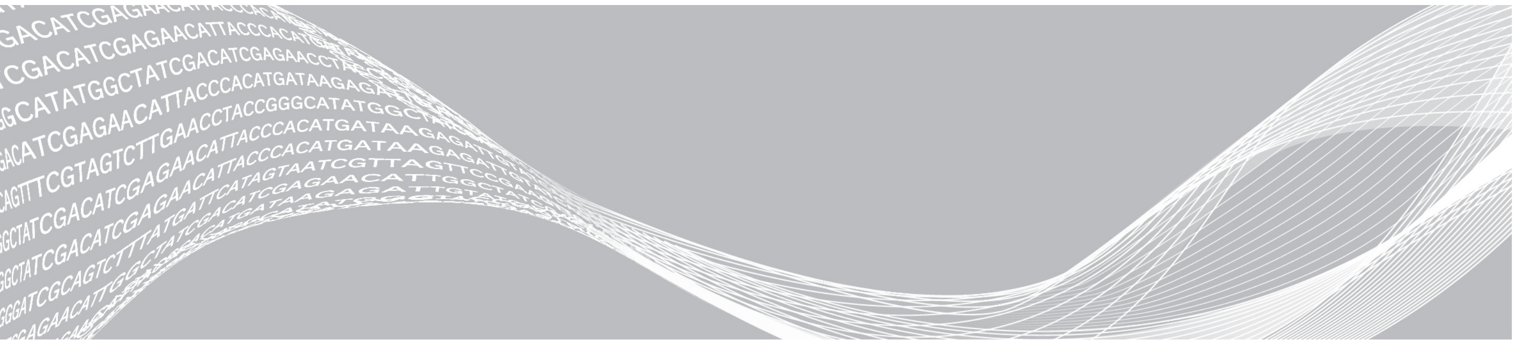
CYP2D6 CNV Caller

User Guide



Document # 1000000119287 v00
May 2020

Introduction

The CYP2D6 CNV Caller, in conjunction with the Global Screening Array, provides the basis for a deeper interrogation of the role of CYP2D6 in drug metabolic pathways.

Caller Capability

The caller module allows you to call copy number status at the gene level and in intron 2, intron 6, and exon 9 of CYP2D6 leveraging markers present in the Infinium Global Screening Array v3.0. The caller can detect copy number (0, 1, 2, 3, 4 and 5+) status. In addition to CNV (Copy Number Variation) calling, the software can also call genotypes for the entire chip and output them in a VCF (Variant Call Format) file that can be used in a downstream analysis package. Auxiliary summary files are generated to provide metrics that can be used to assess the quality of the data.

Trainer Capability

The trainer module provides the user with the ability to generate a new model file (CN Model). This step is recommended when customers:

- Have optimized the commercial GSA v3 cluster file using their training sample set.
- Have generated a new cluster file for their semi-custom GSA v3 beadchip using their training sample set.

Additional information on the cluster generation process can be found at:

https://www.youtube.com/embed/4JTrbMUbVN0

Prerequisites

CNV Caller should be installed on a dedicated system that meets the following minimum host requirements:

Requirement	Minimum
Processor	64-bit
Cores	2 or more
Memory	16 GB RAM
Operating System	“Portable" Linux (linux-x64) supports: • Red Hat Enterprise Linux 7 (x64) • Red Hat Enterprise Linux 6 (x64) • CentOS 7 (x64) • Oracle Linux 7 (x64) • Fedora 28 (x64) • Debian 9 (x64) • Ubuntu 18.04, 16.04, 14.04 (x64) • Linux Mint 18, 17 (x64) • openSUSE 42.3, 15 (x64)SUSE Enterprise Linux (SLES) 12, 15 (SP2, SP3, SP4) (x64) Mac OSX (osx-x64) • OSX 10.13 High Sierra (x64) Win 10 (win10-x64) • Windows 10 Anniversary Update (version 1607) or later versions • Windows Server 2016 or later versions (Full Server, Server Core, or Nano Server)

NOTE

ARM or 32-bit architectures are not supported.

