Takara Bio USA

Cogent™ NGS Analysis Pipeline v1.5.1 User Manual

(032522)

Table of Contents

Table of Figures

Table of Tables

I.   Introduction

A.   Welcome to the Cogent NGS Analysis Pipeline Software

Cogent NGS Analysis Pipeline (CogentAP) is bioinformatics software for analyzing RNA-seq NGS data generated using the following systems and kits from Takara Bio:

Table 1. Applications and kits compatible with Cogent NGS Analysis Pipeline v1.5.1.

System	Experiment type	Kit or application
ICELL8® cx Single-Cell System	Single-cell full-length transcriptome analysis	SMART-Seq® Pro Application Kit - 2 Chip
		SMART-Seq ICELL8 cx Application Kit
	Single-cell differential expression analysis	ICELL8 cx 3′ DE
	Human TCR profiling and 5′ DE	ICELL8 cx Human TCR a/b Profiling*
ICELL8 Single-Cell System	Single-cell full-length transcriptome analysis	SMART-Seq ICELL8 Application Kit
	Single-cell differential expression analysis	ICELL8 3′ DE
	Single-cell differential expression analysis with UMIs	ICELL8 3′ DE for UMI Reagent Kit (430-000236)
	Human TCR profiling and 5′ DE	ICELL8 Human TCR a/b Profiling*
Plate-based	Single-cell full-length transcriptome analysis	SMART-Seq Single Cell PLUS Kit
		SMART-Seq Single Cell Kit
		SMART-Seq v4 PLUS Kit
		SMART-Seq v4 Ultra Low Input RNA Kit
		SMART-Seq HT PLUS Kit
		SMART-Seq HT
	Strand-specific total RNA-seq for mammalian samples (with UMIs)	SMARTer® Stranded Total RNA-Seq Kit v3 - Pico Input Mammalian
	Strand-specific total RNA-seq for mammalian samples (no UMIs)	SMART-Seq Stranded Kit
		SMARTer Stranded Total RNA-Seq Kit v2 - Pico Input Mammalian

*Only the 5′ DE data generated by this kit is analyzed by CogentAP. For analysis of ICELL8 TCR profiling data, we offer the ICELL8 scTCR Analyzer software.

The program takes software-modified input files from sequencers and outputs:

- an HTML report, with results typical to single-cell analysis,
- an R data object (rda file) which can be used as input for **Cogent NGS Discovery Software** (CogentDS), bioinformatics visualization software also available from Takara Bio,
- and raw data files, such as a gene matrix and stats files, which can be used for further analysis.

CogentAP is written in Python and can be run on a Linux server either via a graphical (GUI) or command-line interface.

B.   What's New

1. Cogent NGS Analysis Pipeline v.1.5.1

- New plate-based kit data input options:
  - SMART-Seq Single Cell PLUS Kit & SMART-Seq Single Cell Kit
  - SMART-Seq v4 PLUS Kit & SMART-Seq v4 Ultra Low Input RNA Kit
  - SMART-Seq HT PLUS Kit & SMART-Seq HT
  - SMARTer Stranded Total RNA-Seq Kit v2 - Pico Input Mammalian
  - SMART-Seq Stranded Kit
- Adds Nextera®- and TruSeq®-based adapter sequences for trimming
- Updates to the gene fusion analysis option:
  - New default gene fusion analysis filtering level to improve identification of true fusions
  - Adds a new variable option to customize filtering levels for gene fusion analysis (CLI-only)
- Minor bug fixes:
  - Correctly treats a path statement that ends with `'/'`
  - Install config files updates to accommodate the latest libraries

2. Cogent NGS Analysis Pipeline v1.5

- Improves support for full-length transcriptomics, with additional options for transcript, gene fusion, and immune profiling analysis
- New kit data input options: SMART-Seq Pro Application Kit - 2 Chip

II.   Before You Begin

A.   Supported Operating Systems

CogentAP is designed to be installed on a server running Linux. The following versions of Linux have been tested and are supported for use with the software:

- CentOS 6.9 & 6.10
- RedHat 7.5
- Ubuntu 17

B.   Hardware Requirements

For analyzing the output of Illumina® NextSeq® High-Output sequencing data analysis, the following server requirements (or better) are recommended:

- CPU: 24-cores
- RAM: 64 GB
- Free hard drive space: 500 GB

Testing was also done on MiniSeq™, MiSeq®, HiSeq®, and NovaSeq™ datasets.

- MiniSeq or MiSeq—less computational power may be needed than the specifications described for NextSeq output
- HiSeq or NovaSeq—requires more computational power than described for NextSeq output

Precise hardware requirements were not determined for output from these datasets. Support for performance issues of the servers in conjunction with these dataset types may be limited.

C.   User Account Requirements

CogentAP can be installed in two different access scenarios; the user account requirement depends on which scenario you wish to implement in your environment.

- For use by a single user (single Linux username/account), CogentAP can be installed and run by any account type with install and executable privileges (regular or root access).

- For use by multiple users (accounts), CogentAP can either be installed singly across the multiple accounts (regular or root access) or installed for system-wide access by an administrator with root access.

D.   Additional Hardware and Software Dependencies and Recommendations

- **Bash UNIX shell**

- **Internet connectivity on the server**

  The installation process requires Internet connectivity, as it sources scripts from GitHub, Bioconda, and CRAN and downloads genome information from a Takara Bio FTP server. Please ensure that internet connectivity is available on the UNIX server while installing.

- **Conda**

  CogentAP leverages the open-source package manager Conda for installation of the pipeline and its dependencies. Any tools and applications required by the pipeline are installed through Conda inside a local environment created specifically for CogentAP.

  If Conda is not currently installed on the server, instructions to do so can be found at https://conda.io/miniconda.html. We recommend installation of the lightweight version for Python 3.7+ (typically the 64-bit bash installer), also called Miniconda3 (version 4.10.2 or later).

- **bcl2fastq**

  CogentAP takes as input raw-fastq files, which are converted from the sequencer output FASTQ files using the Illumina software bcl2fastq.

  If you're not sure where to find bcl2fastq in your environment, it can be downloaded from https://support.illumina.com/sequencing/sequencing_software/bcl2fastq-conversion-software.html.

- **Keyboard, monitor, and mouse directly into the server, or a remote access program**

  CogentAP must be run on the Linux server in which it is installed. If users do not have direct console access, a remote access program that enables a Virtual Network Computing (VNC) connection is required through a program such as RealVNC (realvnc.com), TightVNC (tightvnc.com), TigerVNC (tigervnc.org), or similar.

  For more information on VNC, along with other VNC clients that can be used, please see the Wikipedia entry at https://en.wikipedia.org/wiki/Virtual_Network_Computing.

E.   Required Input Files

- Paired-end read FASTQ files

- Sample description file, which can be any one of the following:

  o Well-list file—a text file output by the ICELL8 CellSelect® or ICELL8 cx CellSelect software that contains well-level sample information. For more information, see the ICELL8 cx CellSelect v2.5 Software User Manual, Section III.D.

  o Well-list-like format file—a text file that contains sample information, including columns "Barcode" and "Sample". Each column name is case sensitive. The "Barcode" column contains i7 and i5 indexes concatenated with a plus-sign ("+") (e.g., TAGCGAGT+CCGTTGCG). For more information about the contents of a well-list-like file, please refer to the ICELL8 cx Single-Cell System User Manual, Appendix B, Section D, "<chipID>_WellList.txt".

  o An Illumina sample sheet—a file format used by Illumina for storing biological sample information and metadata associated with a given experiment.

III.   Software Overview
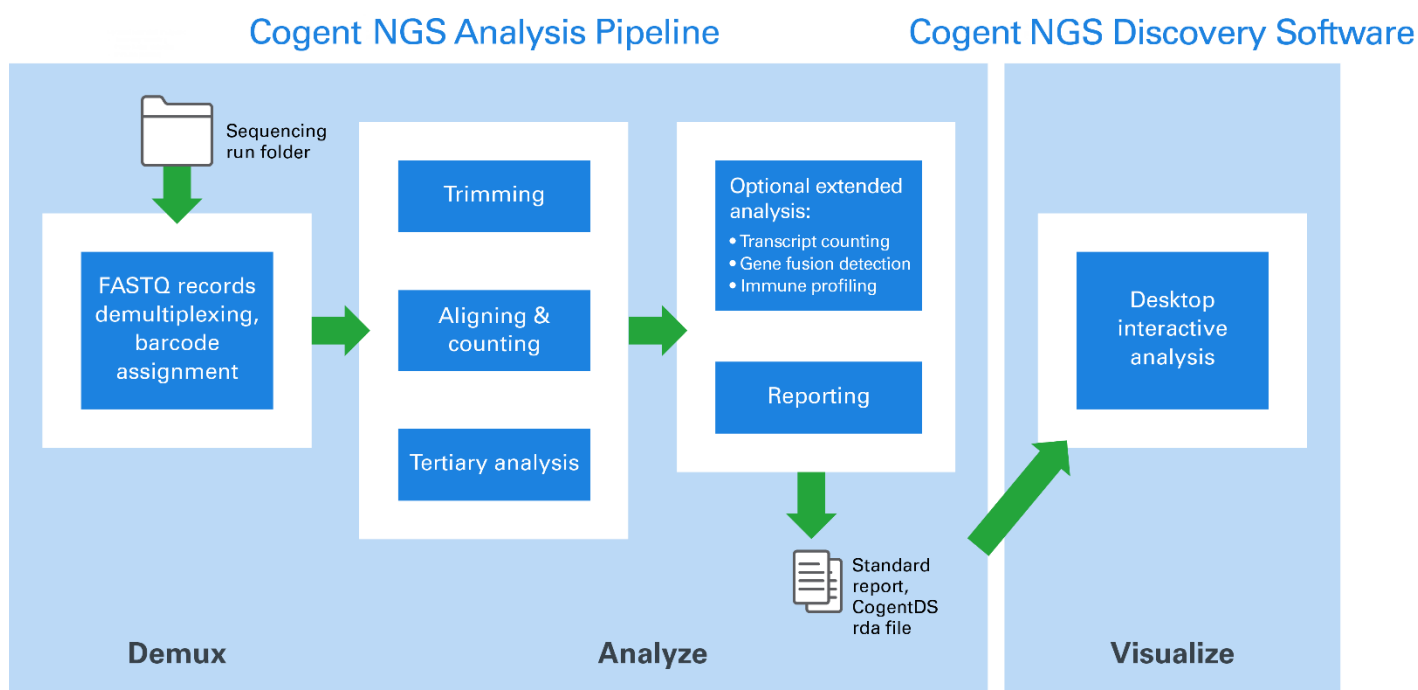
Figure 1. High-level analysis workflow of CogentAP and how its output can be carried over to CogentDS.

CogentAP consists of two main parts, the demultiplexer (demuxer) and the analyzer. The analyzer that can be invoked using a graphical user interface (GUI), called the CogentAP launcher, or can be run on the command line.

- The demultiplexer extracts the barcode from the sequencing data (based on the protocol) and writes it into FASTQ files at the end of the read name. There are two options:

  o The default splits the data up into barcode-level gzip FASTQ files, required for input into the analyzer.

  o The second option leaves the barcode-assigned reads in combined FASTQ files, a format compatible with the previous version of CogentAP (version 1.0).

- The analyzer takes the data sent to it by the demultiplexer and performs the following functions:
  - Read trimming (using Cutadapt)
  - Genome alignment (using the STAR aligner)
  - Calculating Unique Start Stop (USS) positions (only for UMI-enabled kits, using SAMtools and bedtools)
  - Gene expression counting (using Subread)
  - Transcript expression counting (using RSEM, optional)
  - Gene fusion detection (using STAR-Fusion, optional)
  - Immune profiling analysis (using TRUST4, optional)
  - Summarization (using custom scripts)
  - Generating an HTML report (using CogentDS)
- The optional extended analyses (transcript counting, gene fusion, and immune profiling analysis) can be launched independently, taking input directly from the analyzer output directory.

