.5A shows the graphical interface for a family analysis where the user only has to selected the genomes for analyses and specific a background file provided by Opal. The variant files are previously uploaded, validated, and annotated after the user has created an account at Opal. Opal allows for lots of interactive data manipulation and pre-processing steps for VAAST. In addition, VAAST results for a genome under analysis (example Fig. 6.14.5.B) are presented in a rich reporting format and directly integrated with live clinical database links for clinical validation and analysis. The Opal system also provides a powerful reporting infrastructure to be used in clinical laboratories and hospital. Details on the Opal system can be found at *http://www.omicia.com/what.html*.

The VAAST implementation in Opal allows easy access for any clinical research group, where workflows have been pre-validated and tested for accuracy estimates.



VAAST Trio Analysis

Project: FAM CG CEPH Quad Miller

Background: Select a Background...

Project Genomes:
Affected Daughter NA12879 (201424)
Affected Son NA12882 (201423)
Unaffected father NA12877 (201426)
Unaffected mother NA12878 (201425)

Report Genomes:
affected child (201424)
unaffected mother(201425)
unaffected father (201426)

*Only **upgraded** genomes are displayed above. **Upgrade** your genomes now.*

VAAST Trio Report

Proband: **Affected Daughter NA12879**
Unaffected Mother: Unaffected mother NA12878
Unaffected Father: Unaffected father NA12877
Background: 1000 Genomes Project
VAAST Release: 1.0.4

Variant Class	Gene	Position dbSNP	Change	Proband Zygosity	Father Zygosity	Mother Zygosity	Effect	Global MAF	Omicia Score	VAAST V-Score	VAAST G-Score	Evidence
VUS (other)	ENSG0000025 ENST00000539031	chr19 42882830 rs117133731	A→-,- c.286_286delA p.Ser96fs	hom	het	het	frameshift deletion	-	0.143	30.06	30.06	
Known Pathogenic	DHODH ENST00000219240	chr16 72050942	G→A,G c.454G>A p.Gly152Arg	het	-	het	non-synon	-	0.938	16.44	27.25	CV OMIM LSDB
Known Pathogenic	DHODH ENST00000219240	chr16 72055110	G→C,G c.605G>C p.Gly202Ala	het	het	-	non-synon	-	0.87	12.81	27.25	CV OMIM LSDB
VUS (dg)	TMC8 ENST00000439484	chr17 76131070 rs12452890	G→A,G c.856G>A p.Gly286Ser	het	hom	-	non-synon	G:44% A:56%	0.298	14.9	25.28	
VUS (dg)	TMC8 ENST00000439484	chr17 76130575 rs7208422	A→A,T c.666A>T p.Gln222His	het	hom	-	non-synon	A:46% T:54%	0.299	12.39	25.28	
VUS (other)	SDF2L1 ENST00000248958	chr22 21998280 rs73166641	G→A,A c.482G>A p.Arg161His	hom	het	het	non-synon	G:99% A:1%	0.935	16.84	16.84	
VUS (other)	ARID3B ENST00000346246	chr15 74883993	G→A,G c.1258G>A p.Ala420Thr	het	het	-	non-synon	G:100% A:0%	0.09	8.14	14.32	
VUS (other)	ARID3B ENST00000346246	chr15 74884015	G→A,G c.1280G>A p.Arg427Gln	het	-	het	non-synon	-	0.83	7.91	14.32	
VUS (other)	WWC2 ENST00000500747	chr4 184205430 rs45470696	C→T,T c.1981C>T p.Arg661Cys	hom	het	het	non-synon	C:100% T:0%	0.881	12.63	12.63	
VUS (dg)	DNAH5 ENST00000265104	chr5 13864742	G→A,G c.4360C>T p.Arg1454*	het	-	het	stop gained	-	0.817	17.46	30.14	HGMD
VUS (dg)	DNAH5 ENST00000265104	chr5 13792155	C→C,T c.8396G>A p.Arg2799Gln	het	het	-	non-synon	-	0.875	13.63	30.14	
VUS (other)	PCNXL3	chr11	G→A,G	het	-	het	non-synon	-	0.724	15.55	17.45	LSDB

Displaying 1 to 51 of 51 items

Figure 6.14.5 (**A**) Opal/VAAST: simplified Trip-Analysis workflow in Omicia Opal (*http://app.omicia.com*). (**B**) Opal/VAAST report for the *DHODH* gene. The VAAST report in Opal reports additional clinical annotations for each variant such as the American College of Medical Genetics–recommended variant pathogenicity annotation as recorded in ClinVAR, the MAF in the 1000 Genomes data, as well as clinical evidence as provided in OMIM, ClinVAR, HGMD, LSDB, and GWAS. It also displays the genotype for each individual in the study (for a trio: father, mother and affected child).

COMMENTARY

Background Information

Here we present detailed instructions on the use of VAAST in disease-gene discovery projects and Opal with VAAST for clinical reporting projects. For small studies involving publicly available controls and a few sequenced cases, VAAST serves as a highly sensitive variant classifier and gene-prioritization tool. For rare Mendelian diseases segregating in small pedigrees, VST can be used in conjunction with VAAST to identify variants that are likely to be damaging and that are consistent with the expected inheritance pattern. For large studies with matched cases and controls, VAAST performs robust, genome-wide rare variant association analysis that directly incorporates calibrated variant classification scores based on phylogenetic conservation and AAS information. pVAAST implements a highly flexible approach for pedigree analysis that supports a broad range of study designs—from monogenic, Mendelian diseases in a single family to polygenic, common diseases involving hundreds of families. Our most recent implementation of VAAST within Opal allows for more complex diagnostic clinical decision support with a user-friendly, Web-accessible interface. We hope that this guide will serve as a useful aid that will help enable new discoveries through comprehensive analysis of NGS data.