If shared libraries related to the .NET core runtime have not been installed prior to running CYP2D6 CNV Caller, an error might result. See *Troubleshooting* on page 13 for a detailed listing of possible errors.

Installing CYP2D6 CNV Caller

Use the following procedure to install CYP2D6 CNV Caller:

1 Navigate to the support page:
https://support.illumina.com/array_software/cyp2d6-cnv-caller.html

2 Select the installation package for the platform of your choice and download the installation package to your computer.

3 You will be prompted for your MyIllumina credentials

4 To install CYP2D6 CNV Caller, unzip the CYP2D6CNVCaller-cli package to the desired folder.

5 To confirm that the installation has been successful:

a Open a command prompt (Windows) or terminal (Linux and Mac).
b Execute the following command:

```
/path/to/cyp2d6-cnv-caller/cyp2d6-cnv-caller version
```

If the installation was successful, the software version will display. For example:

```
cyp2d6-cnv-caller 2.0.0.x
```

Obtaining FASTA and Index Files

From the CYP2D6 CNV Caller support page:

1 Download the genome FASTA (fa.gz) for your product. Content can be designed for either Build 37 or Build 38 of the human genome.
- For a commercial product:
  - A1 in the manifest name indicates that the content was designed for Build 37, so Build 37 of the FASTA files must be downloaded
  - A2 in the manifest name indicates that the content was designed for Build 38, so Build 38 of the FASTA files must be downloaded.
- For custom products, inspect the manifest CSV file. In the "GenomeBuild" column it should specify either "37" or "38".

2 Unzip the genome FASTA into the folder of your choice using a compression tool that recognizes gzip files (e.g., Gunzip, 7-Zip, etc.)

3 Download the index file (fa.fai) matching the version of the genome build selected.

Using CYP2D6 CNV Caller

This section outlines the typical usage of the CYP2D6 CNV Caller, including a detailed description of the following commands:

- *Call Command* on page 5
- *Train Command* on page 10

Usage

In order to successfully execute commands:

1 Open a command prompt (Windows) or terminal window (Mac or Linux).

2 Navigate to the directory where the caller was installed.

The following is the typical usage syntax for CYP2D6 CNV Caller:

```
cyp2d6-cnv-caller [command] [required parameters] [optional parameters]
```

See *Example* on page 6 for an example use of the Call command.

Commands for the cyp2d6-cnv-caller executable include:

Name	Description
call	Command used to invoke the call algorithm on the given set of inputs
train	Command used to invoke the training algorithm to generate a new CN Model for the given inputs
help	Command used to display the help information

Call Command

The following table provides a list of required input and output parameters for the Call command:

Name	Description
--idat-folder	Specifies the path to the directory where all intensity data IDATs (for the samples to be processed) are located. Note that this path will include the contents of all sub-directories as well.
--bpm-manifest	Used to specify the path to the beadpool manifest in BPM format. **NOTE** BPM contains the norm IDs necessary for normalization.
--csv-manifest	Specifies the path to the beadpool manifest in a CSV format. **NOTE** CSV contains the source sequence required for VCF conversion for indels.
--cluster-file	Specifies the path to the EGT cluster file to be used.
--genome-fasta-file	Specifies the path to the genome FASTA file. The genome build version of the FASTA file must match the genome build version (either 37 or 38) used for the design.
--genome-fasta-index-file	Specifies the path to the genome FASTA index file.
--cn-model	Specifies the path to the CN model for the product.
--output-folder	Specifies the path to the folder where the output files will be saved. The output directory structure must match the directory structure of the IDAT folder.

