Published in final edited form as:

Curr Protoc Hum Genet. ; 87: 7.25.1–7.25.15. doi:10.1002/0471142905.hg0725s87.

Interpreting *de novo* variation in human disease using denovolyzeR

James S. Ware[1,2,3,4], **Kaitlin E. Samocha**[1,2,3], **Jason Homsy**[1,5], and **Mark J. Daly**[1,2,3]

[1]Department of Genetics, Harvard Medical School, Boston MA

[2]Broad Institute of MIT and Harvard, Cambridge MA

[3]Analytical and Translational Genetics Unit, Massachusetts General Hospital and Harvard Medical School, Boston MA

[4]NIHR Cardiovascular Biomedical Research Unit at Royal Brompton Hospital and Imperial College London, London UK

[5]Cardiovascular Research Center, Massachusetts General Hospital, Boston MA

Abstract

Spontaneously arising (*de novo*) genetic variants are important in human disease, yet every individual carries many such variants, with a median of 1 *de novo* variant affecting the protein-coding portion of the genome. A recently described mutational model (Samocha et al., 2014) provides a powerful framework for the robust statistical evaluation of such coding variants, enabling the interpretation of *de novo* variation in human disease. Here we describe a new open-source software package, **denovolyzeR**, that implements this model and provides tools for the analysis of *de novo* coding sequence variants.

Keywords

de novo variant; exome sequencing

Correspondence to: James S. Ware.

INTERNET RESOURCES

http://www.r-project.org/

The home page of the R project. The R software environment can be downloaded from this site.

http://cran.r-project.org/web/packages/denovolyzeR/

The Comprehensive R Archive Network (CRAN) is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. The latest stable version of denovolyzeR should always be available on CRAN, and archived there in perpetuity.

http://denovolyzer.org/

The home page for the denovolyzeR project, with further supporting material

http://github.com/jamesware/denovolyzeR

GitHub is a web-based Git repository hosting service. The latest development version of denovolyzeR is hosted here prior to release on CRAN.

INTRODUCTION

Spontaneously arising (*de novo*) genetic variants are important in human disease. Every individual carries approximately 100 such variants that are not present in their parents' DNA, but rather have arisen via mutational events in the parental germ cell (egg or sperm) or early embryo, with a median of 1 *de novo* variant affecting the protein-coding portion of genome, referred to as the exome (Conrad et al., 2011; Lynch, 2010).

Exome sequencing and analysis of *de novo* variants has successfully identified genes underlying rare and genetically homogeneous Mendelian diseases. In Kabuki syndrome, for example, non-synonymous *de novo* variants were identified in KMT2D (MLL2) in 9 out of 10 unrelated individuals (Ng et al., 2010). An accumulation of this magnitude would be extremely improbable in the absence of a causal role in the disease given both the rarity and independence of *de novo* variants.

By contrast, it is more challenging to dissect the role of *de novo* variants in conditions with high levels of locus heterogeneity, including heritable complex traits and some Mendelian conditions, where *de novo* variants may be spread across many genes, and may make a smaller overall contribution to pathogenesis. Here it may be possible to assess the global contribution of *de novo* coding variants to disease by comparing their frequency in cases and controls, given sufficiently large sample sizes. However, at the level of individual genes, the interpretation of *de novo* variants is complicated by the background mutation rate, which varies greatly between genes. Additionally, as more individuals are sequenced, it is inevitable that multiple *de novo* variants will be observed in some genes by chance.

A statistical framework has recently been developed to address these challenges, with respect to *de novo* single nucleotide variants (SNVs) in coding sequence (Samocha et al., 2014). Briefly, the mutability of each gene is individually determined based on local sequence context, and the probability that a *de novo* event will arise in a single copy of the gene in one generation is calculated. The consequence of each possible *de novo* SNV is computed, and *de novo* probabilities are tabulated for each variant class (e.g. synonymous, missense, etc). In order to more fully assess loss-of-function (lof) variation, the probability of a frameshifting insertion or deletion is also estimated for all genes (proportional to the length of the gene and the ratio of nonsense to frameshifting indels genome-wide under the assumption that the two classes have similar selective pressure against them). For a given study population, *de novo* variants can be evaluated by comparing the observed numbers of variants with the number expected based on this model and the population size, using a Poisson framework. This permits robust significance estimates for the pileup of *de novo* variation in individual genes and gene sets, and increases the power of genome-wide analyses.

In this unit, we describe the application of this statistical framework to analyze *de novo* variants using **denovolyzeR**, an open-source software package written for the R statistical software environment (R Core Team, 2015). We present protocols for four analyses: to assess (i) whether there is a genome-wide excess of *de novo* variation for different functional classes of variant, (ii) whether there is a genome-wide excess of genes with multiple *de novo* variants, (iii) whether individual genes carry an excess of *de novo* variants, and (iv) whether a pre-specified set of genes collectively shows an enrichment of *de novo* variants.

BASIC PROTOCOL 1: ASSESSING THE GENOME-WIDE BURDEN OF *DE NOVO* VARIANTS

This protocol will assess whether there is a genome-wide excess of *de novo* variation for different functional classes of variant.