NOTE: Not all experiment types support the optional analyses. See Table 3 in Section V, "Running the Pipeline", to find what analysis options are supported by your experiment type.

IV. Installation & Configuration Options

Cogent NGS Analysis Pipeline v1.5.1 can be obtained through sign-up on our website at takarabio.com/ICELL8-software. After signing up, an email will be sent to you with information on downloading the install script and a unique authorization code (authcode) required to run the installation on your server.

Once obtained, run through the steps in this section to set up the Linux server and install the software.

A. Verify the Conda Installation

To verify that Conda is installed properly on the server.

1. Type the following command in at the prompt in any directory location on the server.

   `conda -V`

   If Conda is successfully installed, it should return text with the version number.

   e.g.,

   `conda 4.10.3`

2. Check to see if the base Conda environment can be activated. Type the following command into the command-line prompt on the Linux server:

   `conda activate`

   A successful Conda install will result in a change in the prompt, as shown in Figure 2.

```
$ conda activate
(base) $
(base) $ conda dectivate
$
```

Figure 2. Screenshot of the Linux command line showing a successful check of the base Conda environment. If the Conda installation was not completed as required, both commands would return error messages.

3. If Conda is successfully installed and the prompt changed as displayed in Figure 2, type the following command to return to the default Linux prompt:

   ```
   conda deactivate
   ```

   This command will take you back to the Linux prompt and out of the Conda environment.

4. Installation of Miniconda3 typically adds the location of its installation to the user's system environment. This is also required for the successful installation of CogentAP.

   The following steps can be used to verify that the Conda $PATH is configured correctly.

   a. Open the file .bash_profile, which for an individual user account will be located in the home directory:

   ```
   more ~/.bash_profile
   ```

   b. Verify a line similar to the following is showing in the file:

   ```
   export PATH="/home/<USERNAME>/miniconda3/bin:$PATH"
   ```

   where <USERNAME> is replaced by the username of the account that installed Conda.

   e.g., username is 'myacct':

   ```
   export PATH="/home/myacct/miniconda3/bin:$PATH"
   ```

   If the line isn't displaying or the .bash_profile file does not exist, it will need to be manually created and populated. For more information on setting an environment variable, see a UNIX user manual or a forum post like https://stackoverflow.com/a/7502128.

B.   Install Cogent NGS Analysis Pipeline v1.5.1

1. Download the installation script (CogentAP_v1.5.1_Linux64_installer.sh), following the directions (a) on the page seen after submitting the sign-up form on the CogentAP product page or (b) in the email sent to the email address submitted in the form.

2. Move or copy the installation script onto the Linux server into the directory location where you want to install CogentAP.

   NOTE: The account logged into while doing the installation must have read/write privileges to the install directory chosen.

3. From the same directory location in Step 2, run the following command:

   ```
   bash CogentAP_v1.5.1_Linux64_installer.sh <AUTHCODE>
   ```

   <AUTHCODE> will be replaced by the unique authentication code included in the sign-up confirmation email.

   The installation process at this point will take anywhere from 10 minutes to 1 hour to complete, depending on the computational capacity of the server and the download speed of the internet connection.

NOTES

- No further user input is required until the install is complete. The installation procedure may be left to run overnight without supervision. If desired, the user can work in another terminal window simultaneous to the install.

- During the installation process, an ssh connection to an SFTP server is created. The host key to the SFTP server will be automatically added to your `~/.ssh/known_hosts` file.

Once the installation is complete, the following message will display.

```
Successfully installed CogentAP pipeline and dependencies.
Please setup genomes next.
```

Figure 3. Text to console illustrating a successful CogentAP software install on the Linux server.

4. Change into the `CogentAP` folder created during the installation process:

`cd CogentAP`

The `CogentAP` directory contains files and directories required by the pipeline's scripts.
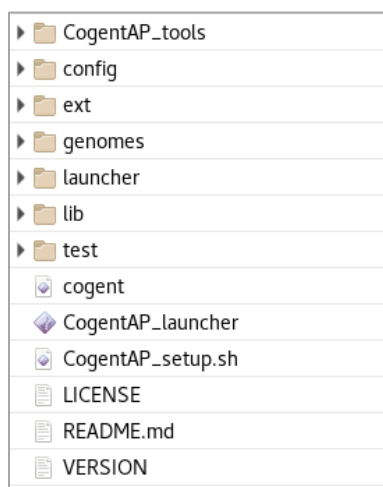
Figure 4. The sub-directory and files list of the `CogentAP` folder post-install.

5. Run the following command to install the human genome build:

`bash CogentAP_setup.sh genome_install hg38`

(Optional): If you will be analyzing sequence data for *Mus musculus* (mice), run the following command to install the house mouse genome build:

`bash CogentAP_setup.sh genome_install mm10`

NOTE: Each genome installation process will take approximately 1 hour to complete, depending on the computational capacity of the server and the download speed of the internet connection.

After a genome build is installed, CogentAP is ready to use.

C.   (Optional) Set Up `$COGENT_AP_HOME` Environmental Variable

For ease of use, we recommend that the CogentAP install directory location be added to the `.bash_profile` as a permanent environmental variable.

Example:

If your account name is 'myacct', the absolute pathname for myacct's home directory is `/home/myacct`, and CogentAP was installed in the `~/bin` directory, edit `.bash_profile` to add the following line:

```
export COGENT_AP_HOME=/home/myacct/bin/CogentAP
```

Once added to the profile, you will either need to log out and back into the account, or load the file in with the command:

```
source ~/.bash_profile
```

The phrase `$COGENT_AP_HOME` can then be used as an alias shortcut to reference `/home/myacct/bin/CogentAP`.

Example:

Running the following while logged in as 'myacct' will change directory to `~/bin/CogentAP`:

```
cd $COGENT_AP_HOME
```

NOTE: Subsequent references to `$COGENT_AP_HOME` in this document refer to the full path where the CogentAP software is installed.

D.   How to Update CogentAP after Installation

To update CogentAP, run the following two commands in sequence:

```
cd $COGENT_AP_HOME
```

```
sh CogentAP_setup.sh update
```

E.   How to Uninstall CogentAP

CogentAP can be uninstalled by deleting the `CogentAP/` software directory from the server.

If `$COGENT_AP_HOME` has been defined in `.bash_profile`, edit the file to remove the reference to `$COGENT_AP_HOME` as well.

NOTE: If you used an older version of this software, called mappa™ Analysis Pipeline, please delete the entire `mappa/` directory to uninstall it.

V.   Running the Pipeline

Before running an analysis, raw-fastq files need to be generated from the sequencer-output FASTQ files. Once those are created, CogentAP can be run in either of two ways:

1. Using a graphical user interface (GUI) (Section V.B)
2. On the command line (CLI) (Section V.C)

Table 2 maps the experiment type and corresponding kits (from Table 1) to the protocol option names/abbreviations used to select them in the GUI and CLI, respectively (Sections V.B and V.C).

Table 2. GUI and CLI protocol selection options for the validated Takara Bio reagent kits.

Experiment type and supported kits	GUI protocol option	CLI protocol option
ICELL8 single-cell full-length transcriptome analysis • SMART-Seq Pro Application Kit - 2 Chip • SMART-Seq ICELL8 cx Application Kit • SMART-Seq ICELL8 Application Kit	ICELL8 full-length transcriptome analysis	`ICELL8_FLA`
Plate-based full-length transcriptome analysis • SMART-Seq Single Cell PLUS Kit • SMART-Seq Single Cell Kit • SMART-Seq v4 PLUS Kit • SMART-Seq v4 Ultra Low Input RNA Kit • SMART-Seq HT PLUS Kit • SMART-Seq HT	Plate-based full-length transcriptome analysis	`SMARTSeq_FLA`
ICELL8 single-cell differential expression analysis • ICELL8 cx 3′ DE • ICELL8 3′ DE	ICELL8 3′DE analysis (no UMI)	`ICELL8_3DE`
ICELL8 single-cell differential expression analysis with UMIs • ICELL8 3′ DE for UMI Reagent Kit (430-000236)	ICELL8 3′DE analysis (with UMI)	`ICELL8_3DE_UMI`
ICELL8 single-cell human TCR 5′ DE • ICELL8 cx Human TCR a/b Profiling • ICELL8 Human TCR a/b Profiling	ICELL8 5′DE analysis (human TCR)	`ICELL8_5DE`
Plate-based strand-specific total RNA-seq for mammalian samples (with UMIs) • SMARTer Stranded Total RNA-Seq Kit v3 - Pico Input Mammalian	Stranded RNA-seq analysis (with UMI)	`Stranded_UMI`
Plate-based strand-specific RNA-seq (no UMIs) • SMART-Seq Stranded Kit • SMARTer Stranded Total RNA-Seq Kit v2 - Pico Input Mammalian	Stranded RNA-seq analysis (no UMI)	`Stranded`

Optional extended analysis ([Section V.D](#))

Gene expression counting is fully supported for all experiment types listed in Table 2.

Transcript expression counting, gene fusion detection, and immune profiling analysis are included specifically to leverage our full-length chemistry advantages over other 3′ chemistries; refer to [Section V.D](#) for how to run these optional extended analyses. While we provide these analyses for our other chemistries (summarized in Table 3, below), we do not recommend it for them.

Table 3. Analysis types available in CogentAP for the Takara Bio chemistries supported by the software.

Experiment Type	Gene expression counting	Transcript expression counting	Gene fusion detection	Immune profiling analysis
ICELL8 single-cell full-length transcriptome analysis • SMART-Seq Pro Application Kit - 2 Chip • SMART-Seq ICELL8 cx Application Kit • SMART-Seq ICELL8 Application Kit	✔	✔	✔	✔
Plate-based full-length transcriptome analysis • SMART-Seq Single Cell PLUS Kit • SMART-Seq Single Cell Kit • SMART-Seq v4 PLUS Kit • SMART-Seq v4 Ultra Low Input RNA Kit • SMART-Seq HT PLUS Kit • SMART-Seq HT	✔	✔	✔	✔
ICELL8 Single-cell differential expression analysis • ICELL8 cx 3′ DE • ICELL8 3′ DE	✔	*	No	No
ICELL8 single-cell differential expression analysis with UMIs • ICELL8 3′ DE for UMI Reagent Kit (430-000236)	✔	*	No	No
ICELL8 Human TCR 5′ DE • ICELL8 cx Human TCR a/b Profiling • ICELL8 Human TCR a/b Profiling	✔	*	No	No
Plate-based strand-specific total RNA-seq for mammalian samples (with UMIs) • SMARTer Stranded Total RNA-Seq Kit v3 - Pico Input Mammalian	✔	†‡	†‡	†‡
Plate-based strand-specific RNA-seq (no UMIs) • SMART-Seq Stranded Kit • SMARTer Stranded Total RNA-Seq Kit v2 - Pico Input Mammalian	✔	†	†	†

*For experiments that do not read the full length of cDNA (3'DE/5'DE), the analysis type is not recommended.

†Not recommended for special cases, like FFPE-samples, which are a common use for the stranded kits.

‡The results will be calculated without UMI collapsing, i.e., not UMI-aware

A.   Generate raw-fastq Files

The CogentAP demultiplexer takes two, one-paired raw-fastq files as input (i.e., not split by barcode). The following procedure converts the sequencer FASTQ output files into the format expected by CogentAP.