The following table provides a list of optional parameters for the Call command:

Name	Description
--disable-genome-cache	True/False flag specifying whether the genome information should be cached or not. Caching the genome information improves the performance of the software but utilizes memory. (Default: true)
--auxilliary-loci	Path to location of the VCF file with auxiliary definitions of loci. If provided, this information is used to handle multi-nucleotide variations when generating the genotyping VCF file.
--call-rate-threshold	Pass/Fail sample call rate threshold. (Default: - 0.98)
--cn-quality-threshold	PASS/FAIL threshold for sample phred quality score across all CYP2D6 copy number regions targeted by the algorithm. (Default: - 20)
--gencall-cutoff	Gencall score cutoff to deem a NoCall. (Default: 0.15)
--log-r-dev-threshold	PASS/FAIL threshold for sample log r standard deviation. This is used when generating sample summary or analysis summary. (Default: 0.2)

Example

The structure of a typical Call command is shown in the following example:

```
cyp2d6-cnv-caller call --idat-folder ./samplesdata --bpm-manifest ./GSA-
   24-v3-0_A1.bpm --csv-manifest ./GSA-24-v3-0_A1.csv --cluster-file
   ./GSA-24-v3-0_A1_ClusterFile.egt --genome-fasta-file ./genome/genome-
   37.fa --genome-fasta-index-file ./genome/genome-37.fai --cn-model .
   /GSA-24-v3-0_A1_CYP2D6_CNV_model.dat --disable-genome-cache --output-
   folder ./mypgx-analysis
```

Understanding the Output Files

CNV VCF File

One CNV VCF file per sample is generated by the software to report the gene level CN status, along with the CN events for intron2, intron 6, and exon 9. The VCF output file includes the following content, with the last two rows showing status of CYP2D6 CNV Call data:

```
##fileformat=VCFv4.1

##source=cyp2d6-cnv-caller 2.0

##reference=file://genome-37.fa

##FORMAT=<ID=CN,Number=1,Type=Integer,Description="Copy number genotype
   for imprecise events. CN=5 indicates 5 or 5+">

##FORMAT=<ID=NR,Number=1,Type=Float,Description="Aggregated normalized
   intensity">

##ALT=<ID=CNV,Description="Copy number variant region">

##FILTER=<ID=Q7,Description="Quality below 7">

##INFO=<ID=CNVLEN,Number=1,Type=Integer,Description="Number of bases in
   CNV hotspot">
```

```
##INFO=<ID=PROBE,Number=1,Type=Integer,Description="Number of probes
   assayed for CNV hotspot">
##INFO=<ID=END,Number=1,Type=Integer,Description="End position of CNV
   hotspot">
##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Structural Variant
   Type">
##CNVOverallPloidy=2
##CNVGCCorrect=True
##contig=<ID=22,length=51304566>
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT NA17263_2_R
22 42522501 CYP2D6CNV:CYP2D6:chr22:42522501:42526883 N <CNV> 53 PASS
   CIEND=0,0;CIPOS=0,0;CNVLEN=4383;PROBE=349;END=42526883;SVTYPE=CNV CN:NR
   2:0.974178
22 42525188 CYP2D6CNV:CYP2D6.intron.2:chr22:42525188:42525739 N <CNV> 32
   PASS CIEND=0,0;CIPOS=0,0;CNVLEN=552;PROBE=59;END=42525739;SVTYPE=CNV
   CN:NR 2:1.00477
22 42523637 CYP2D6CNV:CYP2D6.intron.6:chr22:42523637:42523843 N <CNV> 5
   Q7 CIEND=0,0;CIPOS=0,0;CNVLEN=207;PROBE=35;END=42523843;SVTYPE=CNV
   CN:NR 2:0.999524
22 42522501 CYP2D6CNV:CYP2D6.exon.9:chr22:42522501:42522754 N <CNV> 58
   PASS CIEND=0,0;CIPOS=0,0;CNVLEN=254;PROBE=23;END=42522754;SVTYPE=CNV
   CN:NR 2:0.976732
```

NOTE
For each Copy Number status reported, a phred-scaled score is computed and output in the CNV VCF.