Materials

- A computer running the R software environment, available for UNIX platforms, Windows and MacOS from http://www.r-project.org.
- The **denovolyzeR** package. The latest stable release can be installed directly from the Comprehensive R Archive Network (CRAN) from within R

  ```
  install.packages(“denovolyzeR”)
  ```

- Other download and installation options, including for the latest development version, are described at http://denovolyzer.org
- dplyr and reshape packages. These dependencies may be installed automatically when **denovolyzeR** is installed (depending on your installation route). Otherwise they can be installed by running:

  ```
  install.packages(“dplyr”,”reshape”)
  ```

- A table of *de novo* variants. The minimum input comprises two columns of data: gene names, and variant classes (functional consequence of each variant). Example data is included in the **denovolyzeR** package, and will be used in this protocol. The dataset comprises a data.frame of *de novo* variants identified in 1078 individuals with autism (Samocha et al., 2014), named autismDeNovos. It is assumed that readers are able to import their own data into the R environment, using the read.table function or equivalent (in R, ?read.table will provide help).

1) In R, load the **denovolyzeR** package.

```
library(denovolyzeR)
```

2) Prepare input data. View demonstration data provided with the **denovolyzeR** package. Alternatively, users may import their own data in an equivalent format.

```
dim(autismDeNovos); head(autismDeNovos)
##  [1]   1040    2
```

```
##        gene  class
##  1  BCORL1  mis
##  2  SPANXD  mis
##  3    GLRA2  mis
##  4  RPS6KA3  non
##  5      TSR2  mis
##  6    GNL3L  syn
```

Variant classes must be labeled using the following terms:

“syn” (synonymous), “mis” (missense), “non” (nonsense), “splice” (canonical splice site) or “frameshift”. Alternatively, “lof” may be used collectively for loss-of-function classes (“non + splice + frameshift”). Whichever input format is chosen, summary statistics can be produced for “lof”, “prot” (protein-altering = mis + lof), and “all”. “prot” and “all” are not valid input classes. In-frame insertions/deletions are currently not evaluated within the statistical framework.

A variety of gene identifiers may be used. Valid identifiers recognized by the software include: hgncID, hgncSymbol, enstID, ensgID, geneName. The default option (“geneName”) specifies gene symbols by default, more specifically, these correspond to the “external_gene_name” provided by the Ensembl genome browser (Cunningham et al.). ensgID and enstID refer to Ensembl gene and transcript identifiers. hgncID and hgncSymbol refer to HUGO Gene Nomenclature Committee ID numbers and symbols. Within the R environment, the BiomaRt package (Durinck et al., 2009) from Bioconductor provides tools to convert gene identifiers.

3) Compare the observed burden of *de novo* variation to expectation.

The denovolyzeByClass function will perform the required analysis. The function has three required arguments:

- genes: a vector of gene identifiers, for genes that contain *de novo* variants
- classes: a vector of variant consequences (corresponding to the gene list)
- nsamples: the total number of samples analyzed (including samples without *de novo* variants). For the example data, 1078 individuals were sequenced.

```
denovolyzeByClass(genes=autismDeNovos$gene,
                  classes=autismDeNovos$class,
                  nsamples=1078)
##   class  observed expected enrichment   pValue
##  1  syn       254    302.3      0.840  0.998000
##  2  mis       655    679.0      0.965  0.826000
##  3  lof       131     94.3      1.390  0.000199
##  4 prot       786    773.2      1.020  0.328000
##  5   all     1040   1075.5       0.967  0.864000
```

For each variant class, this function returns the observed number of variants, the expected number of variants, enrichment (= *observed/expected*), and the p value (obtained from a Poisson test).

The output can be customized using the “includeClasses” argument, either to display only a subset of variant classes of interest

```
denovolyzeByClass(genes=autismDeNovos$gene,
                  classes=autismDeNovos$class,
                  nsamples=1078,
                  includeClasses=c(“mis”,”lof”))
##   class  observed  expected  enrichment     pValue
## 1  mis      655    679.0      0.965   0.826000
## 2   lof      131     94.3      1.390   0.000199
```

or to display increased granularity. By default, nonsense, frameshift & splice variants are analyzed in combination as “lof”, but may be analyzed separately.

```
denovolyzeByClass(genes=autismDeNovos$gene,
                  classes=autismDeNovos$class,
                  nsamples=1078,
                  includeClasses=c(“frameshift”,”non”,”splice”,”lof”))
##        class  observed  expected  enrichment   pValue
## 1       non       52      34.8       1.500 0.003740
## 2      splice       15      16.1       0.934 0.638000
## 3  frameshift       64      43.4       1.470 0.002070
## 4        lof      131      94.3       1.390 0.000199
```

Further information on function options, and help generally, is available using the help function.

```
help(denovolyzeByClass)
```

BASIC PROTOCOL 2: ASSESSING THE NUMBER OF GENES WITH MULTIPLE *DE NOVO* VARIANTS

The occurrence of multiple *de novo* events in a single gene, in a cohort of individuals with a common phenotype, may implicate that gene in the pathogenesis of the condition under study. Before evaluating single genes, it is instructive to assess the total number of genes harboring multiple *de novo* variants. Here, the number of genes containing multiple *de novo* variants is compared with an empirical distribution derived by permutation.