Guidelines for Understanding Results

The six columns for the `.simple` report are: rank, gene, *p*-value, confidence interval, gene score, and a variant info field. The full `.vaast` report file contains additional information about the specific variants contributing to the gene score in both the cases and controls and the *p*-value 95% confidence interval (Fig. 6.14.6).

Rank

Gene candidates are sorted by *p*-value, from smallest to largest. When two genes have the same *p*-value, the VAAST score of each gene determines the rank.

Score

The VAAST score combines variant frequency data with amino acid score and phylogenetic conservation information using a composite likelihood ratio test (CLRT). The higher the score, the more likely that the gene contains disease variants under the VAAST model. The VAAST score is the Akaike Information Criterion in the VAAST model, which is $-2\ln(\lambda) - 2m$, where λ is the composite likelihood ratio and m is the number of free parameters in the model (Yandell et al., 2011; Hu et al., 2013). The VAAST score for each variant is reported, and the feature score is the sum of all variant scores in a feature.

p-value

VAAST estimates the *p*-value using a permutation test by comparing the observed VAAST score to the scores obtained by permutation. VAAST 2.0 (Hu et al., 2013) reports the ratio of 1 + the number of successes to the total number of permutations as the *p*-value (VAAST 1.0 reports the ratio of successes to the number of permutations, constrained by the minimum p-value of 1/the total number of possible permutations). Unless the sample sizes are small, VAAST will not completely enumerate all possible orderings of the data, and thus the *p*-value for a gene is a random variable that depends on the number of permutations that may vary between VAAST runs. For a genome-wide search in human data, the significance threshold for a VAAST search is $\sim 2.4 \times 10^{-6}$, or 0.05/21,000 genes. Interpreting the *p*-value without consideration of the confidence interval can lead to erroneous conclusions if the number of permutations is insufficient. For more on this point, see the “Confidence Interval” and “Specifying the number of permutations” sections below.

Confidence interval

We estimate the 95% confidence interval of the *p*-value from the VAAST permutation test, assuming that the number of successes follows a Poisson distribution. The confidence interval of the *p*-value describes the uncertainty in the *p*-value estimate and is one of the most important metrics to consider when interpreting the results of a VAAST analysis. The more permutations run, the smaller the confidence interval around the *p*-value.

Variant field

The first variant field (the 6th column) from the example `10cases_output.simple` output in Basic Protocol 2 is:

```
chr3:87302571;35.83;G->T;D-
  >Y;0,5
chr3:87302557;31.69;A->G;N-
  >S;0,5
```

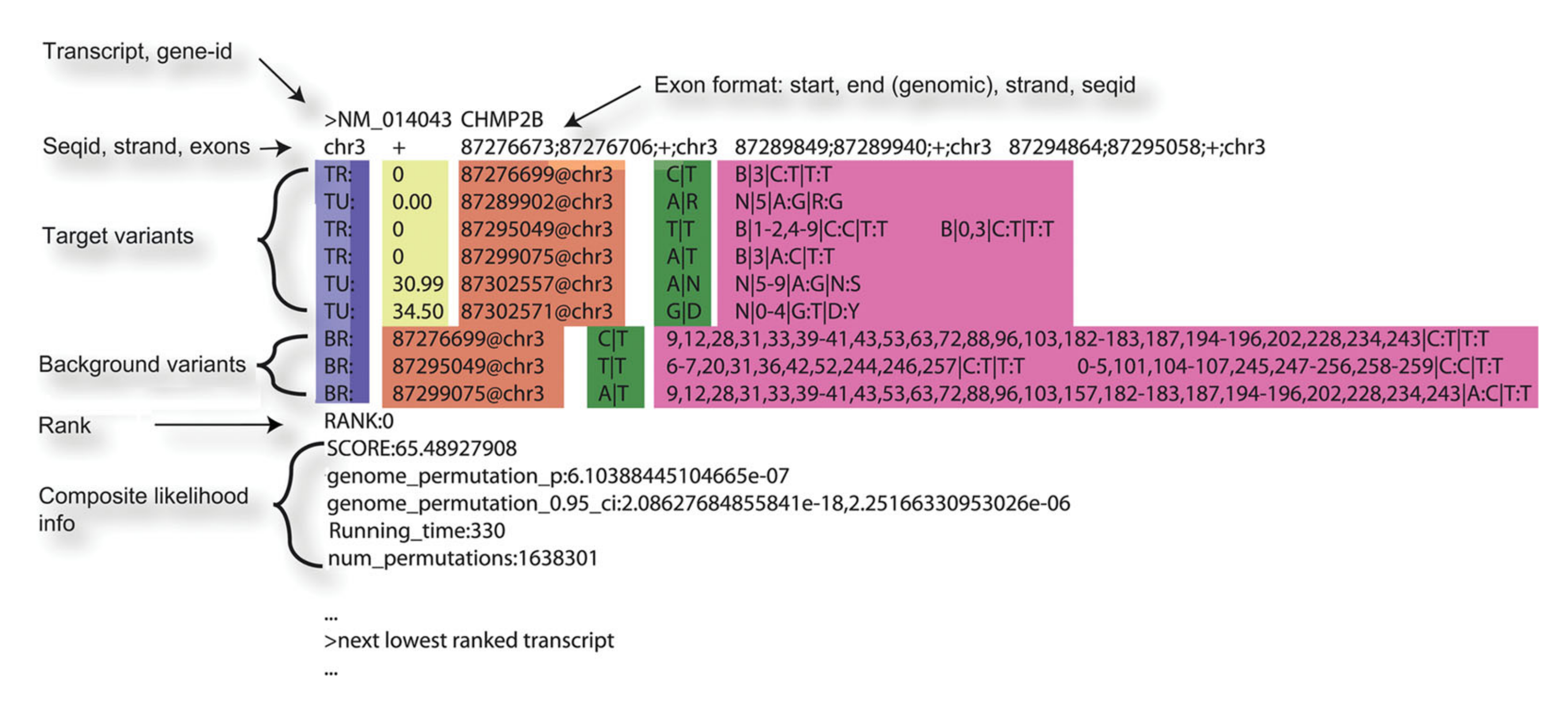
This field provides a list of the most likely disease-causing variants (as judged by VAAST) in the feature. Each variant is separated by a tab, and they are further delimited by semicolons. The first semicolon-delimited field is the variant’s location, the second is the VAAST variant score, the third is the nucleotide change, the fourth is the amino acid change, and the last field gives the count values for the variant in the background and the target. In the example above, there are two variants in *CHMPB2* (the ones that were doped in). These variants are absent in the background and present with 5 copies each in the target.