1. Log in to the server that stores the sequencing run output folder. This server will typically have the program bcl2fastq installed (see Section II.D for more information about bcl2fastq).

2. Change to the directory where you want the raw-fastq files to be created.

3. From the server were CogentAP is installed, copy the `SampleSheet_dummy.csv` file, located in the `$COGENT_AP_HOME/config` folder, into the directory selected in Step 2.

4. Run bcl2fastq with the following syntax:

```
bcl2fastq -R <RUN_FOLDER> \
-o <RUN_ID> \
--no-lane-splitting \
--sample-sheet SampleSheet_dummy.csv <RUN_ID>.stdout \
2> <RUN_ID>.stderr
```

where:
- `<RUN_FOLDER>` is the path to the sequencing run folder and
- `<RUN_ID>` is the ID automatically generated by Illumina sequencer

NOTE: Recent versions of bcl2fastq (2.17 and higher) have a bug where the indexes required for demultiplexing will not be inserted into the raw-fastq if a sample sheet file is not specified in the command syntax. To prevent encountering the issue, we recommend using the `SampleSheet_dummy.csv` option when generating the raw-fastq files from the bcl2fastq command.

5. Retrieve the raw-fastq files from the `<RUN_ID>` folder located in the working directory from Step 2. These are typically named in the syntax `Undetermined_*.fastq.gz`.

NOTE: To reduce downstream processing time, we recommend that the raw-fastq files are moved to a directory on the server where CogentAP is installed.

B.   GUI

The CogentAP GUI program is called `CogentAP_launcher`. Since it is graphical, it should be accessed by connecting to the Linux server using a remote access tool like VNC (see Section II.D, above).

1. Connect to the Linux console where CogentAP is installed, either remotely or via a graphical interface on the server.

2. Once connected, navigate using the file browser functionality to the folder in which CogentAP is installed. In Figure 5, the CogentAP install folder is located on the desktop.
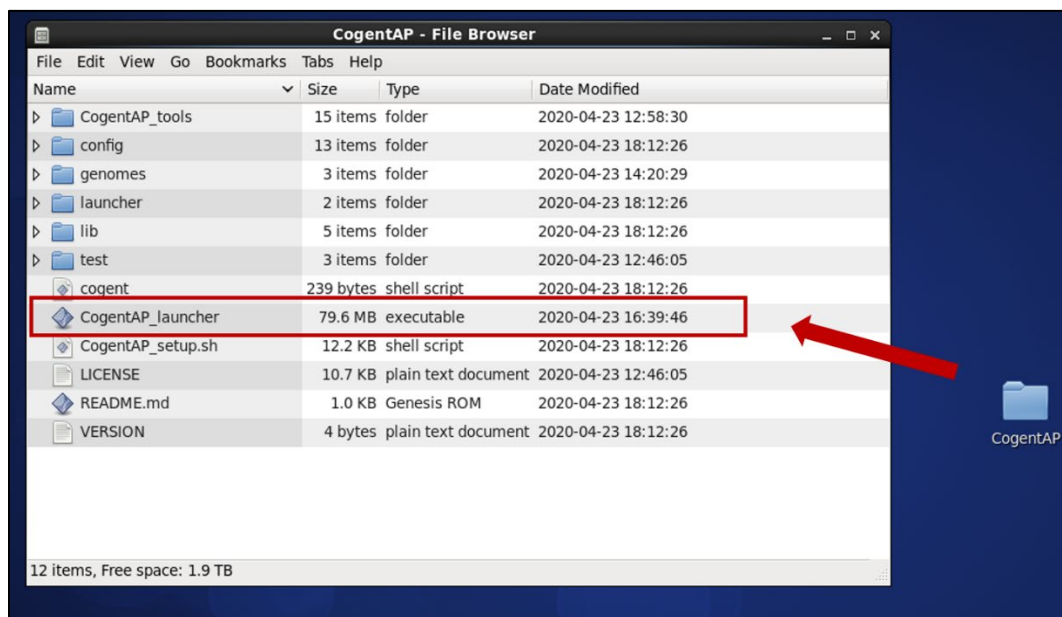
Figure 5. Where to find the `CogentAP_launcher`. The file browser is open to highlight the location of the executable file within the `CogentAP` install directory, found on the desktop.

3. In the install folder, locate the `CogentAP_launcher` file (Figure 5) and double-click on it to run. This will open the CogentAP GUI (Figure 6, left).
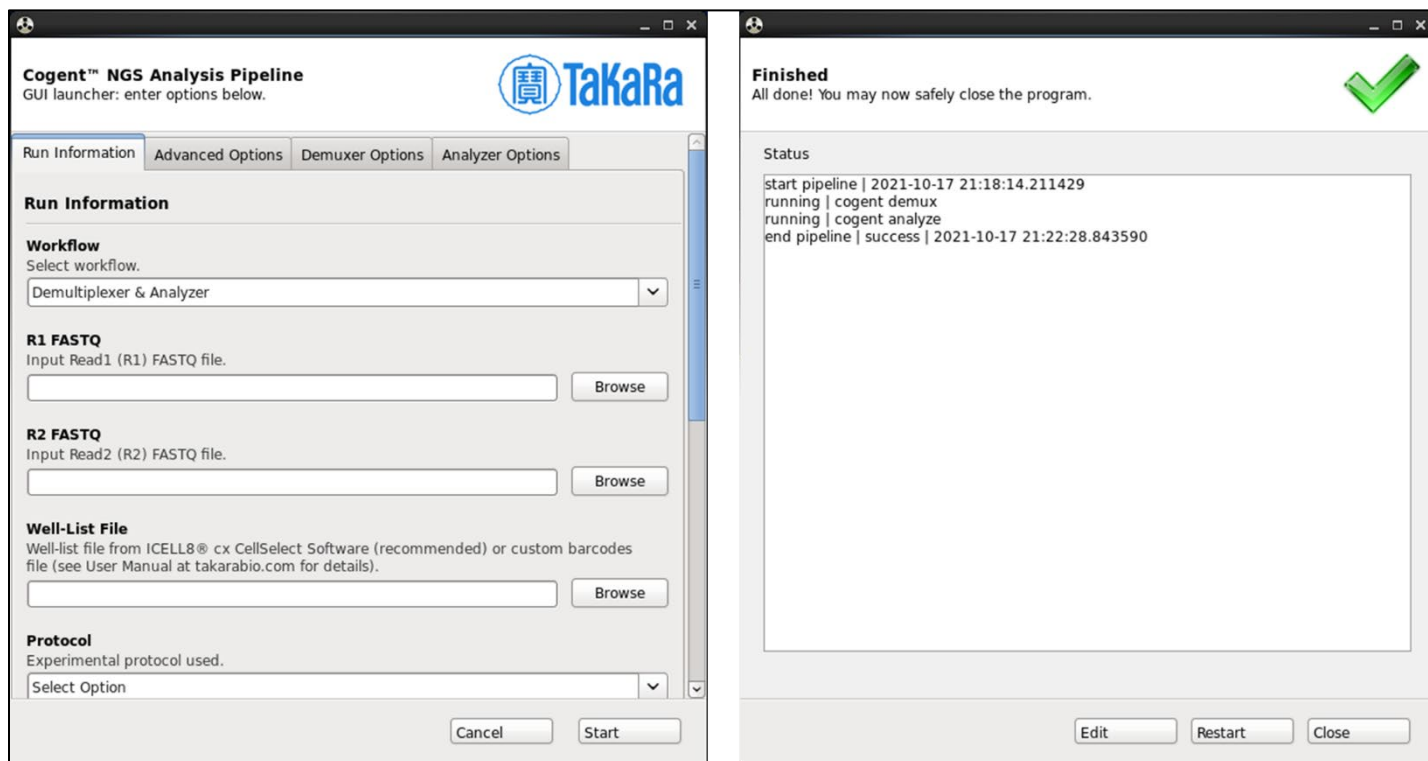
Figure 6. High-level view of the CogentAP GUI. (Left) Launcher interface. **(Right)** Progress window with status messages after the completion of an analysis run.

The launcher has four tabs: *Run Information*, *Advanced Options, Demuxer Options*, and *Analyzer Options*.

a. The *Run Information* tab takes as input:
- The workflow (default being 'Demultiplexer & Analyzer')
- R1 and R2 raw-fastq files (generated in Section V.A)
- The well-list file (see Section II.E for more information about this file)
- The protocol used in the experiment (e.g., ICELL8 full-length transcriptome analysis)

Protocol
Experimental protocol used.

Select Option	⌄

Select Option
ICELL8 full-length transcriptome analysis
Plate-based full-length transcriptome analysis
ICELL8 3′DE analysis (no UMI)
ICELL8 3′DE analysis (with UMI)
ICELL8 5′DE analysis (human TCR)
Stranded RNA-seq analysis (with UMI)
Stranded RNA-seq analysis (no UMI)
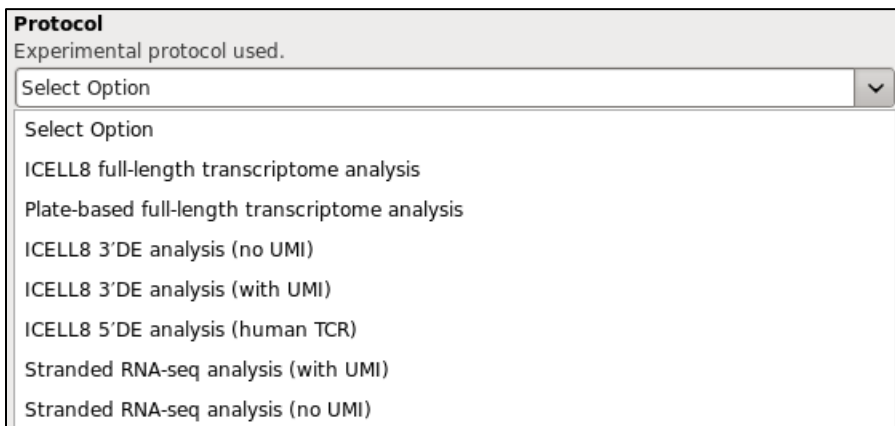
Figure 7. "Protocol" drop-down menu options of the GUI. Refer to Table 2, above, for how to correlate these options with a specific reagent kit.

- The genome to use for alignment (default: hg38)
- The directory location in which to output the results
- The name for a new analysis folder, which will be created in the output directory location specified above and will contain the analysis output files

NOTE: Use the scrollbar on the right of the window to view all the fields in the initial interface.

The other three tabs are intended for advanced users only, but their purposes are listed below:

b. *Advanced Options* can be used to modify parameters such as: the number of processor cores available for analysis, the number of mismatches to consider during barcode assignment, etc.

c. *Demuxer Options* can be used to modify parameters on the demultiplexer process.

d. *Analyzer Options* tab can be used to modify parameters of the analyzer's process such as: skip trimming, extended analysis (Section V.D), etc.

4. Once all the fields have been appropriately filled, click on [Start] to launch the pipeline. The process typically takes a few hours to run.

5. Upon successful completion, the UI displays a "Finished" message as shown in the right panel of Figure 6 (above).

C.   Command-Line Interface

CogentAP starts from the main script, `cogent`, and has defined subcommands, listed below.

Demux and Analyze

- `demux`
- `analyze`

Optional Extended Analysis

- `run_transcript_analysis`
- `run_fusion_analysis`
- `run_immune_analysis`

Additional Commands

- `merge_bam`
- `add_genome`

These scripts can be launched from any location (working directory) on the Linux server where the CogentAP software is installed. The full list of arguments can be accessed using the syntax:

```
cogent <COMMAND> -h
```

The `demux` and `analyze` commands are the core functionality of CogentAP and are described below. Section V.D covers the extended analysis options, while Section V.E describes the additional available commands.