Materials

As for protocol 1

1) Ensure the **denovolyzeR** library and data for analysis are loaded.

```
library(denovolyzeR)
```

2) The denovolyzeMultiHits function will perform the required analysis. The same three arguments are required as for BASIC PROTOCOL 1: genes (vector of genes containing *de novo* variants), classes (a vector of variant consequences) and nsamples (number of samples). In addition, nperms determines the number of permutations run (defaults to 100).

The function addresses the questions “given nVars variants in a set of genes, how many genes are expected to contain more than one variant? Do we observe more than this?”

```
denovolyzeMultiHits(genes=autismDeNovos$gene,
                   classes=autismDeNovos$class,
                   nsamples=1078,
                   nperms=100)
##      obs  expMean  expMax  pValue  nVars
##  syn   3       3.6      11    0.62   254
##  mis  31      20.3      31    0.04   655
##  lof   5       1.2       5    0.01   131
## prot  43      29.0      43    0.01   786
##  all   66     47.4       64    0.00  1040
```

For each variant class, the function returns the observed number of genes containing multiple *de novo* variants in the user data provided (“obs”), the average number of genes containing multiple hits across nperms permutations (“expMean”), the maximum number of genes containing multiple hits in any permutation (“expMax”), and an empirical p value (“pValue”). In this case some of the p values are returned as 0, indicating < 1/*nperms* (in this case <0.01). We can obtain a better estimate by increasing the number of permutations:

```
denovolyzeMultiHits(genes=autismDeNovos$gene,
                   classes=autismDeNovos$class,
                   nsamples=1078,
                   nperms=5000,
                   includeClasses=“prot”)
##      obs  expMean  expMax  pValue nVars
##  prot  43      28.1      46  0.0026   786
```

Note that the exact numbers may change slightly between runs of denovolyzeMultiHits due to stochastic changes in the permutations. These stochastic fluctuations are likely to be small for large numbers of permutations. Finally, the function reports the total number of *de novo* variants of each class, which is the number used as input to the permutation (“nVars”).

3) This function can be run in two modes. The expected number of genes containing >1 hit is obtained by permutation: given nVars *de novo* variants, how many genes contain >1 variant? There are two options for selecting nVars. By default, this number is derived from the input data - in other words, the total number of lof variants that are permuted across the defined gene list is the total number of lof variants in the input data. An alternative approach uses the expected number of lof variants in the gene list, as determined by the model.

In the example above autismDeNovos contains 131 lof variants, so by default this is the number used in the permutation:

```
sum(autismDeNovos$class %in% c(“frameshift”,”non”,”splice”))
##  [1] 131
denovolyzeMultiHits(genes=autismDeNovos$gene,
                   classes=autismDeNovos$class,
                   nsamples=1078,
                   includeClasses=“lof”)
##    obs  expMean  expMax  pValue  nVars
## lof   5      0.9      5    0.01   131
```

The expected number of *de novo* variants is controlled by the nVars argument, whose default value is “actual”. This is a conservative approach, addressing the question: “given the number of variants in our dataset, do we see more genes with >1 variant than expected?” An alternative approach simply asks whether there are more genes with >1 variant than our *de novo* model predicts. This is accessed by setting nVars=“expected”.

```
denovolyzeMultiHits(genes=autismDeNovos$gene,
                   classes=autismDeNovos$class,
                   nsamples=1078,
                   includeClasses=“lof”,
                   nVars=“expected”)
##     obs  expMean  expMax  pValue    nVars
##  lof   5      0.5       3       0 94.26139
```

BASIC PROTOCOL 3: ASSESSING THE FREQUENCY OF *DE NOVO* VARIANTS IN INDIVIDUAL GENES

In the previous protocol, we assessed whether there were more genes containing multiple *de novo* variants than expected by chance. In the example data, we noted five genes with multiple loss-of-function hits. In this next protocol, we will determine whether any individual genes carry an excess of *de novo* variants, using the denovolyzeByGene function.

Materials

As for protocol 1

1. Ensure the **denovolyzeR** library and data for analysis are loaded.

```
library(denovolyzeR)
```

2. Call the denovolyzeByGene function. The same three arguments are required as for the previous protocols: genes (vector of names of genes containing *de novo* variants), classes (a vector of variant consequences) and nsamples (number of samples). This function will return one row per gene, ordered according the significance of any enrichment in *de novo* variants. Given the size of the data, we will only view the first few lines here, using the head function.

```
head(
  denovolyzeByGene(genes=autismDeNovos$gene,
                   classes=autismDeNovos$class,
                   nsamples=1078)
  )
##               lof.obs  lof.exp  lof.pValue  prot.obs  prot.exp
prot.pValue
##  DYRK1A         3      0   2.69e-08       3      0.1    2.77e-05
##  SCN2A          3      0   1.83e-06       5      0.1    3.70e-07
##  CHD8            3      0   7.19e-07       4      0.2    2.44e-05
##  RFX8            0      0   1.00e+00       2      0.0    2.34e-05
##  SUV420H1        1      0   6.37e-03       3      0.1     3.17e-05
##  POGZ            2       0   1.23e-04       2      0.1     5.07e-03
```