Using `vaast_pdf_reporter`

This section provides examples of using `vaast_pdf_reporter` to aid in the interpretation of results from the *CHMPB2* dataset presented in Basic Protocol 2. The `vaast_pdf_reporter` script provides a summary of results from a VAAST analysis:

```
vaast_pdf_reporter.pl
  10cases_output.vaast
```

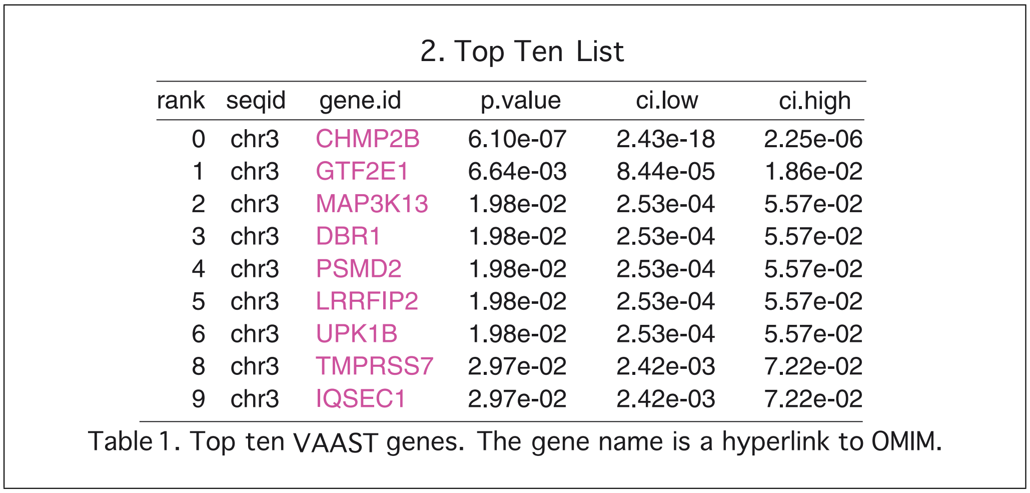
The output used to generate Figures 6.14.7, 6.14.8, and 6.14.9 was generated from the following command line:


Transcript, gene-id
Exon format: start, end (genomic), strand, seqid
Seqid, strand, exons
Target variants
Background variants
Rank
Composite likelihood info

```
>NM_014043  CHMP2B
chr3   +    87276673;87276706;+;chr3   87289849;87289940;+;chr3   87294864;87295058;+;chr3
TR:  0      87276699@chr3   C|T   B|3|C:T|T:T
TU:  0.00   87289902@chr3   A|R   N|5|A:G|R:G
TR:  0      87295049@chr3   T|T   B|1-2,4-9|C:C|T:T      B|0,3|C:T|T:T
TR:  0      87299075@chr3   A|T   B|3|A:C|T:T
TU:  30.99  87302557@chr3   A|N   N|5-9|A:G|N:S
TU:  34.50  87302571@chr3   G|D   N|0-4|G:T|D:Y
BR:  87276699@chr3   C|T   9,12,28,31,33,39-41,43,53,63,72,88,96,103,182-183,187,194-196,202,228,234,243|C:T|T:T
BR:  87295049@chr3   T|T   6-7,20,31,36,42,52,244,246,257|C:T|T:T      0-5,101,104-107,245,247-256,258-259|C:C|T:T
BR:  87299075@chr3   A|T   9,12,28,31,33,39-41,43,53,63,72,88,96,103,157,182-183,187,194-196,202,228,234,243|A:C|T:T
RANK:0
SCORE:65.48927908
genome_permutation_p:6.10388445104665e-07
genome_permutation_0.95_ci:2.08627684855841e-18,2.25166330953026e-06
Running_time:330
num_permutations:1638301

...
>next lowest ranked transcript
...
```

Figure 6.14.6 Full VAAST report for the *CHMP2B* gene. The first column (blue) describes each variant's origins: TR, present in the target and background; TU, present in target but not background; and BR, background. Target variants' VAAST scores are present in the second column (yellow). Notice that only two variants in the target are contributing to the overall gene score. The variants' positions are reported (orange) as the physical position and seqid separated by an @. The reference base and reference amino acid are separated by a | symbol in the following column (green). The final column(s) (pink) display the genotypes for all the target and background individuals, by their numerical IDs. Non-reported IDs have the reference genotype. Each genotype column contains one genotype and the associated individual's IDs. Each genotype column is split into three fields separated by a | symbol: IDs, Genotype (DNA), Genotype (amino acid). Individual IDs are associated with the file-index info field at the bottom of the target and background CDR files. In the case of a no-call (^ : ^) the amino acid annotation is present, while the genotype is not considered in the CLRT. The data presented in this figure were truncated for presentation purposes.



2. Top Ten List

rank	seqid	gene.id	p.value	ci.low	ci.high
0	chr3	CHMP2B	6.10e-07	2.43e-18	2.25e-06
1	chr3	GTF2E1	6.64e-03	8.44e-05	1.86e-02
2	chr3	MAP3K13	1.98e-02	2.53e-04	5.57e-02
3	chr3	DBR1	1.98e-02	2.53e-04	5.57e-02
4	chr3	PSMD2	1.98e-02	2.53e-04	5.57e-02
5	chr3	LRRFIP2	1.98e-02	2.53e-04	5.57e-02
6	chr3	UPK1B	1.98e-02	2.53e-04	5.57e-02
8	chr3	TMPRSS7	2.97e-02	2.42e-03	7.22e-02
9	chr3	IQSEC1	2.97e-02	2.42e-03	7.22e-02

Table 1. Top ten VAAST genes. The gene name is a hyperlink to OMIM.