Demux and Analyze

In general, the demuxer (`cogent demux`) is run first to generate demultiplexed FASTQ files. The resulting directory of FASTQ files is then used as input to run the analyzer (`cogent analyze`) to obtain the output files described in Section VII.

See the full list of 'demux' and 'analyze' arguments in Figures 8 and 9 (continued on the next two pages).

`%COGENT_AP_HOME%/cogent demux -h`

```
usage: cogent demux [-h] -i R1_FILE -b BCS_FILE -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded} -o OUT_DIR
                    [-p R2_FILE] [--no_split_fastqs] [-m {0,1}] [-w BACKGROUND_BARCODES_FILE] [-u UMI_LEN] [-n N_PROCESSES] [--n_writers N_WRITERS]
                    [--no_gz] [--undetermined_fq] [--i7_rc {Auto,True,False}] [--i5_rc {Auto,True,False}] [--read_buffer READ_BUFFER] [--hpc] [--prog PROG]

descripton:
  Script to de-multiplex barcoded reads from sequence data stored in FASTQ files. User options are
  designed to simplify de-multiplexing for experiments derived from Takara protocols. Barcode
  (and optionally UMI) sequences are extracted and stored in the read name. Users may specify whether the
  resulting de-multiplexed data are merged, or split into individual barcode-level files.

required arguments:
  -i R1_FILE, --input_fastq R1_FILE
                        Input Read1 (R1) FASTQ file.
  -b BCS_FILE, --barcodes_file BCS_FILE
                        Well List file from Takara's CellSelect Software (Recommended), or custom Barcodes File (see User Manual @
                        https://takarabiousa.github.io for details).
  -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}, --type_of_experiment {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_
FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}
                        Experimental protocol used.
  -o OUT_DIR, --output_dir OUT_DIR
                        Name of output directory to store results.
  -p R2_FILE, --paired_fastq R2_FILE
                        Input Read2 (R2) FASTQ file.

optional arguments:
  -h, --help            show this help message and exit
  --no_split_fastqs     Output merged FASTQ file(s). Barcodes are written into read names and merged into large FASTQ file. By default output into barcode-
                        level FASTQ files.
  -m {0,1}, --mismatch {0,1}
                        Number of allowed mismatched bases per barcode [Default: 1].
  -w BACKGROUND_BARCODES_FILE, --all_well_barcodes_file BACKGROUND_BARCODES_FILE
                        Barcodes file used to calculate background contamination from unselected barcodes. This file is built-in for standard protocols
                        (defined by option `-t`), but can be customized for rare cases where a custom chip or barcode set is used.
  -u UMI_LEN, --umi_length UMI_LEN
                        Overwrite UMI length. Default length is automatically detected by experiment type. [Default: None].
  -n N_PROCESSES, --n_processes N_PROCESSES
                        Number of demultiplexing (worker) processes to spawn during execution [Default: 16].
  --n_writers N_WRITERS
                        Number of writing processes to spawn during execution [Default: 1].
  --no_gz               Do not compress (gzip) output FASTQ files.
  --undetermined_fq     Save Undetermined/Unselected/Short reads to Undetermined FASTQ files.
  --i7_rc {Auto,True,False}
                        Reverse-complement I7 Index (Full Length protocol only). Enter "Auto" to detect and auto-correct the reverse complementation of
                        I5/I7 indices by certain Illumina sequencers. Otherwise manually override with "True" or "False" [Default: "Auto"].
  --i5_rc {Auto,True,False}
                        See help section for `--i7_rc`.
  --read_buffer READ_BUFFER
                        Buffer size of data sent to each demultiplexing (worker) process in GB [Default: 0.01].
  --hpc                 Work as high performance computing mode. To use this option, changing file descriptors (ulimit -n) as unlimited is recommended.
  --prog PROG           Number of reads to process before updating status in log file [Default 10,000,000].
```

Figure 8. The output of `cogent demux -h` at the command line.

`%COGENT_AP_HOME%/cogent analyze -h`

```
usage: cogent analyze [-h] -g {hg38,mm10} -o OUT_DIR -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded} -i INPUT
                      [--threads THREADS_NUM] [--cores CORES_NUM] [--skip_trimming] [--transcript] [--immune] [--fusion] [--debug {none,moderate,high}]

description:
  Script to perform counting analysis for exons and genes by fastq input data.
  The input to this script demux are files output by Cogent demux.
  The fastq files are expected to contain the barcode info in the read name.
  Optionally it can also contain UMI info following the BC.
  The modules currently included are:
    - Trimming (cutadapt)
    - Alignment (STAR)
    - Counting (featureCounts)
    - Summarization (TBUSA)
    - Reporting (TBUSA, CogentDS)

required arguments:
  -g {hg38,mm10}, --genome {hg38,mm10}
                      Choose from pre-built "hg38" or the name of custom genome that you created
  -o OUT_DIR, --output_dir OUT_DIR
                      An output directory to be created to store results
  -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}, --type_of_experiment {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_
FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}
                      Experimental protocol used
  -i INPUT, --input INPUT
                      Directory contains results from demux command. The directory must contain FASTQ files after demultiplexing.

optional arguments:
  -h, --help          show this help message and exit
  --threads THREADS_NUM
                      Number of processes to spawn during execution [Default: 16]
  --cores CORES_NUM   Number of cores used for each step. [Default: 1]
  --skip_trimming     Using this argument will skip trim reads for adapters [Default: FALSE]
  --transcript        Generate transcript expression matrix [Default: FALSE]
  --immune            Generate immune profiling matrix [Default: FALSE]
  --fusion            Generate gene fusion matrix [Default: FALSE]
  --debug {none,moderate,high}
                      Specify debug mode. [Default: moderate]
```

Figure 9. The output of `cogent analyze -h` at the command line.

D.   Optional Extended Analysis

1. Transcript Analysis

a) *GUI*

In the GUI, the transcript analysis option can be selected in the *Analyzer Options* tab.
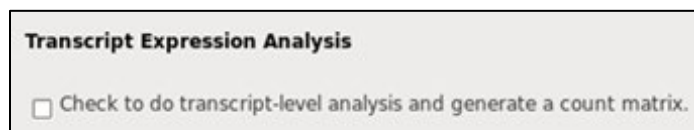
Figure 10. The check box to enable transcript analysis in the GUI *Analyzer Options* tab.

b) *CLI*

Transcript level expression analysis is launched by the command

`cogent run_transcript_analysis`

You can also launch this as an option while running the analyzer with the option `--transcript`

`cogent analyze --transcript`

to launch transcript analysis at the same time. See additional options for `run_transcript_analysis` in Figure 11, below.

The resulting `CogentDS.analysis.rda` file includes both gene-level expression and transcript-level expression.

`%COGENT_AP_HOME%/cogent run_transcript_analysis -h`

```
usage: cogent run_transcript_analysis [-h] -g {hg38,mm10} -o OUT_DIR -t
                                      {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded} -i INPUT_DIR
                                      [--threads THREADS_NUM] [--cores CORES_NUM] [--debug {none,moderate,high}]

description:
  A command to perform counting analysis for transcripts.
  The input to this command is result directory from analyze command.
  The directory is expected to contain BAM files (.toTranscriptome.out.bam) and stats.csv
  This transcript analysis ignores UMI even if UMI enabled experiment type is specified.

required arguments:
  -g {hg38,mm10}, --genome {hg38,mm10}
                        Choose from pre-built "hg38" or the name of custom genome that you created
  -o OUT_DIR, --output_dir OUT_DIR
                        An output directory to be created to store results
  -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}, --type_of_experiment {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_
FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}
                        Experimental protocol used
  -i INPUT_DIR, --input_dir INPUT_DIR
                        Directory contains results from analyze command. The directory must contain BAM files (*.Aligned.toTranscriptome.out.bam) and
                        *_stats.csv.

optional arguments:
  -h, --help            show this help message and exit
  --threads THREADS_NUM
                        Number of processes to spawn during execution [Default: 16].
  --cores CORES_NUM     Number of cores used for each step. [Default: 1]
  --debug {none,moderate,high}
                        Specify debug mode. [Default: moderate]
```

Figure 11. The output of `cogent run_transcript_analysis -h` at the command line.

2. Gene Fusion Analysis

a) GUI

In the GUI, the gene fusion analysis option can be selected in the *Analyzer Options* tab.
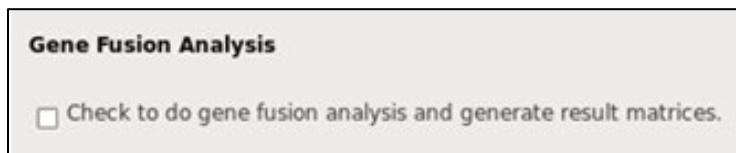
Figure 12. The check box to enable gene fusion analysis in the GUI *Analyzer Options* tab.

b) CLI

Gene fusion analysis is launched by the command

`cogent run_fusion_analysis`

You can also launch this as an option while running the analyzer with the option `--fusion`

`cogent analyze --fusion`

to launch gene fusion analysis at the same time.

The sensitivity of the fusion analysis command can be controlled by the optional argument, `-f`. The table below lists the possible input values (0–4), the description of each filtering level, how it corresponds to the original options of STAR-Fusion, and its relative stringency. Option value '1' is used as the default by the `--fusion` option.

For additional options for `run_fusion_analysis`, see Figure 13.

Table 4. Option values of `-f` in `run_fusion_analysis` command.

`-f` value	Description	Comment	STAR-Fusion parameters[*]
0	Strict	Most stringent—will possibly filter out true fusions	(default)
1	**Max sensitivity[†]**	**CogentAP default (recommended[‡])**	**`--max_sensitivity`**
2	Max sensitivity + no exclusion of reverse transcriptase artifacts	Will possibly include false positives. Use for very hard-to-detect fusions.	`--max_sensitivity` `--no_RT_artifact_filter`
3	Full monty[†]	Will greatly increase false positive rate[§]. Use external software for custom filtering instead (e.g., MS-Excel, R).	`--full_Monty`
4	No filters	Most lenient	`--no_filter` `--skip_FFPM` `--no_annotation_filter` `--no_RT_artifact_filter` `--skip_EM`

*The parameters used internally by CogentAP when calling the STAR-Fusion program during analysis. This information is provided for advanced users to assist them with choosing the `-f` value to best fit their experimental goals. For additional information about the parameters, refer to the STAR-Fusion documentation.

†Term used by STAR-Fusion developer to refer to a suite of parameters called by this `-f` value (reference).

‡Recommendation based on results of internal testing.

§Based on information from the developer (reference).

The resulting `CogentDS.analysis.rda` file includes gene fusion detection and all other analysis done with the `analyze` command.

`%COGENT_AP_HOME%/cogent run_fusion_analysis –h`

```
usage: cogent run_fusion_analysis [-h] -g {hg38,mm10} -o OUT_DIR -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded} -i
                                  INPUT_DIR [-f {0,1,2,3,4}] [--threads THREADS_NUM] [--cores CORES_NUM] [--debug {none,moderate,high}]

description:
  A command to perform gene fusion detection analysis.
  The input to this command is result directory from analyze command.
  The directory is expected to contain junction information files (.Chimeric.out.junction) and stats.csv
  This analysis ignores UMI even if UMI enabled experiment type is specified.

required arguments:
  -g {hg38,mm10}, --genome {hg38,mm10}
                        Choose from pre-built "hg38" or "mm10". Other custom genome builds are not supported.
  -o OUT_DIR, --output_dir OUT_DIR
                        An output directory to be created to store results
  -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}, --type_of_experiment {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_
FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}
                        Experimental protocol used
  -i INPUT_DIR, --input_dir INPUT_DIR
                        Directory contains results from analyze command. The directory must contain genematrix, *_stats.csv and chimeric files
                        (*.Chimeric.out.junction).

optional arguments:
  -h, --help            show this help message and exit
  -f {0,1,2,3,4}        Specify stringency of filtering from 0 to 4. 0 is most stringent and 4 is most lenient. [Default: 1]
  --threads THREADS_NUM
                        Number of processes to spawn during execution [Default: 16].
  --cores CORES_NUM     Number of cores used for each step. [Default: 1]
  --debug {none,moderate,high}
                        Specify debug mode. [Default: moderate]
```

Figure 13. The output of `cogent run_fusion_analysis -h` at the command line.