denovolyzeR will output one line for every gene that contains at least one variant in the input data. In order to view only genes with multiple hits, we can use the subset function to select genes with more than one observed protein-altering variant:

```
library(dplyr)
denovolyzeByGene(genes=autismDeNovos$gene,
                 classes=autismDeNovos$class,
                 nsamples=1078) %>%
  subset(prot.obs>1)
##            lof.obs  lof.exp  lof.pValue  prot.obs  prot.exp prot.pValue
##  DYRK1A      3     0.0    2.69e-08       3      0.1    2.77e-05
##  SCN2A        3     0.0    1.83e-06       5      0.1    3.70e-07
##  CHD8         3     0.0    7.19e-07       4      0.2    2.44e-05
##  RFX8          0     0.0    1.00e+00       2      0.0    2.34e-05
##  SUV420H1     1     0.0    6.37e-03       3      0.1    3.17e-05
##  POGZ         2     0.0    1.23e-04       2      0.1    5.07e-03
##  ARID1B       2     0.0    1.64e-04       2      0.2    1.13e-02
##  PLEKHA8      0     0.0    1.00e+00       2      0.0    4.47e-04
##  TUBA1A       0     0.0    1.00e+00       2      0.0    5.37e-04
##  SLCO1C1      0     0.0    1.00e+00       2      0.0    7.41e-04
##  NTNG1        0     0.0    1.00e+00       2     0.0     8.20e-04
##  TSNARE1      0     0.0    1.00e+00       2     0.0    1.20e-03
##  MEGF11       0     0.0    1.00e+00       2      0.1    1.88e-03
##  SRBD1         0     0.0    1.00e+00       2      0.1    1.92e-03
##  TBR1          1     0.0    5.25e-03       2     0.1     1.93e-03
##  KRBA1        0     0.0    1.00e+00       2     0.1     2.02e-03
##  KIRREL3       0     0.0    1.00e+00       2     0.1     2.15e-03
##  NR3C2         1     0.0    8.62e-03       2     0.1     2.18e-03
##  ABCA13       0     0.0    1.00e+00       3      0.3     2.71e-03
##  UBE3C         0     0.0    1.00e+00       2      0.1    3.28e-03
##  SETD5         0     0.0    1.00e+00       2      0.1    3.91e-03
##  AGAP2         0     0.0    1.00e+00       2      0.1    4.21e-03
##  ZNF423        0     0.0    1.00e+00       2      0.1     5.71e-03
##  GSE1          0     0.0    1.00e+00       2      0.1     5.74e-03
##  ZNF638        1     0.0    1.49e-02       2      0.1    6.61e-03
##  ADCY5         0     0.0    1.00e+00      2      0.1     6.71e-03
##  SCN1A         0     0.0    1.00e+00       2      0.1     9.09e-03
##  LAMB2        0     0.0    1.00e+00       2      0.2     1.15e-02
##  MYO7B        0     0.0    1.00e+00       2      0.2     1.16e-02
##  PLXNB1        1     0.0    1.37e-02       2      0.2     1.20e-02
##  ZFYVE26       1     0.0    2.01e-02       2      0.2     1.33e-02
##  SBF1           0     0.0    1.00e+00       2      0.2     1.34e-02
##  BRCA2         0     0.0    1.00e+00       2      0.2    1.48e-02
##  TRIO           0     0.0    1.00e+00       2      0.2     2.32e-02
##  ALMS1         0     0.0    1.00e+00       2      0.2     2.39e-02
##  RELN          1     0.0    3.91e-02       2      0.2     2.63e-02
##  ANK2          1     0.0    3.27e-02       2      0.3     2.85e-02
##  KMT2C         1     0.1    5.09e-02       2      0.3     4.33e-02
##  FAT1           0     0.0    1.00e+00       2      0.3    4.44e-02
##  GPR98          0     0.0    1.00e+00       2      0.4    5.21e-02
##  AHNAK2       0     0.0    1.00e+00       2      0.4    6.25e-02
##  MUC5B         0     0.0    1.00e+00       2      0.4    7.34e-02
##  SYNE1         0     0.1    1.00e+00       2      0.6    1.31e-01
```

In this example we have used the pipe notation “%>%” to pass the output of **denovolyzeR** to the subset function. The pipe is available as part of the dplyr package, which is required for **denovoloyzeR** installation.

The p-values returned are not corrected for multiple testing. These default options apply two tests (“lof” and “prot”) across 19618 genes, so a Bonferroni corrected p-value threshold at $\alpha = 0.05$ would be 1.3×10^{-06} (0.05/2 * 19618).

By default this function compares the number of lof variants against expectation for each gene, and then the total number of protein-altering variants (lof + missense). It can also be configured to return other classes if relevant, using the includeClasses argument.

```
head(
   denovolyzeByGene(genes=autismDeNovos$gene,
                    classes=autismDeNovos$class,
                    nsamples=1078,
                    includeClasses=“syn”)
   )
##           syn.obs  syn.exp  syn.pValue
## PBLD          2       0    3.04e-05
## ADNP2        2       0    4.96e-04
## SPRR2D       1       0    1.70e-03
## C1ORF146     1       0    2.78e-03
## PTMS         1       0    2.78e-03
## RBM20        1       0    3.49e-03
```

BASIC PROTOCOL 4: ASSESSING A PRE-SPECIFIED GENE SET

This protocol assesses whether a pre-specified set of genes collectively shows an enrichment of *de novo* variants. Note that any of the previous analyses can be restricted to a pre-specified gene set in the same way, using the includeGenes argument. This may be appropriate if a smaller panel of genes have been sequenced (rather than whole exome sequencing), or to explore biologically relevant gene sets, e.g. defined by gene ontology, or expression profile.