Figure 6.14.7 Example table 1 from `VAAST-pdf-reporter`.

```
VAAST/bin/VAAST -m lrt -iht d
  -j 0.0000026 -o 10case_ouput
  -d 1e7 -p 20
  refGene_hg19_chr3.gff3
  1KG_chr3_Background.cdr
  10cases_union.cdr
```

This command line includes options discussed in the “Advanced Options” section, below, and produces the following output file:

```
10cases_output.vaast.report
  .vaast.pdf
```

The PDF file contains several useful results, including the top ten ranked genes (Fig. 6.14.7), a quantile-quantile (QQ) plot of expected versus empirical *p*-values (Fig. 6.14.8), and a Manhattan plot displaying the -log$_{10}$ *p*-values across the genome (Fig. 6.14.9). In this dataset, *CHMPB2* is the only gene that reaches genome-wide significance (Figs. 6.14.7-6.14.9).

Specifying the optimal number of permutations

The optimal number of permutations is a tradeoff between statistical accuracy and compute time. The total number of possible permutations for a given set of target and background genomes is described by the binomial coefficient (often referred to as ‘*n* choose *k*’), and this value grows rapidly as the number of target and background genomes increases. For a VAAST analysis with a single target genome and 1000 background genomes, there are 1001 ways to choose a single target from the combined pool of all genomes, and VAAST can consider all possible permutations to calculate the exact *p*-value. With 10 genomes in the target and 256 in the background—as in the example above—there are 4.1×10^{17} combinations, and it is neither possible nor necessary to perform all possible permutations. When the number of permutations specified on the command line is less than the number of possible permutations, VAAST randomly permutes the status of the background and target genomes.

If fewer than 100,000 permutations are specified, *p*-values (and consequently gene ranks) will often show significant random variation between VAAST runs with the same datasets and command-line parameters. If the confidence interval is large, *p*-values and gene ranks will fluctuate from run to run. For most applications, 10,000 permutations will be fast, but will likely result in poor statistical resolution. Increasing the number of permutations will provide tighter confidence intervals and more stable gene rankings.

VAAST supports two forms of parallelization: by feature (`--mp1`) and by permutation (`--mp2`). Each of these options indicates how many CPU cores should be allocated to VAAST. The `--mp1` option instructs VAAST to distribute a separate feature to each core. The `--mp2` option instructs VAAST to distribute a batch of permutations, for the same feature, to each core, allowing a single feature to be processed in parallel by multiple cores. A good strategy for a genome-wide analysis is to perform an initial analysis scoring all features in the genome using the `--mp1` option, but with a relatively small number of permutations (`-d`). The *p*-values from this initial run will not be accurate, and the confidence intervals will be wide. A second VAAST analysis should be