3. Immune Profiling Analysis

a) GUI

In the GUI, the immune profiling analysis option can be selected in the *Analyzer Options* tab.

Immune Profiling Analysis

☐ Check to do immune profiling analysis and generate result matrices.

Figure 14. The check box to enable immune profiling analysis in the GUI *Analyzer Options* tab.

b) CLI

Gene fusion analysis is launched by the command

`cogent run_immune_analysis`

You can also launch this as an option while running the analyzer with the option `-- immune`

`cogent analyze --immune`

to launch gene fusion analysis at the same time. See additional options for `run_immune_analysis` in Figure 15, below.

The resulting `CogentDS.analysis.rda` file includes detected clonotypes and all other analysis done with the `analyze` command.

`%COGENT_AP_HOME%/cogent run_immune_analysis -h`

```
usage: cogent run_immune_analysis [-h] -g {hg38,mm10} -o OUT_DIR -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded} -i
                                  INPUT_DIR -a ANALYZE_DIR [--threads THREADS_NUM] [--cores CORES_NUM] [--debug {none,moderate,high}]

description:
  A command for performing immune-profiling on split fastqs.
  The input to this script are files output by Cogent demux.
  The fastq files are expected to contain the barcode info in the read name.
  The modules currently included are:
        -- read assembly & clonotype identification (Trust4)

required arguments:
  -g {hg38,mm10}, --genome {hg38,mm10}
                        Choose from pre-built "hg38" or "mm10". Other custom genome builds are not supported.
  -o OUT_DIR, --output_dir OUT_DIR
                        An output directory to be created to store results
  -t {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}, --type_of_experiment {ICELL8_3DE,ICELL8_3DE_UMI,ICELL8_5DE,ICELL8_
FLA,Stranded_UMI,SMARTSeq_FLA,Stranded}
                        Experimental protocol used
  -i INPUT_DIR, --input_dir INPUT_DIR
                        Directory contains results from demux command. The directory must contain FASTQ files after demultiplexing.
  -a ANALYZE_DIR, --analyze_dir ANALYZE_DIR
                        Directory contains results from analyze command. The directory must contain genematrix and *_stats.csv.

optional arguments:
  -h, --help            show this help message and exit
  --threads THREADS_NUM
                        Number of processes to spawn during execution [Default: 16].
  --cores CORES_NUM     Number of cores used for each step. [Default: 1]
  --debug {none,moderate,high}
                        Specify debug mode. [Default: moderate]
```

Figure 15. The output of `cogent run_immune_analysis -h` at the command line.

E.   Additional Commands

1. Merge BAM Files

NOTE: This function is only available as a command-line option.

Genome-coordinated BAM files (`*.Aligned.out.bam`) generated, per barcode, by the `analyze` command can be merged with the command, below.

`cogent merge_bam`

Any CSV or TSV file containing group names corresponding to barcodes can be used as input.

Examples:

- If you want to study read coverage of a gene of interest on a genome viewer, use this command to merge the alignment results from a group of cells of a certain sub-type.

- By using downloaded metadata from CogentDS, you can merge BAM files per cluster.

See Figure 16, below, for additional options for `merge_bam` and Section VII.F for more details about the output BAM files.

```
%COGENT_AP_HOME%/cogent merge_bam -h
```

```
usage: cogent merge_bam [-h] -f INPUT_FILE -c TARGET_COLUMN -i INPUT_DIR -o OUT_DIR [-n CORES_NUM]

description:
  A command to merge bam files according to custom groups.
  The input to this command is either CSV or TSV contains barcode and any columns.
  This command performs file merging by specified column name in input file.
  BAM files are merged by names described in the column and corresponding barcodes.

required arguments:
  -f INPUT_FILE, --file INPUT_FILE
                         A list including "Barcode" and a target column specified by -c option. Either CSV or TSV is acceptable.
  -c TARGET_COLUMN, --column TARGET_COLUMN
                         Column name used to create groups for merging.
  -i INPUT_DIR, --input_dir INPUT_DIR
                         Directory contains results from analyze command. The directory must contain "bam" directory and BAM files
                         (*.Aligned.out.bam).
  -o OUT_DIR, --output_dir OUT_DIR
                         An output directory to be created to store results

optional arguments:
  -h, --help             show this help message and exit
  -n CORES_NUM, --n_cores CORES_NUM
                         Number of cores used during merging [Default: 8].
```

Figure 16. The output of `cogent merge_bam -h` at the command line.

2. Add a Genome Build

The human and mice genome builds available from our server ([Section V.B](#), "Installation") are recommended for use in the pipeline, but genomes of other species can be added into the software post-install.

NOTE: Extended analysis for gene fusion detection or immune profiling is not supported for custom genome builds added through this process.

To add custom genome data to CogentAP:

1. Download the following two files for the genome of interest:

   - The FASTA file containing all the sequences (chromosomes and contigs)

   - The GTF file containing the annotation, importantly the gene information for analysis

   NOTE: As FASTA and GTF files are a standard file format, files from any source should work with this script. However, this script has only been tested on genomes downloaded from Ensembl.

   If a problem is encountered using files from another source, it is recommended to try the import using the Ensembl files of the genome.

2. Run the script:

   ```
   $COGENT_AP_HOME/cogent add_genome \
   -g <common_species_name> \
   -f <FASTA_FILENAME> \
   -a <GTF_FILENAME>
   ```

   where `<common_species_name>` is the name of the genome being added and the `<FASTA_FILENAME>` and `<GTF_FILENAME>` are the exact path and file names for the FASTA and GTF files unzipped in Step 2, respectively, on the server.

   For additional help with this script, type:

   ```
   $COGENT_AP_HOME/cogent add_genome -h
   ```

Example:

(using the fruit fly genome from Ensembl.org)

1. Download the FASTA file
   (`Drosophila_melanogaster.BDGP6.dna.chromosome.2L.fa`) using FTP:

   ```
   wget ftp://ftp.ensembl.org/pub/release-
   94/fasta/drosophila_melanogaster/dna/Drosophila_melanogaster.BDGP
   6.dna.toplevel.fa.gz
   ```

   or copy and paste the URI:

   ```
   http://ftp.ensembl.org/pub/release-
   94/fasta/drosophila_melanogaster/dna/Drosophila_melanogaster.BDGP
   6.dna.toplevel.fa.gz
   ```

   into the address bar of a web browser and follow the prompts to save the file locally.

2. Download the GTF file (`Drosophila_melanogaster.BDGP6.94.gtf`) using FTP:

   ```
   wget ftp://ftp.ensembl.org/pub/release-
   94/gtf/drosophila_melanogaster/Drosophila_melanogaster.BDGP6.94.g
   tf.gz
   ```

   or copy and paste:

   ```
   http://ftp.ensembl.org/pub/release-
   94/gtf/drosophila_melanogaster/Drosophila_melanogaster.BDGP6.94.g
   tf.gz
   ```

   into the address bar of a web browser and follow the prompts to save the file locally.

3. Unzip the two downloaded files to find the
   `Drosophila_melanogaster.BDGP6.dna.chromosome.2L.fa` and
   `Drosophila_melanogaster.BDGP6.94.gtf` files.

4. If the FASTA and GTF files are stored in the `~/ensembl` directory (for example), run the
   following script, where 'fruitfly' is the `<common_species_name>`:

   ```
   $COGENT_AP_HOME/cogent add_genome \
   -g fruitfly \
   -f ~/ensembl/Drosophila_melanogaster.BDGP6.dna.chromosome.2L.fa \
   -a ~/ensembl/Drosophila_melanogaster.BDGP6.94.gtf
   ```

F. Processing Time

The time taken by the pipeline will vary based on the hardware specifications of the server on which it is run, the size of the raw-fastq input files, and where the files are stored.

During testing, a combined demultiplexing and analysis run for data generated by MiSeq (~25M read pairs) against raw-fastq files stored locally (on the same server CogentAP was installed) typically took about 1–1.5 hours to process. A NextSeq High Output run (~400M read pairs) from local raw-fastq files typically took ~20–25 hours to complete.

Input taken from a NovaSeq run (~2G read pairs, or more) will take even longer. In this type of scenario, using the skip trimming option (in the GUI, under the *Analyzer Options* tab, in the CLI, the cogent

`analyze --skip_trimming` option) will tell the analyzer to skip the trimming phase of its process, which can significantly improve processing time. The results are similar in a typical dataset with or without trimming.

If the raw-fastq files are instead stored on a network drive, these baselines might be exceeded.

VI. Test Dataset

A mini dataset file (referred to here as test dataset) is included in the CogentAP distribution package; it can be found under the main installation folder in a sub-folder called $COGENT_AP_HOME/test/ (Figure 17, below). This dataset can be used to test the running of the pipeline end-to-end and will provide a sample of the output files. The output (report and stats only) from the test dataset is also included in the CogentAP installation and can be found in $COGENT_AP_HOME/test/out_test/. These output files can be used to compare to the output of your test run to verify everything is working correctly.

NOTE: The test dataset should not be used for inference purposes. CogentAP output statistics and plots will only be meaningful with a real dataset.
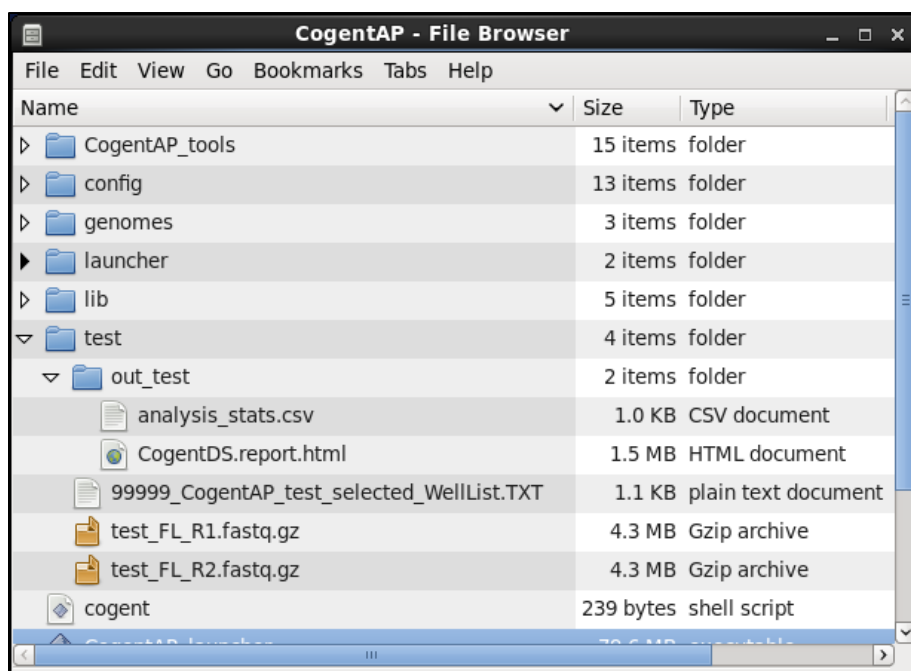

Figure 17. The `test/` folder under `$COGENT_AP_HOME`. The sample `*.fastq.gz` files and example output directory `out_test/` can be found there.