Materials

As for protocol 1

1. Ensure the **denovolyzeR** library and data for analysis are loaded.

```
library(denovolyzeR)
```

2. Define a gene set. This should be a vector of genes, which may be entered by hand, or read from file using read.table or equivalent. In this example, we use an example gene set included with the **denovolyzeR** package, a list of 837 genes that interact with the fragile X mental retardation protein (FMRP), taken from (Darnell et al., 2011).

```
nrow(fmrpGenes);head(fmrpGenes)
##  [1] 837
##              ensgID            enstID hgncID  hgncSymbol geneName
## 1  ENSG00000142599 ENST00000337907   9965       RERE     RERE
## 2  ENSG00000149527 ENST00000449969  29037      PLCH2    PLCH2
## 3  ENSG00000078369 ENST00000378609   4396       GNB1     GNB1
## 4  ENSG00000157933 ENST00000378536  10896         SKI       SKI
## 5  ENSG00000171735 ENST00000303635  18806    CAMTA1  CAMTA1
## 6  ENSG00000188157 ENST00000379370    329      AGRN     AGRN
```

3. Evaluate the frequency of *de novo* events in our pre-specified genelist, using the denovolyzeByClass function. Specify the genelist using the includeGenes argument, which defaults to “all”, but accepts a vector of genes.

```
denovolyzeByClass(genes=autismDeNovos$gene,
                    classes=autismDeNovos$class,
                    nsamples=1078,
                    includeGenes=fmrpGenes$geneName)
##   class observed expected enrichment  pValue
## 1  syn       28     33.6      0.835 8.53e-01
## 2  mis       83     74.4      1.110 1.74e-01
## 3   lof       32      9.1      3.500 3.18e-09
## 4  prot      115     83.6      1.380 6.47e-04
## 5   all      143    117.1      1.220 1.13e-02
```

In this example we see a highly significant enrichment of *de novo* lof variants in genes that interact with FMRP in our cohort of autism cases. Care should be taken to ensure that the same gene identifiers are used throughout the analysis. For example, if the list of genes containing *de novo* variants includes KMT2D (previously known as MLL2) but the gene set uses the old symbol MLL2, these will not be matched. The function will give a warning if gene identifiers are used that are not found in the internal mutation probability tables.

For many genes, the Ensembl gene name and HGNC symbol will be identical, but in some instances they differ (e.g. where there is no HGNC identifier, and Ensembl uses a symbol from another source). Note that we receive a warning if we pass a list of genes described using Ensembl gene symbols (the demonstration data), but tell the software to match to HGNC symbols.

```
denovolyzeByClass(genes=autismDeNovos$gene,
                    classes=autismDeNovos$class,
                    nsamples=1078,
                    geneId=”hgncSymbol”)
## Warning in parseInput(genes, classes, nsamples, groupBy,
includeGenes,
## includeClasses, : 3 gene identifiers in input list do not match the
## probability table, and are excluded from analysis.
##   class observed expected enrichment   pValue
## 1  syn      254    302.3      0.840 0.998000
## 2  mis      652    679.0      0.960 0.854000
## 3   lof      131     94.3      1.390 0.000199
## 4  prot      783    773.2      1.010 0.368000
## 5   all    1037   1075.5      0.964 0.883000
```

Similarly, we will get a warning if “includeGenes” contains non-matching identifiers

```
denovolyzeByClass(genes=autismDeNovos$gene,
                    classes=autismDeNovos$class,
                    nsamples=1078,
                    includeGenes=fmrpGenes$enstID)
## Warning in parseInput(genes, classes, nsamples, groupBy,
includeGenes,
## includeClasses, : 837 gene identifiers in “includeGene” are not in
the
## probability table, and are excluded from analysis.
## [1] class observed expected enrichment pValue
## <0 rows> (or 0-length row.names)
```

SUPPORT PROTOCOL 1: GETTING HELP

Help on any of the functions described is available using the standard R help functions, e.g. help(denovolyze) or ?denovolyze. Additional details are also available in the package vignette, accessed using browseVignettes(“denovolyzeR”).