GT VCF File

The system produces one GT VCF file per sample containing a header, CYP2D6 CNV Call, score, and genotype calls at a minimum. The VCF output file includes the following content, with the last two rows showing examples of CYP2D6 CNV Call data and a GT call data:

```
##fileformat=VCFv4.1
##source= cyp2d6-cnv-caller 2.0.0
##reference=file://genome-37.fa
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=String,Description="GenCall score. For
   merged multi-locus entries, min(GenCall) score is reported.">
##FORMAT=<ID=BAF,Number=1,Type=Float,Description="B Allele Frequency">
##FORMAT=<ID=LRR,Number=1,Type=Float,Description="Log R Ratio">
##contig=<ID=1,length=249250621>
##contig=<ID=2,length=243199373>
##contig=<ID=3,length=198022430>
##contig=<ID=4,length=191154276>
##contig=<ID=5,length=180915260>
```

```
##contig=<ID=6,length=171115067>
##contig=<ID=7,length=159138663>
##contig=<ID=8,length=146364022>
##contig=<ID=9,length=141213431>
##contig=<ID=10,length=135534747>
##contig=<ID=11,length=135006516>
##contig=<ID=12,length=133851895>
##contig=<ID=13,length=115169878>
##contig=<ID=14,length=107349540>
##contig=<ID=15,length=102531392>
##contig=<ID=16,length=90354753>
##contig=<ID=17,length=81195210>
##contig=<ID=18,length=78077248>
##contig=<ID=19,length=59128983>
##contig=<ID=20,length=63025520>
##contig=<ID=21,length=48129895>
##contig=<ID=22,length=51304566>
##contig=<ID=MT,length=16569>
##contig=<ID=X,length=155270560>
##contig=<ID=Y,length=59373566>
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT 202937470021_R06C01
1 567667 rs9651229 C T . PASS . GT:GQ:BAF:LRR 0/0:2:0.009217858:-
   0.2798456
```

NOTE
The genotype data is also reported in GTC files.

QC Summary Files

Sample Analysis File

The system produces a sample analysis summary file containing the following details per sample (with the data for each sample in a single row, and labels in the column header):

- Sample ID
- Sample Name
- Sample Folder Path
- Call Rate
- Log R Ratio Std Dev
- Gender Estimate
- Intron 2 Phred Quality Score
- Intron 2 Phred Quality Score

- Exon 9 Phred Quality Score
- CYP2D6 Genic Region Phred Quality Score
- Pass/Fail

Aggregate Analysis Summary File

The system produces an aggregated analysis summary file containing the following details (with labels in column A and data in column B):

- Total Number of samples
- % Passed
- Total gain for intron 2 across samples
- Total loss for intron 2 across samples
- Total gain for intron 6 across samples
- Total loss for intron 6 across samples
- Total gain for exon 9 across samples
- Total loss for exon 9 across samples
- Total gain for CYP2D6 genic regions across samples
- Total loss for CYP2D6 genic regions across samples

bedGraph Summary File

The system produces a bedGraph file, which is used to display continuous-valued data in track format. This file can be loaded in a visualization tool, such as Integrated Genome Viewer (IGV). The bedGraph track report contains the following information:

Variable	Value
track type	bedGraph
name	"NA17287_2_R.LRR"
maxHeightPixels	200:200:200
graphType	points
viewLimits	-2:2
windowingFunction	none
color	200,100,0
altColor	0,100,200

Warning/Error Messages

The following scenarios result in a warning message:

- The CSV manifest, cluster file, or FASTA files do not match the ones used to generate the CN model.
- FASTA files and FASTA index files are out of sync.

For the following scenarios, the system reports user-actionable messages to the terminal output (as either a warning or an error):

- User uses the wrong beadchip
- User points to input files (e.g., iDAT and BPM files) that do not match (either intentionally or by accident).

- User points to a corrupted input file.
- The input folder does not contain the requisite files.
- An input file is corrupt.

Examples of such notifications may include:

Error	Type	Cause
Reference allele is not queried for locus: {identifier}	Warning	True reference allele does not match any alleles in the manifest (common if no MNV defined in Aux loci VCF).
Skipping non-mapped locus: {locus.Name}	Warning	Locus has no chromosome position, can't find the REF and ALT from the genome.
Skipping indel: {locus.Name}	Warning	If no CSV manifest, all the indels are skipped for VCF generation. Can also occur if the indel context (deletion/insertion) couldn't be determined.
Invalid manifest file path {manifestPath}	Error	Application couldn't find manifest file provided, user error.
Failed to load cluster file: {e.Message}	Error	Corrupt or unreachable cluster file.