A run using the test data can be started either from the GUI (Section A) or the command line (Section B).

The test run using either method should take ~5–10 min to complete successfully.

## A.   Test Data through CogentAP Launcher UI
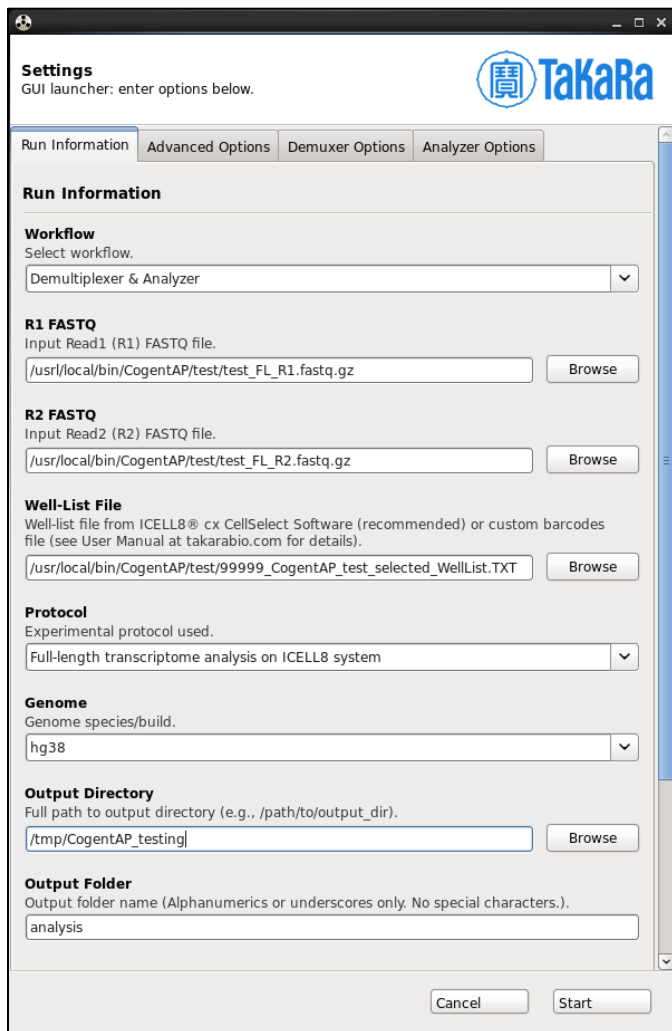
Figure 18. Configuration of the fields in the UI launcher for the test dataset files.

The parameters defined in the CogentAP launcher fields in Figure 18 are listed below. The paths are written as $COGENT_AP_HOME being /usr/local/bin/. Once completed, the output folder (analysis/) will be located under the directory specified as 'Output Directory' (/tmp/CogentAP_testing/).

Table 5. Parameter information for example CogentAP launcher run depicted in Figure 18.

Field name	Input value
Workflow	Demultiplexer & Analyzer
R1 Fastq	/usr/local/bin/CogentAP/test/test_FL_R1.fastq.gz
R2 Fastq	/usr/local/bin/CogentAP/test/test_FL_R2.fastq.gz
Well List File	/usr/local/bin/CogentAP/test/99999_CogentAP_test_selected_WellList.TXT
Protocol	Full Length
Genome	hg38
Output Directory	/tmp/CogentAP_testing/
Output Folder	analysis

B.   Test Data through the Command-line Interface

1. Run the demultiplexer script.

```
$COGENT_AP_HOME/cogent demux \
-i $COGENT_AP_HOME/test/test_FL_R1.fastq.gz \
-p $COGENT_AP_HOME/test/test_FL_R2.fastq.gz \
-b $COGENT_AP_HOME/test/99999_CogentAP_test_selected_WellList.TXT \
-t ICELL8_FLA \
-o Desktop/install_test/demux \
-n 8
```

```
$ Desktop/CogentAP/cogent demux
> -i Desktop/CogentAP/test/test_FL_R1.fastq.gz \
> -p Desktop/CogentAP/test/test_FL_R2.fastq.gz \
> -b Desktop/CogentAP/test/99999_CogentAP_test_selected_WellList.TXT \
> -t ICELL8_FLA \
> -o Desktop/install_test/demux \
> -n 8
```

Figure 19. Example syntax for running the demux script on the provided test data via the command line.

2. Once the demuxer is finished, run the data through the analysis script.

```
$COGENT_AP_HOME/cogent analyze \
-i Desktop/install_test/demux\
-g hg38 \
-o Desktop/install_test/analyze \
-t ICELL8_FLA
--threads 8
```

```
$ Desktop/CogentAP/cogent analyze
> -i Desktop/install_test/demux \
> -g hg38 \
> -o Desktop/install_test/analyze \
> -t ICELL8_FLA \
> --threads 8
```

Figure 20. Example syntax for running the analyzer script on the provided test data via the command line.

Once completed, the output folder (Step 1: `demux/`; Step 2: `analyze/`) will be located under the directory specified as 'Output Directory' (`Desktop/install_test`, specified by the `-o` option parameter).

VII.   Output Files

The pipeline produces output files that serve two purposes:

1. Summarization of results using typical statistics and plots

2. Facilitating further analyses using our interactive R kit, Cogent NGS Discovery Software (CogentDS), or any other tertiary analysis tool

A.   Output Folder Structure

Both the CogentAP GUI launcher and CLI are designated to output to a folder specified in the run parameters, but the file structure within that directory is slightly different between the two methods.

1. GUI

From the GUI, the contents of the output file will resemble Figure 21:
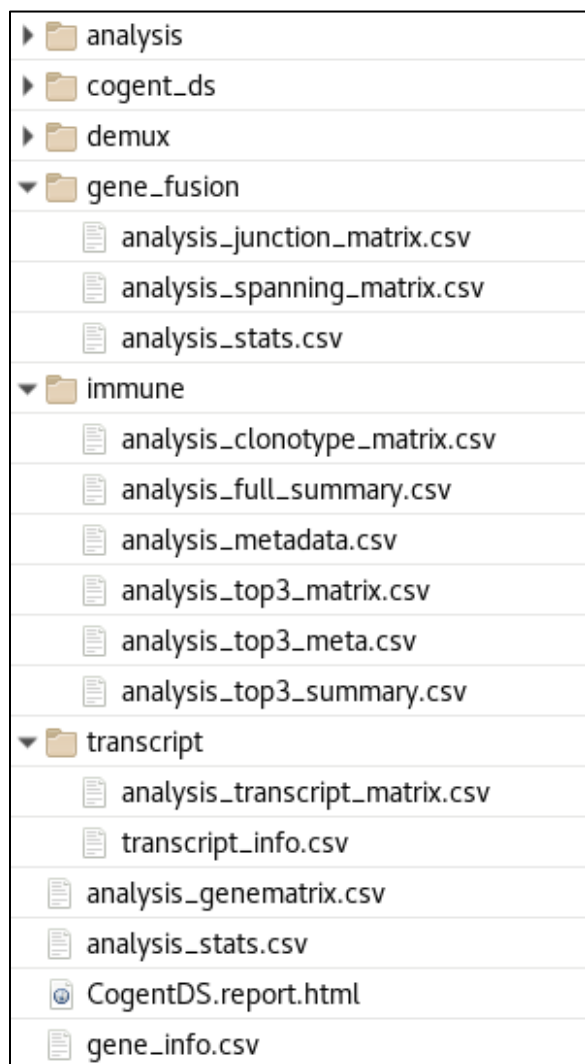
Figure 21. Folders and files of the output directory by launcher UI.

2. CLI

From the command-line interface, the contents of the output file will resemble Figure 22:
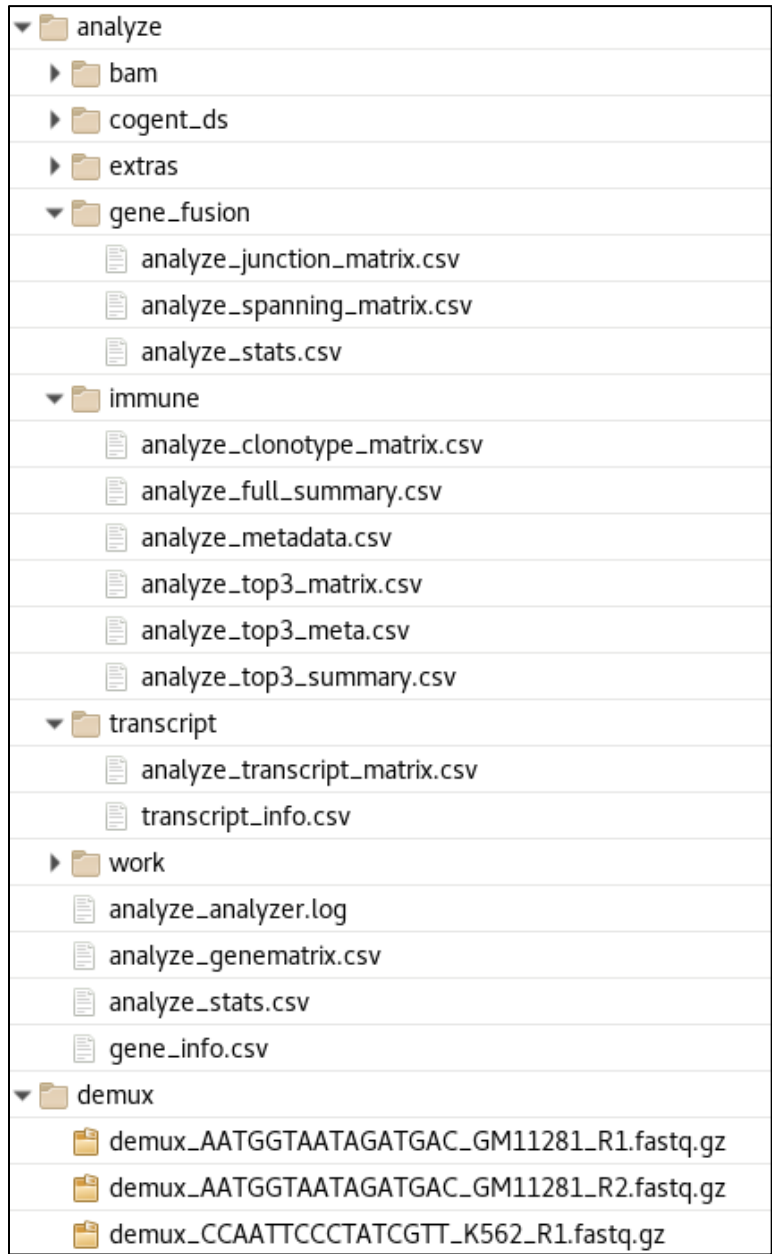
Figure 22. Folders and files of the output directory by command-line interface.

B.   HTML Report

The HTML report is generated by the same report process as CogentDS, using standard parameters, and contains the example statistics and plots listed below. For complete details, please see the Cogent NGS Discovery Software User Manual.

1. Experiment overview

	Read Count	% of Barcoded Reads
Barcoded Reads	390,431,081	100.00
Trimmed Reads	387,754,773	99.31
Unmapped Reads	16,289,719	4.17
Mapped Reads	371,465,054	95.14
Uniquely Mapped Reads	326,977,024	83.75
Multimapped Reads	44,488,030	11.39
Total Exon Reads	310,842,529	79.62
Unique Exon Reads	282,442,377	72.34
Ambiguous Exon Reads	28,400,152	7.27
Total Intron Reads	9,403,003	2.41
Unique Intron Reads	8,300,033	2.13
Ambiguous Intron Reads	1,102,970	0.28
Intergenic Reads	6,731,492	1.72
Additional Information		
Mitochondrial Reads	10,931,191	2.80
Ribosomal Reads	1,910,426	0.49

Figure 23. Example experiment overview section of the HTML report.