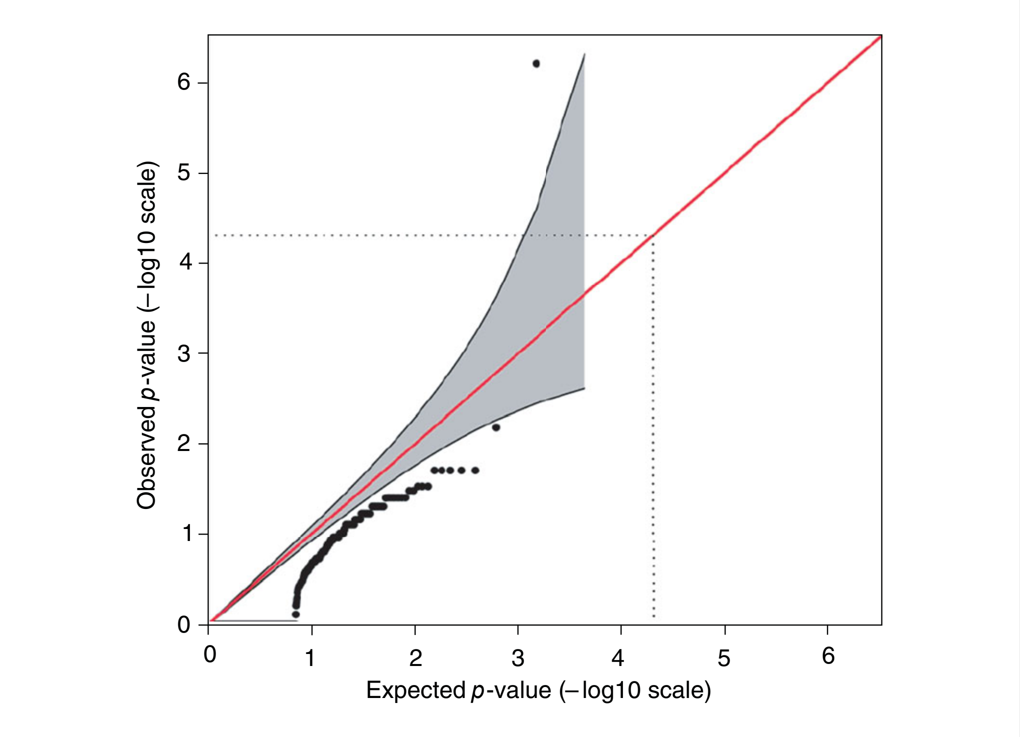


Figure 6.14.8 Example QQ plot from `VAAST-pdf-reporter`. The expected *p*-values are plotted on the *x* axis and the VAAST *p*-values are plotted on the *y* axis, on a –log base 10 scale. The gray area shades the 95% confidence intervals. The red line plots $y = x$, corresponding to the scenario where the observed and expected *p*-values are perfectly matched. The QQ plot provides a measure of potential inflation in Type I error that may result from systematic differences between cases and controls. In this plot, the expected *p*-value is usually lower than the observed *p*-value, indicating that the test is slightly conservative. This is a typical result when cases and controls are well matched and sample sizes are small. As sample sizes and permutation counts increase, the QQ plot becomes less conservative and more continuous. In this example, there are 10 cases and 256 controls, and only one gene exceeds the upper 95% confidence interval, *CHMP2B*. When cases and controls are poorly matched, for instance due to population stratification or mismatched bioinformatic pipelines, the observed –log *p*-values can be much greater than the expected values, which indicates that Type I errors are inflated and the *p*-values are nonconservative. With publicly available controls, some inflation in Type I error rate is often unavoidable, but several techniques are available to minimize the problem, which can greatly improve the quality of the gene prioritization results (see Variant-calling section above). The solid line black line on the left side of the *x* axis represents genes with a VAAST score of 0; these genes have the maximum *p*-value of 1 but should be included when determining the number of multiple comparisons, because the VAAST score includes case-control status information.

performed with the top-ranking features (e.g., all those features whose confidence interval indicate that they could reach the desired significance level). The second analysis is performed using `--mp2`, a higher permutation number, and the `--features` option. The `--gw` option in VAAST automates this two-step procedure. For example, the following command performs a genome-wide VAAST search with 2,000 permutations using parallelization by features. From the results of the first run, VAAST then selects genes with low *p*-values and performs a second round of analysis with 1 million permutations using parallelization by permutations. Throughout the whole process, 50 CPUs are used, as specified by the `-p` parameter:

```
VAAST/bin/VAAST -m lrt -iht d
  -o 10case_ouput --less_ram
  --gw 1e7 -p 20
  refGene_hg19_chr3.gff3
  1KG_chr3_Background.cdr
  10cases_union.cdr
```

Advanced Options

VAAST has many options that can be used to improve the speed and accuracy of VAAST searches, which are fully documented in the VAAST User's Guide. The most frequently used options are discussed below.

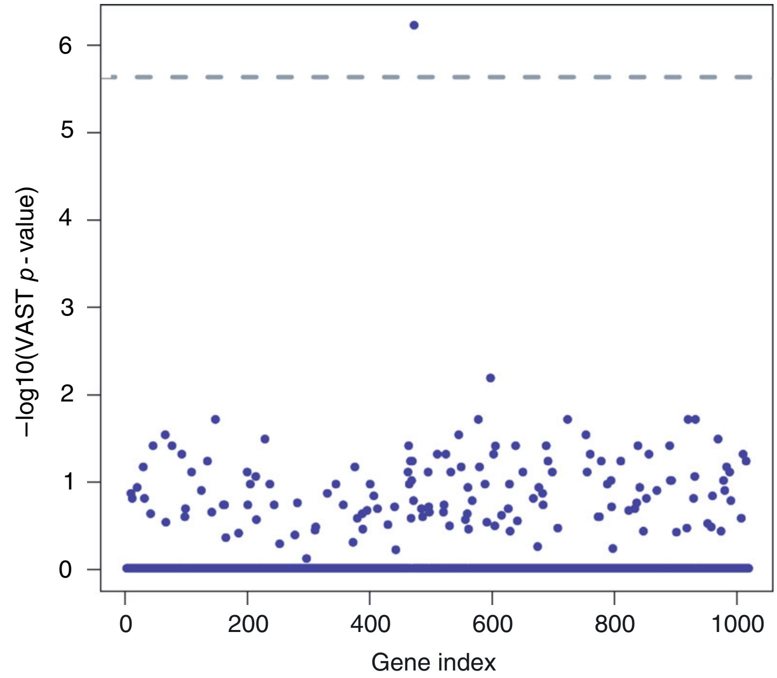


Figure 6.14.9 Manhattan plot for the *CHMP2B*. The *x* axis is an index of the genes on chromosome 3, and the *y* axis is the negative log10 VAAST *p*-value. The gray dashed line represents the genome-wide significance level for human data (see text). *CHMP2B* is the only gene that achieves genome-wide significance.

General options

`-c --chrom`: Specifies specific chromosome(s) to be analyzed.

`--mask_regions`: Specifies a user provided BED file identifying genomic positions or regions that will be masked from the analysis (i.e., not scored). This option can be useful when excluding regions enriched for known false positives or to exclude specific sites with increased missing genotype or Hardy-Weinberg disequilibrium rates. Note that the `--variant_mask` option in VAAST 1.0 only masks individual variants (not regions), and may be faster when only individual variants are included in the masking file.

`--regions`: This option is similar to `--mask_regions`, but instead specifies a BED file containing only those regions which will be scored.

`--restart`: This option is used to restart a VAAST job that was interrupted (e.g., system rebooted or job was killed). VAAST will read the temporary files in the output directory to resume from the point where the job was interrupted.

Disease and population options

`-iht --inheritance`: This option applies an inheritance model variant filter for damaging alleles. Arguments are `r` (recessive) or `d` (dominant). The dominant option allows only one allele in each feature to receive a score in each individual. The recessive option allows up to two alleles to receive a score in each individual (the best-scoring homozygous variant or the two best scoring heterozygous variants). By default, all variants with a positive score are evaluated for all individuals. For large genes, the recessive option can help mitigate batch effects in which rare variants are more frequent in cases relative to controls.

`-pnt --penetrance`: Specifies whether damaging alleles are expected to be fully penetrant. By default, the flag is set to `i` to indicate incomplete penetrance. The behavior of `-pnt c` (complete penetrance) is different under the two inheritance (dominant/recessive) models. Under a recessive model with complete penetrance, any target variant that is homozygous in any of the background individuals will not be scored. In addition, under the recessive model, any combination of two heterozygous alleles in the same feature that are shared between any target and any background genome will not be scored. Under a dominant model, any variant that is shared between any target and any background genome will be ignored. The complete penetrance option imposes a very stringent filter that does not take into consideration missing genotypes and thus should be used with great caution.