SUPPORT PROTOCOL 2: VIEWING THE MUTATIONAL PROBABILITY TABLES

Users may want to view or export the probability tables that underpin these analyses. These are best accessed using the viewProbabilityTable function.

```
probabilityTable <- viewProbabilityTable()
nrow(probabilityTable); head(probabilityTable)
## [1] 19618
##    hgncID hgncSymbol            enstID            ensgID    geneName
syn
## 1       5       A1BG  ENST00000263100  ENSG00000121410       A1BG
8.997970e-06
## 2      7        A2M  ENST00000318602  ENSG00000175899        A2M
```

```
1.543159e-05
## 3     16   SERPINA3  ENST00000467132  ENSG00000196136    SERPINA3
5.694582e-06
## 4     17      AADAC  ENST00000232892  ENSG00000114771      AADAC
4.252483e-06
## 5     18       AAMP  ENST00000248450  ENSG00000127837       AAMP
6.496774e-06
## 6     19      AANAT  ENST00000250615  ENSG00000129673      AANAT
3.530488e-06
##           mis          non        splice     frameshift          lof
## 1  1.738961e-05  5.763794e-07  2.639868e-07  6.532817e-07  1.493648e-06
## 2  3.545894e-05  1.960148e-06  1.477263e-06  4.616751e-07  3.899086e-06
## 3  1.176919e-05  4.433874e-07  1.387157e-07  5.276555e-07  1.109759e-06
## 4  1.018458e-05  5.312742e-07  1.748982e-07  1.051152e-06  1.757324e-06
## 5  1.313861e-05  5.042914e-07  4.247556e-07  3.344019e-06  4.273066e-06
## 6  7.729807e-06  1.707018e-07  9.988864e-08  4.132204e-07  6.838108e-07
##           prot           all
## 1  1.888326e-05  2.788123e-05
## 2  3.935803e-05  5.478962e-05
## 3  1.287895e-05  1.857353e-05
## 4  1.194191e-05  1.619439e-05
## 5  1.741167e-05  2.390845e-05
## 6  8.413618e-06  1.194411e-05
```

This may be useful, for example, to verify that the input gene list contains the correct identifiers

```
#Count the number of input gene names
length(autismDeNovos$gene)
## [1] 1040
#Count how many are in the “geneName” column of the probability table:
sum(autismDeNovos$gene %in% probabilityTable$geneName)
## [1] 1040
#Count how many are in the “hgncSymbol” column of the probability
table:
sum(autismDeNovos$gene %in% probabilityTable$hgncSymbol)
## [1] 1037
#Count how many are in the “enstID” column of the probability table:
sum(autismDeNovos$gene %in% probabilityTable$enstID)
## [1] 0
```

SUPPORT PROTOCOL 3: USING AN ALTERNATIVE MUTATIONAL PROBABILITY TABLE

denovolyzeR relies on a pre-computed tabulation of the probability of *de novo* variation arising in each gene, as described in the Introduction and Background Information. The default probability table was generated by calculating the probability of *de novo* events for every base of the canonical Gencode transcripts, as defined in Gencode v19. It is beyond the scope of this protocol to describe methods to compute these tables, but **denovolyzeR** does allow for the import of alternative tables, if required. For example, the original paper describing this analytical framework (Samocha et al., 2014) calculated mutational probabilities based on RefSeq transcript definitions, whereas **denovolyzeR** now uses Gencode definitions. Tables may also be computed to include other functional consequences (e.g. damaging missense variants, as determined by *in silico* SNV consequence prediction algorithms).

Materials

An alternative probability table. Examples are available to download from http://denovolyzer.org/

1. Locate and load the chosen probability table. For this example, we will use “probTable_Samocha2014.rda” downloaded from the above link, which is now located in our Downloads folder:

```
#pathToProbabilityTable=”~/Downloads” #replace this with the path on
your local system
load(file.path(pathToProbabilityTable,”probTable_Samocha2014.rda”))
head(probTable_Samocha2014)
##         refseqID     geneID class         value
## 1    NM_017582   UBE2Q1   syn  5.428059e-06
## 2    NM_014372     RNF11   syn  2.306612e-06
## 3    NM_014455    RNF115   syn  3.153658e-06
## 4    NM_001357     DHX9   syn  1.205980e-05
## 5  NM_001101376 FAM183A   syn  1.572484e-06
## 6  NM_001042549     NSL1   syn  2.643121e-06
```

This table has two sets of gene identifiers: “refseqID””, and gene symbols (“geneID”).

2. Run chosen analysis, specifying the chosen probability table using the “probTable” argument, and the appropriate gene identifier.

```
denovolyzeByClass(genes=autismDeNovos$gene,
                    classes=autismDeNovos$class,
                    nsamples=1078,
                    probTable=probTable_Samocha2014,
                    geneId=”geneID”)
## Warning in parseInput(genes, classes, nsamples, groupBy,
includeGenes,
## includeClasses, : 43 gene identifiers in input list do not match
the
## probability table, and are excluded from analysis.
##   class  observed  expected  enrichment    pValue
## 1  syn      247    295.8      0.835  0.998000
## 2  mis      625    668.6      0.935  0.957000
## 3   lof      125     92.4      1.350  0.000721
## 4  prot      750    761.0      0.985  0.660000
## 5   all      997   1056.9      0.943  0.969000
```

In this instance there is a warning that not all of the input identifiers are recognized. This is because there is not a one-to-one mapping between the identifiers associated with RefSeq and Gencode transcripts.

COMMENTARY

Background Information

The mutational model is described in full detail in (Samocha et al., 2014). Briefly, it is based on a determination of the probability of each base in the coding sequence of the human genome mutating to each of the other possible bases. The predicted impact of these changes is aggregated across the gene to establish the probability of specific types of mutation (synonymous, missense, etc).

Previous work established that the mutability of a base is sufficiently modeled by accounting for the local sequence context of one nucleotide on either side of the base of interest (Krawczak et al., 1998; Kryukov et al., 2007). We analyzed human variation and trinucleotide context using intergenic single nucleotide polymorphisms (SNPs) from the 1000 Genomes project to create a mutation rate table, which provides the relative probabilities of each possible trinucleotide (XY^1Z) to trinucleotide (XY^2Z) change.