2. Correlation analyses

	Min	1st Qu	Median	Mean	3rd Qu	Max
Cancer	0.46	0.71	0.75	0.75	0.79	0.93
Normal	0.40	0.68	0.72	0.71	0.76	0.88
PosCtrl	0.74	0.78	0.80	0.80	0.82	0.84

Note: Correlation Analysis performed using the 2000 genes identified for clustering.
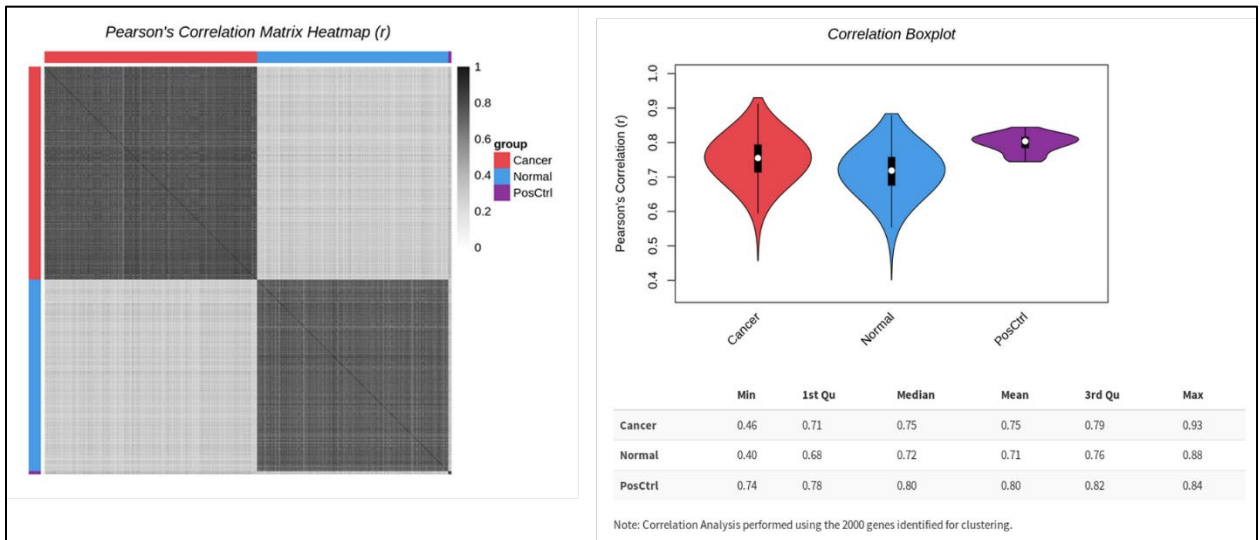
Figure 24. Example correlation analyses section of the HTML report.

3. Reads, Genes, Mito, and Ribo

	Genes per Barcode (Mean)	Genes per Barcode (Median)
Cancer	4,752.80	4,839.00
ln(CPM10000) > 0.1	4,147.58	4,165.00
ln(CPM10000) > 1.0	1,237.64	1,237.00
Normal	4,788.21	4,864.00
ln(CPM10000) > 0.1	4,148.93	4,172.00
ln(CPM10000) > 1.0	1,178.00	1,172.00
PosCtrl	6,085.75	6,177.50
ln(CPM10000) > 0.1	5,274.50	5,340.00
ln(CPM10000) > 1.0	1,496.75	1,520.50
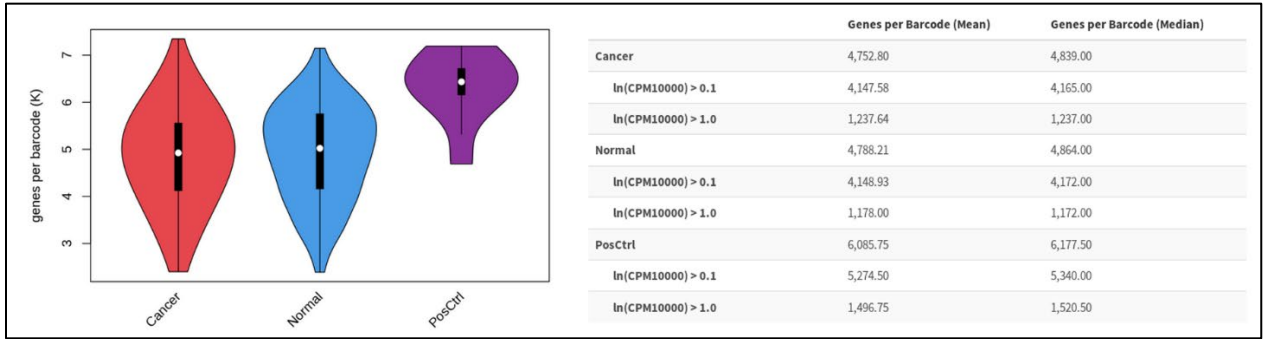
Figure 25. Example gene counts chart from the HTML report.

4. Clustering tables
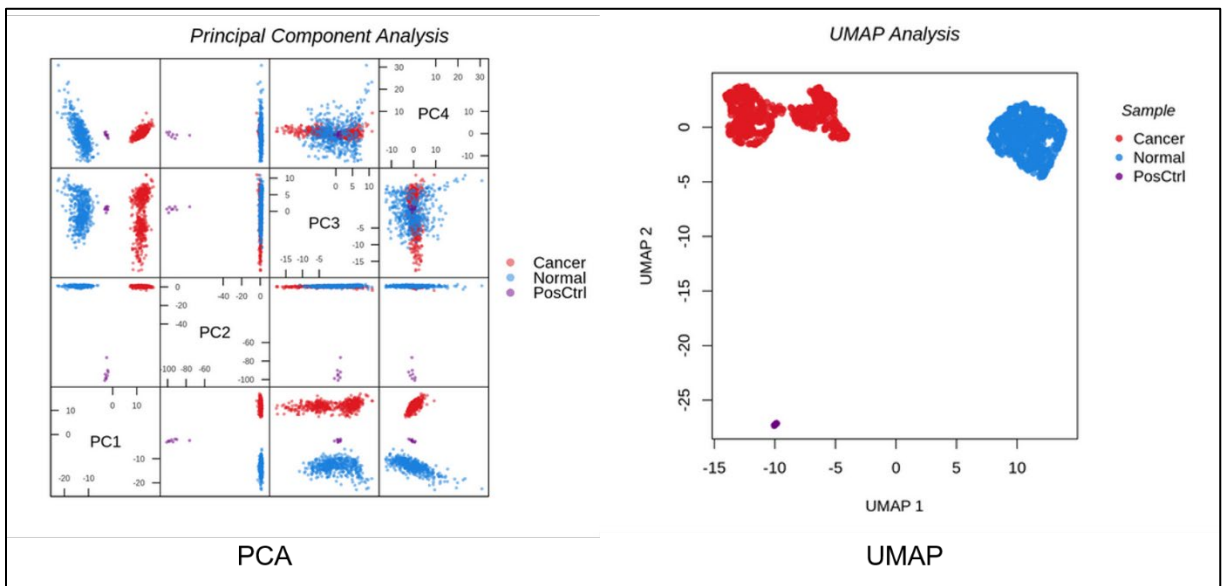
Figure 26. Example clustering tables from the HTML report.

C. `CogentDS.analysis.rda` File

During the generation of the HTML report, an rda file, `CogentDS.analysis.rda`, is created with the results of the various analysis modules. This file can be used directly as input into CogentDS to perform additional analysis, saving processing time in that tool.

D.   Raw Data Files

CogentAP generates several raw-data CSV files, based on the experiment type and types of analysis performed on the demultiplexed data. The tables below list possible raw output files, grouped by interface type (GUI or CLI), as found within the output folder specified during the analysis run (Section VII.A). For more details about the files themselves, please refer to the Appendix.

Table 6. Raw data files generated by CogentAP GUI.

Analysis option	Referred to as	File name	Subfolder
(Default)	Stats	`analysis_stats.csv`	
	Gene matrix	`analysis_genematrix.csv`	
	Gene info	`gene_info.csv`	
Transcript Expression Analysis	Gene transcript info	`transcript_info.csv`	transcript/
	Transcript matrix	`analysis_transcript_matrix.csv`	
Gene Fusion Analysis	Junction matrix	`analysis_junction_matrix.csv`	
	Spanning matrix	`analysis_spanning_matrix.csv`	gene_fusion/
	Fusion_stats	`analysis_stats.csv`	
Immune Profiling Analysis	Clonotype matrix	`analysis_clonotype_matrix.csv`	
	Metadata	`analysis_metadata.csv`	immune/
	Summary	`analysis_full_summary.csv`	
	Top 3 clonotype matrix	`analysis_top3_matrix.csv`	
	Top 3 metadata	`analysis_top3_meta.csv`	
	Top 3 summary	`analysis_top3_summary.csv`	

Table 7. Raw data files generated by CogentAP CLI.

Analysis command	Referred to as	File name	Subfolder
`analyze`	Stats	`analyze_stats.csv`	
	Gene matrix	`analyze_genematrix.csv`	
	Gene info	`gene_info.csv`	
`--transcript`	Gene transcript info	`transcript_info.csv`	transcript/
	Transcript matrix	`analyze_transcript_matrix.csv`	
`--fusion`	Junction matrix	`analyze_junction_matrix.csv`	
	Spanning matrix	`analyze_spanning_matrix.csv`	gene_fusion/
	Fusion_stats	`analyze_stats.csv`	
`--immune`	Clonotype matrix	`analyze_clonotype_matrix.csv`	
	Metadata	`analyze_metadata.csv`	immune/
	Summary	`analyze_full_summary.csv`	
	Top 3 clonotype matrix	`analyze_top3_matrix.csv`	
	Top 3 metadata	`analyze_top3_meta.csv`	
	Top 3 summary	`analyze_top3_summary.csv`	

E.   `Extras` Folder

The `extras/` folder contains similar result files as Sections VII.B–D above, in the same format, but the data is calculated by also including intron regions of genes. Additionally, when used with a UMI enabled kit, results without UMI calculations are stored in this folder. You can also use these files for tertiary analysis.

- From the GUI, `extras/` can be found in `<output folder>/analysis`

- From the CLI, it will be in the folder specified by the value of the `cogent analyze` command output_dir (`-o`) argument

F.   `BAM` Files

The `bam/` folder contains BAM files generated during the alignment step in gene or transcript expression analysis. These files are required for gene expression, transcript expression, and gene fusion analysis.

- `*.Aligned.out.bam:` Output files from genome alignment. These files are used for downstream gene expression analysis.

- `*.Aligned.toTranscriptome.out.bam:` Output files containing transcriptome-based alignment information that are used for transcript expression analysis.

- `*.Chimeric.out.junction:` Output files with chimeric junction information used in gene fusion analysis.

Appendix. Analysis Raw Data Files

NOTE: The raw data output files listed in this appendix are all in CSV format.

A. Default Analysis Files

Table 8. Raw data output files generated by the default CogentAP analysis command.

Interface	Referred to as	File name
GUI	Stats	`analysis_stats.csv`
	Gene matrix	`analysis_genematrix.csv`
	Gene info	`gene.info.csv`
CLI	Stats	`analyze_stats.csv`
	Gene matrix	`analyze_genematrix.csv`
	Gene info	`gene.info.csv`

1. Stats File

The Stats file contains barcode-level statistics across the analysis pipeline. Starting from barcoded reads, it summarizes the number of reads after each step in the pipeline: trimmed reads, mapped reads, exon/intron/intergenic reads, mitochondrial reads, ribosomal reads, etc. It also lists the number of genes detected per barcode.