`-lh --locus_heterogeneity`: By default this option is set to "yes." Setting locus heterogenetiy to "n(o)" applies a filter that requires every individual in the target to have a genotype that is scored by VAAST in order for a feature to receive a score. Under the dominant inheritance model, this requires every individual in the target to have at least one allele with a VAAST score greater than 0 in the feature. Under the recessive model, every individual must have at least two alleles with a VAAST score greater than 0. Like penetrance, setting locus heterogeneity to "n" applies a stringent filter that may result in false negatives due to missing data. This option should be reserved for monogenic Mendelian diseases to test the hypothesis that a single gene can fully explain the phenotype in the target population.

`--protective`: By default, VAAST only scores variants where the minor allele frequency (MAF) is higher in the target than the background. The protective option allows VAAST to also consider potential protective alleles where the MAF is lower in the target than the background.

`--rate`: This option sets the maximum MAF of variants in the background genomes, penalizing the score of variants that exceed this frequency in the background genomes. This flag can be used to restrict the analysis to rare variants.

Feature options

`--parent_feature`: Determines which features in the GFF3 file will be scored by VAAST. The default is mRNA. One record will be present in the VAAST output for each `parent_feature` in the GFF3 file.

`--child_feature` : Because scoring transcripts is a primary goal of a VAAST analysis and because transcripts can be discontinous features with exons separated by introns, VAAST provides a `child_feature` option that will allow only the portions of a `parent_feature` to score that have a corresponding `child_feature` region. This option describes the child feature in the GFF3 file that will contribute to the score in the parent feature. Features in the GFF3 file use the Parent attribute to describe the parent/child relationship between features (i.e., describe which CDSs belong to which mRNA). All features given to the `child_feature` option must be annotated in the GFF3 file because VAAST will not infer features. When the `--all_variants` option is specified, the default value is "exon"; otherwise, the default value is CDS.

`--features`: When passed a comma-separated list of `parent_feature` IDs (for example: NM_0123456,NM_0654321) VAAST will only score the given features. In addition, the `--features` option can take a file containing a separate feature ID on each line. This flag can be used to obtain the VAAST score on a single gene or transcript or to refine the *p*-value on a limited set of features that appear to be significant in a genome-wide VAAST run.

Scoring options

`-k --codon_bias`: Enabled by default, this instructs VAAST to include amino acid substitution frequencies information in the composite likelihood ratio calculation. We highly recommend enabling this option for all VAAST runs, except when the `--protective` option is used. In VAAST 2.0, the amino acid substitution matrix also incorporates the phylogenetic conservation information, as measured by PhastCons scores (generated from multiple-species protein alignment in vertebrates; Yang, 1995; Meyer et al., 2013).

`-e --all_variants`: This option instructs VAAST to score all variants in a feature, including synonymous and noncoding variants.

`-d --gp`: This parameter sets the number of permutations for each feature. This option can be combined with `--features` or `--regions` to provide an increased permutation count for particular genes.

`--gw`: If this parameter is set, VAAST will first perform a genome-wide search with lower number of permutations, identify highly significant features, and then perform intensive permutations over these features to get more accurate *p*-values. This parameter takes an integer that specifies the maximum number of permutations for each feature. For genome-wide VAAST searches, using `--gw` generates the same results as `-d` option, but is much more efficient.

`-g --grouping`: This option instructs VAAST to group together variants in the target genome when the number of occurrences of the minor allele is below the given value. The default value is 4. A value of 0 will disable grouping.

--mean_prior: This option changes the behavior of VAAST for variant grouping. By default, the weight of grouped variants is the product of individual variants. If this option is set, VAAST will instead use the geometric mean of the weights from all grouped variants as the final weight. We recommend setting this option when the number of target genomes is greater than 10. This option may become a default value for larger sample sizes in future releases.

-j --significance: Establishes a desired threshold for statistical significance. With -j, VAAST will stop the permutation test when the *p*-value is less than or greater than the specified value with 95% confidence. Often -j is set for 2.4×10^{-6} (genome-wide significance after multiple-test correction on a genome with 21,000 genes).

--min_suc: Forces VAAST to keep running permutation tests on a gene until at least the given number of tests in the permutation have a CLRT score higher than the specified value. The number of permutations is still constrained by the -d option. If the maximal number of permutations (-d) is reached, VAAST will stop regardless of the --min_suc parameter.

--indel: Instructs VAAST to score genomic insertions and deletions (indels) that overlap features. Empirically, indels are more sensitive to differences in sequencing platforms and variant-calling pipeline than SNVs, and thus are more likely to introduce false positive signals, particularly when the controls and cases are not matched.

--splice_site: Allows VAAST to score variants that affect splice donor and splice acceptors, if these variants are annotated in the CDR file. This functionality will eventually be enabled by default but is currently in beta testing.

The example below uses several of the advanced options described above. Disease and population options including dominant inheritance pattern (-iht d), indel scoring (--indel), disabling grouping (-g 0), setting significance threshold (-j 0.0000026), and incomplete penetrance (-pnt i). This example also specifies several performance parameters: parallel processing with 10 CPUs (-p 20) and perform 10,000,000 permutations per feature (-d 1e7):

```
VAAST/bin/VAAST -m lrt -iht d
  -g 0 -j 0.0000026 -o
  10case_ouput -d 1e7 -p 20
  --indel
  refGene_hg19_chr3.gff3
  1KG_chr3_Background.cdr
  10cases_union.cdr.
```