We then consider each base in the coding sequence and use the mutation rate table to determine its probability of mutating to the other bases. We predict the impact of the mutation on the protein product and aggregate the probabilities by mutation type across each gene. These per-gene probabilities are then adjusted according to the completeness of sequencing coverage for each gene, and a regional divergence score, reflecting divergence between humans and macaques, that captures small regional differences in genome mutability that are not fully captured by local trinucleotide context.

Given that the number of *de novo* variants per trio follows a Poisson distribution (Neale et al., 2012), we use the Poisson distribution to evaluate the excesses of *de novo* events. As an example, to determine if a particular gene has more *de novo* loss-of-function variants than expected, we multiply that gene’s probability of a loss-of-function mutation by the number of trios and by 2 (for the number of chromosomes) in the study. This gives the expected number of *de novo* loss-of-function variants (denoted as lambda, λ) with which the observed number is compared. Specifically, the ppois command in R is used. The three *de novo* loss-of-function variants seen in DYRK1A are used in the example below. With denovolyze, we get the following result:

```
denovolyzeByGene(genes=autismDeNovos$gene,
                    classes=autismDeNovos$class,
                    nsamples=1078,
                    includeGenes=”DYRK1A”)
##       gene  class obs  exp   pValue
## 1  DYRK1A   lof   3  0.0 2.69e-08
## 2  DYRK1A  prot   3  0.1 2.77e-05
```

We can reproduce this with the ppois function. Note that by default, ppois(q,lambda) will return the probability of observing $\leq q$ events for a given λ. We are interested in computing $p(obs \geq q)$. ppois(q,lambda,lower.tail=FALSE) gives us $p(obs > q)$, and therefore we must use ppois(q-1,lambda,lower.tail=FALSE) to obtain $p(obs \geq q)$.

```
n_lof_dyrk1a <- 3
probabilityTable[probabilityTable$geneName==”DYRK1A”,”lof”]
## [1] 2.528297e-06
prob_lof_dyrk1a <- 2.528297e-06
n_trios <- 1078
ppois(q=n_lof_dyrk1a-1, #observed - 1
      lambda=prob_lof_dyrk1a*n_trios*2, #expected
      lower.tail=FALSE)
## [1] 2.688463e-08
```

Control subjects—Since this analytic framework compares *de novo* events to a model-derived expectation, there is no direct comparison of cases with controls. As described above, direct case-control comparison is not statistically powerful in this context. It is nonetheless recommended to include a control arm in analyses, for example by repeating the analyses described above on a cohort of controls which have been subjected to the same sequencing and *de novo* identification pipeline. While cases and controls are not directly compared, it is valuable to confirm that the burden of variants in the control population does not deviate substantially from model-derived expectations, in order to validate the model for the specific sequencing approaches used in each study.

Large variants—This analytical framework assesses single nucleotide variants and small (single basepair) insertions and deletions only. Larger insertions, deletions and other structural variants are not assessed.

Critical Parameters

The number of samples (nsamples) should be the total number of samples in the study, not just those that carry *de novo* variants.

The analysis described in BASIC PROTOCOL 2 is highly sensitive to the choice of argument passed to nVars. Full details are provided in step 3 of that protocol.

Troubleshooting

This methodology is dependent on accurate upstream identification of *de novo* variants. Mendelian violations (putative variants in sequence data that do not follow normal Mendelian inheritance patterns) comprise technical artefacts and sequencing errors as well as true *de novo* variants. Indeed, a set of variants specifically selected as Mendelian violations will inevitably be enriched for such errors. A detailed description of strategies for accurate *de novo* variant detection is beyond the scope of this protocol, but stringent quality control should be applied. Variant confirmation with two independent technologies (e.g. next-generation sequencing and Sanger sequencing) remains the gold standard.

It is also recommended to apply the same variant detection and analysis pipeline to a cohort of control trios as outlined above.

Time Considerations

These analyses are not especially computationally intensive, and will run on a desktop or laptop computer in seconds. The denovolyzeMultiHits function uses permutation, and computation time increases linearly with the number of permutations. Elapsed times (in seconds) to run the three principal functions on *de novo* variants from 1078 samples, using default settings, on a MacBook Air (1.7GHz i7, 8Gb RAM) are as follows:

```
system.time(denovolyzeByClass(genes=autismDeNovos$gene,classes=autismDe
Novos$class,nsamples=1078))[“elapsed”]
## elapsed
##  0.136
system.time(denovolyzeMultiHits(genes=autismDeNovos$gene,classes=autism
DeNovos$class,nsamples=1078,nperms=1000))[“elapsed”]
## elapsed
##  5.397
system.time(denovolyzeByGene(genes=autismDeNovos$gene,classes=autismDeN
ovos$class,nsamples=1078))[“elapsed”]
## elapsed
##  0.159
```

Acknowledgments

J.S.W. is supported by the NIHR Biomedical Research Unit in Cardiovascular Disease at Royal Brompton & Harefield NHS Foundation Trust and Imperial College London, Academy of Medical Sciences, Wellcome Trust, British Heart Foundation Arthritis Research U.K., Fondation Leducq, and an HICF award from Wellcome Trust and Department of Health. J.G.H. is supported by the John S. LaDue Fellowship at Harvard Medical School. M.J.D. is supported by grants R01MH089208 (NIH) and 1U01MH100229-01 (NIH/NIMH).