Train Command

The train command is used to generate a CN Model. The CN Model is a required input to the CNV Caller.

A new CN Model should be generated by customers in the following two cases:

- The customer has optimized their cluster file using their own GenTrain data set.
- The customer has generated a new cluster file for the GSA v3 (plus add-on content).

In both cases, the train command should be executed using the GenTrain data sets that were used to optimize/generate the cluster file.

A minimum of 96 samples is required for this command.

Required Parameters

The following table provides a list of required input parameters for the Train Command:

Name	Description
--idat-folder	Used to specify the path to the directory that the IDAT files for the GenTrain dataset are located. The GenTrain dataset is the set of samples used to generate the Cluster File.
--bpm-manifest	Used to specify the path to the beadpool manifest in BPM format.
--csv-manifest	Specifies the path to the beadpool manifest in a CSV format.
--cluster-file	Specifies the path to the cluster file for the product being processed.
--genome-fasta-file	Specifies the path to the genome FASTA file. Ensure that the version of the genome build matches the version used to design the product (can either be build GRCh37 or GRCh38).
--genome-fasta-index-file	Specifies the path to the genome FASTA index file.

Optional Parameters

The following table provides a list of optional parameters:

Name	Description
--pad size	Used by the GC Normalization algorithm and set to 10000 by default.
--output folder	The location where to output the CN Model (defaults to the current working directory).

Example

The structure of a typical Train command is shown in the following example:

```
cyp2d6-cnv-caller train --idat-folder ./samplesdata --bpm-manifest ./GSA-
   24-v3-0_A1.bpm --csv-manifest ./GSA-24-v3-0_A1.csv --cluster-file
   ./GSA-24-v3-0_A1_ClusterFile.egt --genome-fasta-file ./genome/genome-
   37.fa --genome-fasta-index-file ./genome/genome-37.fai --output-folder
   ./MyCNModels
```

CN Models

The CN Model file is a binary file containing, but not limited to, the following information:

- GcContents:
  - GcContents
  - Manifest Indexes
  - Hotspot Indexes
  - Cluster File
  - Manifest file path
  - Fasta Contigs
  - Pad Size
- CnModels:
  - Model Parameters
  - Starting Parameters
  - Model Mean Bounds
  - Aggregated Models

The GC Content information is used by the normalization algorithm, while the CN Model information is used by the CN Calling algorithm.

Filename	Format	Location
Commercial Beadpool Manifest	BPM Format – GRCh37	Infinium Global Screening Array v3.0 Manifest File (BPM Format – GRCh37)
Commercial BeadPool Manifest File	CSV Format – GRCh37	Infinium Global Screening Array v3.0 Manifest File (CSV Format – GRCh37)
Commercial Beadpool Manifest File	BPM Format – GRCh38	Infinium Global Screening Array v3.0 Manifest File (BPM Format - GRCh38)
Commercial Beadpool Manifest File	CSV Format – GRCh38	Infinium Global Screening Array v3.0 Manifest File (CSV Format - GRCh38)
Commercial Cluster File	EGT	Infinium Global Screening Array v3.0 Cluster File

The following table describes file formats relevant to the train command, GC Content and CN Model information:

Format	Filename	Description
DAT	Data File	• Standard file format used to store the model file (CYP2D6_CNV_model.dat).
EGT	Cluster file	• This is an input to the genotype calling algorithm. • A cluster file is provided for any commercial product, which is the result of the Illumina GenTrain process. • It is strongly advised that a commercial cluster file be optimized by the customer to their lab processing and sample population. • A cluster file needs to be generated for any custom or semi-custom project. • Cluster file can be generated and optimized using only GenomeStudio.™
BPM	Beadpool Manifest	• It is an input to the genotype calling algorithm. • It is available to download for any commercial product on the product page and is generated for any full or semi-custom products.
IDAT	Raw Intensity Files	• Result from instrument scanning a beadchip. • Two IDAT files generated per sample, one from the red channel, one from the green channel. • Input to the genotype algorithm.
GTC	Genotype Call File	• Result of the genotype call algorithm. • Contains GT for each marker on the product. • Contains sample quality metrics. • Does not contain the marker location which needs to be extracted from the manifest. • Binary proprietary format that can be parsed using Beadarray Library File parser: *https://github.com/Illumina/BeadArrayFiles*
VCF	Variant Call Format	• File format used to store variant call information from sequencing. • Commonly used file format for downstream analysis and to analyze sequencing data. • Provides the ability for array genotyping data and sequencing data to be anlyzed side by side.

Error Messages

The train command reports the following messages to the customer when it encounters an error:

Error	Type	Cause
Reference allele is not queried for locus: {identifier}	Warning	True reference allele does not match any alleles in the manifest (common if no MNV defined in Aux loci VCF).
Skipping non-mapped locus: {locus.Name}	Warning	Locus has no chromosome position, can't find the REF and ALT from the genome.
Skipping indel: {locus.Name}	Warning	If no CSV manifest, all the indels are skipped for VCF generation. Can also occur if the indel context (deletion/insertion) could not be determined.
Invalid manifest file path {manifestPath}	Error	Application could not find manifest file provided, user error.
Failed to load cluster file: {e.Message}	Error	Corrupt or unreachable cluster file.

Troubleshooting

If the ICU package is missing on a Linux system, the following error will result:

```
FailFast: Couldn't find a valid ICU package installed on the system. Set
   the configuration flag System.Globalization.Invariant to true if you
   want to run with no globalization support.


at System.Environment.FailFast(System.String)
at System.Globalization.GlobalizationMode.GetGlobalizationInvariantMode()
at System.Globalization.GlobalizationMode..cctor()
at System.Globalization.CultureData.CreateCultureWithInvariantData()
at System.Globalization.CultureData.get_Invariant()
at System.Globalization.CultureInfo..cctor()
at System.StringComparer..cctor()
at System.AppDomain.InitializeCompatibilityFlags()
at System.AppDomain.Setup(System.Object)


Aborted (core dumped)
```

For Rhel 6 in particular, users should follow the instructions outlined in the *How to use .NET Core on RHEL 6/CentOS 6* article at GitHub to set up the necessary libraries for a .NET Core application.

Technical Assistance

For technical assistance, contact Illumina Technical Support.

Website:	www.illumina.com
Email:	techsupport@illumina.com

Illumina Customer Support Telephone Numbers

Region	Toll Free	Regional
North America	+1.800.809.4566	
Australia	+1.800.775.688	
Austria	+43 800006249	+43 19286540
Belgium	+32 80077160	+32 34002973
China	400.066.5835	
Denmark	+45 80820183	+45 89871156
Finland	+358 800918363	+358 974790110
France	+33 805102193	+33 170770446
Germany	+49 8001014940	+49 8938035677
Hong Kong, China	800960230	
Ireland	+353 1800936608	+353 016950506
Italy	+39 800985513	+39 236003759
Japan	0800.111.5011	
Netherlands	+31 8000222493	+31 207132960
New Zealand	0800.451.650	
Norway	+47 800 16836	+47 21939693
Singapore	+1.800.579.2745	
South Korea	+82 80 234 5300	
Spain	+34 911899417	+34 800300143
Sweden	+46 850619671	+46 200883979
Switzerland	+41 565800000	+41 800200442
Taiwan, China	00806651752	
United Kingdom	+44 8000126019	+44 2073057197
Other countries	+44.1799.534000	

Safety data sheets (SDSs)—Available on the Illumina website at support.illumina.com/sds.html.

Document # 1000000119287 v00

Illumina
5200 Illumina Way
San Diego, California 92122 U.S.A.
+1.800.809.ILMN (4566)
+1.858.202.4566 (outside North America)
techsupport@illumina.com
www.illumina.com