Mitigating the Effects of Mismatched Cases and Controls

The results of VAAST analyses are often not as clear as the examples presented above. Many studies sequence a relatively small number of affected individuals and compare the resulting variants directly to publicly available control data, such as the 1000 Genomes (Abecasis et al., 2012). This is a highly efficient approach but can result in substantial inflation in Type I error rates due to differences in sequencing platform and variant-calling pipelines, and in some cases, poorly matched target and background populations. The extent of false-positive rate inflation can be evaluated by visual inspection of the QQ-plot (Fig. 6.14.8). When false-positive rates are high, the goal of a VAAST analysis shifts from identifying statistically significant disease-gene associations to disease-gene prioritization and discovery. This effort often involves careful evaluation of the top genes in a VAAST analysis to identify candidates for follow-up functional validation or sequencing experiments. Problematic variants can be identified by comparing allele frequencies in other publicly available datasets, such as the NHLBI exome variant server [NHLBI GO Exome Sequencing Project (ESP), Seattle]. Discordant allele frequencies between large databases are an indication that the variant may be poorly represented in the VAAST background dataset, or that the variant is an artifact of certain variant-calling pipelines. A visual inspection of the aligned reads supporting a variant call using Samtool's tview (Li et al., 2009) or Integrative Genome Viewer (Robinson et al., 2011) can identify potential artifacts resulting from problematic alignments or poor read support. For variants that pass bioinformatic validation, confirmatory genotyping is often a cost-effective strategy.

A complementary approach for mitigating the effects of mismatched cases and controls involves the creation of an a priori list of genes or variants that are likely false positives (Kohane et al., 2012). This list can be generated by evaluating a second target set with an unrelated phenotype that matches the sequencing protocol and variant-calling pipeline of the original target set. The second target set can be compared to the background using VAAST to identify the top-ranking genes and variants, which will often be the result of systematic differences in sequencing protocols and variant-calling pipelines. In our experience, large genes with repeated domains or many homologs, such as titan, mucins (Kamphans et al., 2013), and zinc fingers, are particularly prone to sequencing and variant-calling batch effects.

Acknowledgments

This work was supported by NIH grants R01 GM104390, R01 GM59290, SBIR grant R44HG3667, SBIR grant R44HG006579, and ARRA GO grant RC2HG005619. H.H. was supported by the MD Anderson Cancer Center Odyssey Program, and B.K. was supported by NIH grant 1ULTR001067.

Literature Cited

Abecasis, G.R., Auton, A., Brooks, L.D., DePristo, M.A., Durbin, R.M., Handsaker, R.E., Kang, H.M., Marth, G.T., and McVean, G.A. 2012. An integrated map of genetic variation from 1,092 human genomes. *Nature* 491:56-65.

Challis, D., Yu, J., Evani, U.S., Jackson, A.R., Paithankar, S., Coarfa, C., Milosavljevic, A., Gibbs, R.A., and Yu, F. 2012. An integrative variant analysis suite for whole exome next-generation sequencing data. *BMC Bioinformatics* 13:8.

Coonrod, E.M., Margraf, R.L., Russell, A., Voelkerding, K.V, and Reese, M.G. 2013. Clinical analysis of genome next-generation sequencing data using the Omicia platform. *Exp. Rev. Mol. Diagn.* 13:529-540.

DePristo, M.A., Banks, E., Poplin, R., Garimella, K.V, Maguire, J.R., Hartl, C., Philippakis, A.A., del Angel, G., Rivas, M.A., Hanna, M., McKenna, A., Fennell, T.J., Kernytsky, A.M., Sivachenko, A.Y., Cibulskis, K., Gabriel, S.B., Altshuler, D., and Daly, M.J. 2011. A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet.* 43:491-498.

Drmanac, R., Sparks, A.B., Callow, M.J., Halpern, A.L., Burns, N.L., Kermani, B.G., Carnevali, P., Nazarenko, I., Nilsen, G.B., Yeung, G., et al. 2010. Human genome sequencing using unchained base reads on self-assembling DNA nanoarrays. *Science* 327:78-81.

Eilbeck, K., Lewis, S.E., Mungall, C.J., Yandell, M., Stein, L., Durbin, R., and Ashburner, M. 2005. The Sequence Ontology: A tool for the unification of genome annotations. *Genome Biol.* 6:R44.

Hu, H., Huff, C.D., Moore, B., Flygare, S., Reese, M.G., and Yandell, M. 2013. VAAST 2.0: Improved variant classification and disease-gene identification using a conservation-controlled amino acid substitution matrix. *Genet. Epidemiol.* 37:622-634.

Hu, H., Roach, J.C., Coon, H., Guthery, S.L., Voelkerding, K.V., Margraf, R.L., Durtschi, J.D., Tavtigian, S.V., Shankaracharya, Wu, W., Scheet, P., Wang, S., Xing, J., Glusman, G., Hubley, R., Li, H., Garg, V., Moore, B., Hood, L., Galas, D.J., Srivastava, D., Reese, M.G., Jorde, L.B., Yandell, M., and Huff, C.D. 2014. A unified test of linkage analysis and rare-variant association. *Nature Biotech.* In press.

Kamphans, T., Sabri, P., Zhu, N., Heinrich, V., Mundlos, S., Robinson, P.N., Parkhomchuk, D., and Krawitz, P.M. 2013. Filtering for compound heterozygous sequence variants in non-consanguineous pedigrees. *PloS One* 8:e70151.

Karolchik, D., Hinrichs, A.S., Furey, T.S., Roskin, K.M., Sugnet, C.W., Haussler, D., and Kent, W.J. 2004. The UCSC table browser data retrieval tool. *Nucleic Acids Res.* 32:D493-D496.

Kohane, I. S., Hsing, M., and Kong, S.W. 2012. Taxonomizing, sizing, and overcoming the incidentalome. *Genet. Med.* 14:399-404.

Lander, E.S., Linton, L.M., Birren, B., Nusbaum, C., Zody, M.C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., Funke, R., Gage, D., et al. 2001. Initial sequencing and analysis of the human genome. *Nature* 409:860-921.