LITERATURE CITED

Conrad DF, Keebler JE, DePristo MA, Lindsay SJ, Zhang Y, Casals F, Idaghdour Y, Hartl CL, Torroja C, Garimella KV, Zilversmit M, Cartwright R, Rouleau GA, Daly M, Stone EA, Hurles ME, Awadalla P. Variation in genome-wide mutation rates within and between human families. Nature Genetics. 2011; 43:712–714. [PubMed: 21666693]

Cunningham F, Amode MR, Barrell D, Beal K, Billis K, Brent S, Carvalho-Silva D, Clapham P, Coates G, Fitzgerald S, Gil L, Giron CG, Gordon L, Hourlier T, Hunt SE, Janacek SH, Johnson N, Juettemann T, Kahari AK, Keenan S, Martin FJ, Maurel T, McLaren W, Murphy DN, Nag R, Overduin B, Parker A, Patricio M, Perry E, Pignatelli M, Riat HS, Sheppard D, Taylor K, Thormann A, Vullo A, Wilder SP, Zadissa A, Aken BL, Birney E, Harrow J, Kinsella R, Muffato M, Ruffier M, Searle SM, Spudich G, Trevanion SJ, Yates A, Zerbino DR, Flicek P. Ensembl 2015. Nucleic Acids Research. 2015; 43:D662–669. [PubMed: 25352552]

Darnell JC, Van Driesche SJ, Zhang C, Hung KY, Mele A, Fraser CE, Stone EF, Chen C, Fak JJ, Chi SW, Licatalosi DD, Richter JD, Darnell RB. FMRP stalls ribosomal translocation on mRNAs linked to synaptic function and autism. Cell. 2011; 146:247–261. [PubMed: 21784246]

Durinck S, Spellman PT, Birney E, Huber W. Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Nature Protocols. 2009; 4:1184–1191. [PubMed: 19617889]

Krawczak M, Ball EV, Cooper DN. Neighboring-nucleotide effects on the rates of germ-line single-base-pair substitution in human genes. American Journal of Human Genetics. 1998; 63:474–488. [PubMed: 9683596]

Kryukov GV, Pennacchio LA, Sunyaev SR. Most rare missense alleles are deleterious in humans: implications for complex disease and association studies. American Journal of Human Genetics. 2007; 80:727–739. [PubMed: 17357078]

Lynch M. Rate, molecular spectrum, and consequences of human mutation. Proceedings of the National Academy of Sciences of the United States of America. 2010; 107:961–968. [PubMed: 20080596]

Neale BM, Kou Y, Liu L, Ma'ayan A, Samocha KE, Sabo A, Lin CF, Stevens C, Wang LS, Makarov V, Polak P, Yoon S, Maguire J, Crawford EL, Campbell NG, Geller ET, Valladares O, Schafer C, Liu H, Zhao T, Cai G, Lihm J, Dannenfelser R, Jabado O, Peralta Z, Nagaswamy U, Muzny D, Reid JG, Newsham I, Wu Y, Lewis L, Han Y, Voight BF, Lim E, Rossin E, Kirby A, Flannick J, Fromer M, Shakir K, Fennell T, Garimella K, Banks E, Poplin R, Gabriel S, DePristo M, Wimbish JR, Boone BE, Levy SE, Betancur C, Sunyaev S, Boerwinkle E, Buxbaum JD, Cook EH Jr, Devlin B, Gibbs RA, Roeder K, Schellenberg GD, Sutcliffe JS, Daly MJ. Patterns and rates of exonic de novo mutations in autism spectrum disorders. Nature. 2012; 485:242–245. [PubMed: 22495311]

Ng SB, Bigham AW, Buckingham KJ, Hannibal MC, McMillin MJ, Gildersleeve HI, Beck AE, Tabor HK, Cooper GM, Mefford HC, Lee C, Turner EH, Smith JD, Rieder MJ, Yoshiura K, Matsumoto N, Ohta T, Niikawa N, Nickerson DA, Bamshad MJ, Shendure J. Exome sequencing identifies MLL2 mutations as a cause of Kabuki syndrome. Nature Genetics. 2010; 42:790–793. [PubMed: 20711175]

R Core Team. R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing; 2015.

Samocha KE, Robinson EB, Sanders SJ, Stevens C, Sabo A, McGrath LM, Kosmicki JA, Rehnstrom K, Mallick S, Kirby A, Wall DP, MacArthur DG, Gabriel SB, DePristo M, Purcell SM, Palotie A, Boerwinkle E, Buxbaum JD, Cook EH Jr, Gibbs RA, Schellenberg GD, Sutcliffe JS, Devlin B, Roeder K, Neale BM, Daly MJ. A framework for the interpretation of de novo mutation in human disease. Nature Genetics. 2014; 46:944–950. This paper provides the first exposition of the analytical framework implemented in the denovolyzeR package, and demonstrates the application of this framework to the study of autism spectrum disorders and intellectual disability. [PubMed: 25086666]