Li, H. and Homer, N. 2010. A survey of sequence alignment algorithms for next-generation sequencing. *Brief. Bioinform.* 11:473-483.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R.; 1000 Genome Project Data Processing Subgroup. 2009. The sequence alignment/map format and SAMtools. *Bioinformatics* 25:2078-2079.

McElroy, J.J., Gutman, C.E., Shaffer, C.M., Busch, T.D., Puttonen, H., Teramo, K., Murray, J.C., Hallman, M., and Muglia, L.J. 2013. Maternal coding variants in complement receptor 1 and spontaneous idiopathic preterm birth. *Hum. Genet.* 132:935-942.

McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., Gabriel, S., Daly, M., and DePristo, M.A. 2010. The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.* 20:1297-1303.

Meyer, L.R., Zweig, A.S., Hinrichs, A.S., Karolchik, D., Kuhn, R.M., Wong, M., Sloan, C.A., Rosenbloom, K.R., Roe, G., Rhead, B., Raney, B.J., Pohl, A., Malladi, V.S., Li, C.H,. Lee, B.T., Learned, K., Kirkup, V., Hsu, F., Heitner, S., Harte, R.A., Haeussler, M., Guruvadoo, L., Goldman, M., Giardine, B.M., Fujita, P.A., Dreszer, T.R., Diekhans, M., Cline, M.S., Clawson, H., Barber, G.P., Haussler, D., and Kent, W.J. 2013. The UCSC Genome Browser database: Extensions and updates 2013. *Nucleic Acids Res.* 41:D64-D69.

Nielsen, R., Paul, J.S., Albrechtsen, A., and Song, Y.S. 2011. Genotype and SNP calling from next-generation sequencing data. *Nat. Rev. Genet.* 12:443-451.

O'Rawe, J., Jiang, T., Sun, G., Wu, Y., Wang, W., Hu, J., Bodily, P., Tian, L., Hakonarson, H., Johnson, W.E., Wei, Z., Wang, K., and Lyon, G.J. 2013. Low concordance of multiple variant-calling pipelines: Practical implications for exome and genome sequencing. *Genome Med.* 5:28.

Online Mendelian Inheritance in Man, OMIM. 2013. McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University, Baltimore, Md. *http://omim.org/*.

Pabinger, S., Dander, A., Fischer, M., Snajder, R., Sperk, M., Efremova, M., Krabichler, B., Speicher, M.R., Zschocke, J., and Trajanoski, Z. 2013. A survey of tools for variant analysis of next-generation genome sequencing data. *Brief. Bioinform.* 14:1-23.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M.A.R., Bender, D., Maller, J., Sklar, P., De Bakker, P.I.W., Daly, M.J., and Sham, P.C. 2007. PLINK: A tool set for whole-genome association and population-based linkage analyses. *Am. J. Hum. Genet.* 81:559-575.

Reese, M.G., Moore, B., Batchelor, C., Salas, F., Cunningham, F., Marth, G.T., Stein, L., Flicek, P., Yandell, M., and Eilbeck, K. 2010. A standard variation file format for human genome sequences. *Genome Biol.* 11:R88.

Roach, J.C., Glusman, G., Smit, A.F.A., Huff, C.D., Hubley, R., Shannon, P.T., Rowen, L., Pant, K.P., Goodman, N., Bamshad, M., Shendure, J., Drmanac, R., Jorde, L.B., Hood, L., and Galas, D.J. 2010. Analysis of genetic inheritance in a family quartet by whole-genome sequencing. *Science* 328:636-639.

Robinson, J.T., Thorvaldsdóttir, H., Winckler, W., Guttman, M., Lander, E.S., Getz, G., and Mesirov, J.P. 2011. Integrative genomics viewer. *Nat. Biotechnol.* 29:24-26.

Rope, A.F., Wang, K., Evjenth, R., Xing, J., Johnston, J.J., Swensen, J.J., Johnson, W.E., Moore, B., Huff, C.D., Bird, L.M., Care, J.C., Opitz, J.M., Stevens, C.A., Schank, C. Fain, H.D., Robison, R., Dalley, B., Chin, S., South, S.T., Pysher, T.J., Jorde, L.B., Hakonarson, H. Lillehaug, J.R., Biesecker, L.G., Yandell, M., Arnesen, T., and Lyon, G.J. 2011. Massively parallel sequencing identifies a previously unrecognized X-linked disorder resulting in lethality in male infants owing to amino-terminal acetyltransferase deficiency. *Genome Biol.* 12:P13.

Shapiro, M.D., Kronenberg, Z., Li, C., Domyan, E.T., Pan, H., Campbell, M., Tan, H., Huff, C.D., Hu, H., Vickrey, A.I., Nielsen, S.C., Stringham, S.A., Hu, H., Willerslev, E., Gilbert, M.T,. Yandell, M., Zhang, G., and Wang, J. 2013. Genomic diversity and evolution of the head crest in the rock pigeon. *Science* 339:1063-1067.

Shirley, M.D., Tang, H., Gallione, C.J., Baugher, J.D., Frelin, L.P., Cohen, B., North, P.E., Marchuk, D.A., Comi, A.M., and Pevsner, J. 2013. Sturge-Weber syndrome and port-wine stains caused by somatic mutation in GNAQ. *New Engl. J. Med.* 368:1971-1979.

Wei, Z., Wang, W., Hu, P., Lyon, G.J., and Hakonarson, H. 2011. SNVer: A statistical tool for variant calling in analysis of pooled or individual next-generation sequencing data. *Nucleic Acids Res.* 39:e132-e132.

Yandell, M., Huff, C., Hu, H., Singleton, M., Moore, B., Xing, J., Jorde, L.B., and Reese, M.G. 2011. A probabilistic disease-gene finder for personal genomes. *Genome Res.* 21:1529-1542.

Yang, Z. 1995. A space-time process model for the evolution of DNA sequences. *Genetics* 139:993-1005.