The columns shown in this file depend on the reagent kit used to generate the input data. As an example, you can see columns related to UMI when you use ICELL8_3DE_UMI (for the ICELL8 3′ DE Kit with UMIs) or Stranded_UMI (SMARTer Stranded Total RNA-Seq Kit v3 - Pico Input Mammalian), but not ICELL8_FLA (SMART-Seq Pro Application Kit - 2 Chip). Figure 27 shows a sample excerpt from this file.

Figure 27. Example Stats file output.

The tables below document all potential columns that might appear in the Stats file. Not all stats files will include every column listed.

Table 9. Columns that will be present in the `*_stats.csv` file output by CogentAP (input workflow agnostic).

Column name	Description
Barcode	Detected barcodes. This value will usually be the sample name from the well-list or well-list-like file, but there are three exceptions, documented in the table below.
Sample	Sample names described in sample description file.
Barcoded_Reads	Number of reads after demultiplexing.
Trimmed_Reads	Number of remained reads after trimming.
Unmapped_Reads	Number of reads not mapped to genome.
Mapped_Reads	Number of reads mapped to genome.
Multimapped_Reads	Number of reads mapped to multiple genomic locations.
Uniquely_Mapped_Reads	Number of reads mapped to one genomic location. These reads are used for counting.
Exon_Reads	Number of reads assigned to an exonic region.
Ambiguous_Exon_Reads	Number of reads assigned to exonic regions of multiple genes.
Intron_Reads	Number of reads assigned to an intronic region.
Ambiguous_Intron_Reads	Number of reads assigned to intronic regions of multiple genes.
Gene_Reads	Number of reads assigned to a gene region (exon + intron).
Intergenic_Reads	Number of reads assigned to an intergenic region.
No_of_Genes	Number of detected genes.
Mitochondrial_Reads	Number of reads assigned to mitochondrial chromosome.
Ribosomal_Reads	Number of reads assigned to a ribosomal gene. Fraction of ribosomal reads should be calculated based on number of Uniquely_Mapped_reads.

The "Barcode" column, in addition to the samples named in the well-list or well-list-like file, will also have three additional rows, which are described in the following table.

Table 10. Additional "Barcode" rows present in the Stats output file.

Barcode field value	Description
Short	Number of reads containing N in barcode or having shorter length than barcode.
Unselected	Number of reads having a barcode included in Chip's description, but not included in sample description file.
Undetermined	Number of reads having undetermined barcode.

The table below lists additional columns that will be present in the stats file when the input FASTQ files result from the ICELL8 3′ DE for UMI Reagent Kit (Cat. No. 640005) workflow on the ICELL8 Single-Cell System (Cat. No. 640000).

Table 11. Additional columns in the stats file for 3′ DE analysis with UMIs on ICELL8 system.

Column name	Description
No_of_UMIs	Number of UMI variations detected after demultiplexing.
Exon_nUMIs	Number of deduplicated reads assigned to an exonic region. Deduplication is done by UMI.

Column name	Description
Intron_nUMIs	Number of deduplicated reads assigned to an intronic region. Deduplication is done by UMI.
Gene_nUMIs	Number of deduplicated reads assigned to a gene region (exon + intron). Deduplication is done by UMI.

The table below lists additional columns that will be present in the stats file when the input FASTQ files result from the SMARTer Stranded Total RNA-Seq Kit v3 - Pico Input Mammalian protocol.

Table 12. Additional columns in the stats file for the SMARTer Stranded Total RNA-Seq Kit v3 - Pico Input Mammalian protocol.

Column name	Description
No_of_UMIs	Number of UMI variations detected after demultiplexing.
Exon_nUMIs	Number of deduplicated reads assigned to an exonic region. Deduplication is done by UMI.
Exon_nUSSs	Number of deduplicated reads assigned to an exonic region. Deduplication is done by Unique Start&Stop Site (USS).
Exon_nUMIs_USSs	Number of deduplicated reads assigned to an exonic region. Deduplication is done by both UMI and USS.
Intron_nUMIs	Number of deduplicated reads assigned to an intronic region. Deduplication is done by UMI.
Intron_nUSSs	Number of deduplicated reads assigned to an intronic region. Deduplication is done by Unique Start&Stop Site (USS).
Intron_nUMIs_USSs	Number of deduplicated reads assigned to an intronic region. Deduplication is done by both UMI and USS.
Gene_nUMIs	Number of deduplicated reads assigned to a gene region (exon + intron). Deduplication is done by UMI.
Gene_nUSSs	Number of deduplicated reads assigned to a gene region (exon + intron). Deduplication is done by Unique Start&Stop Site (USS).
Gene_nUMIs_USSs	Number of deduplicated reads assigned to a gene region (exon + intron). Deduplication is done by both UMI and USS.
Strand_Specificity	Ratio of reads detected as correct strand after mapping to genome.

The table below lists an additional column that will be present in the stats file when transcript analysis was performed.

Table 13. Additional column in the stats file for transcript analysis.

Column name	Description
No_of_Transcripts	Number of detected transcripts.

2. Gene Matrix File

The gene matrix file (also referred to as the gene table or counts matrix) is also in CSV format and contains gene counts for each barcode, with the genes in the rows and barcodes/cells in the columns. The file contains raw counts that can then be normalized and transformed using CogentDS. An example is shown below.

GeneID	AACCGGTTTT	AACCGGTTTC	AACCGGTTTC	AACCGGTTTA	AACCGGTTAA	AACCGGTTAT	AACCGGTTCT	AGAGTTCTGT	AGAGTTCTTA	AGAGTTCTCG ...
ENSG00000116688_MFN2	63	0	0	0	0	0	0	0	0	0
ENSG00000226396_AL031727.1	47	0	34	72	71	60	11	72	64	64
ENSG00000142676_RPL11	6290	4229	339	1321	3972	946	1344	3300	2778	4289
ENSG00000117614_SYF2	1181	65	0	0	0	0	0	0	0	0
ENSG00000186501_TMEM222	146	0	0	0	0	0	0	0	0	0
ENSG00000158195_WASF2	1703	7357	1480	1	0	5	0	0	0	1
ENSG00000169403_PTAFR	793	13	0	0	0	0	1	0	0	186
ENSG00000126698_DNAJC8	2062	0	0	0	1	1	847	1051	1201	1422
ENSG00000060688_SNRNP40	228	0	0	0	0	0	0	0	0	0
ENSG00000243970_PPIEL	186	0	0	0	1	0	0	0	4	0
ENSG00000084072_PPIE	247	0	0	0	0	1	0	0	0	0
ENSG00000131238_PPT1	470	0	0	0	0	0	0	39	1642	0
ENSG00000236876_TMSB4XP1	13	6	5	0	17	15	23	11	12	5
ENSG00000065978_YBX1	1168	2460	1	0	0	1	0	0	183	1
ENSG00000066322_ELOVL1	1732	0	1	0	0	0	0	0	0	0
ENSG00000142937_RPS8	240	5632	317	1122	1229	1797	226	4364	1907	832
ENSG00000085832_EPS15	172	2	0	242	1	0	0	10	767	0
ENSG00000117054_ACADM	2787	0	0	0	0	0	0	0	0	0
ENSG00000183291_SELENOF	74	0	0	0	1592	0	0	366	563	0
ENSG00000228502_EEF1A1P11	16	5	10	4	7	3	0	3	8	0
...										

Figure 28. Example of a gene matrix file.

3. Gene Info File

The gene info file contains the main annotations for the genes as described in the GTF file that is part of the genome build.

Table 14. Columns in the `gene_info.csv` output file.

Column name	Description
Gene_ID	Gene ID used in CogentDS concatenated Ensembl_ID and Gene_Name.
Ensembl_ID	The ID used in Ensembl
Gene_Name	The gene symbol
Gene_Biotype	The gene classification
Gene_Length	The gene length, used for some normalizations.

An example file screenshot is shown below.

Gene_ID	Ensembl_ID	Gene_Name	Gene_Biotype	Gene_Length
ENSG00000223972_DDX11L1	ENSG00000223972	DDX11L1	transcribed_unprocessed_pseudogene	1735
ENSG00000227232_WASH7P	ENSG00000227232	WASH7P	unprocessed_pseudogene	1351
ENSG00000278267_MIR6859-1	ENSG00000278267	MIR6859-1	miRNA	68
ENSG00000243485_MIR1302-2HG	ENSG00000243485	MIR1302-2HG	lincRNA	1021
ENSG00000284332_MIR1302-2	ENSG00000284332	MIR1302-2	miRNA	138
ENSG00000237613_FAM138A	ENSG00000237613	FAM138A	lincRNA	1219
ENSG00000268020_OR4G4P	ENSG00000268020	OR4G4P	unprocessed_pseudogene	840
ENSG00000240361_OR4G11P	ENSG00000240361	OR4G11P	transcribed_unprocessed_pseudogene	1414
ENSG00000186092_OR4F5	ENSG00000186092	OR4F5	protein_coding	2618
ENSG00000238009_AL627309.1	ENSG00000238009	AL627309.1	lincRNA	3726
ENSG00000239945_AL627309.3	ENSG00000239945	AL627309.3	lincRNA	1319
ENSG00000233750_CICP27	ENSG00000233750	CICP27	processed_pseudogene	3812
ENSG00000268903_AL627309.6	ENSG00000268903	AL627309.6	processed_pseudogene	755
ENSG00000269981_AL627309.7	ENSG00000269981	AL627309.7	processed_pseudogene	284
ENSG00000239906_AL627309.2	ENSG00000239906	AL627309.2	antisense	323
ENSG00000241860_AL627309.5	ENSG00000241860	AL627309.5	processed_transcript	6195

Figure 29. Example of a gene info file.

B.   Transcript Analysis Files

These files are only generated when the transcript analysis option (Section V.D.1) is used.

Table 15. Raw data output files generated by CogentAP transcript analysis. Files can be found in the `transcript/` subfolder of the output folder defined during the analysis run.

Interface	Referred to as	File name
GUI	Gene transcript info	`transcript_info.csv`
	Transcript matrix	`analysis_transcript_matrix.csv`
CLI	Gene transcript info	`transcript_info.csv`
	Transcript matrix	`analyze_transcript_matrix.csv`

1.   Transcript Info File

The transcript info file contains the main annotation for the transcripts as described in the GTF file that is part of the genome build. This file has similar format to gene info file. In the case for transcript info file, both gene ID and transcript ID are included in the file.

Table 16. Columns in the `transcript_info.csv` output file.

Column name	Description
Transcript_ID	Transcript ID used in CogentDS concatenated Ensembl_ID and Transcript_Name.
Ensembl_ID	The ID used in Ensembl
Transcript_Name	The transcript symbol
Gene_ID	Gene ID that the transcript is derived from.
Gene_Name	The gene symbol that the transcript is derived from.
Transcript_Biotype	The transcript classification
Transcript_Length	The transcript length, used for some normalizations.

Transcript_ID	Ensembl_ID	Transcript_Name	Gene_ID	Gene_Name	Transcript_Biotype	Transcript_Length
ENST00000456328_DDX11L1-202	ENST00000456328	DDX11L1-202	ENSG00000223972	DDX11L1	processed_transcript	1657
ENST00000450305_DDX11L1-201	ENST00000450305	DDX11L1-201	ENSG00000223972	DDX11L1	transcribed_unprocessed_pseudogene	632
ENST00000488147_WASH7P-201	ENST00000488147	WASH7P-201	ENSG00000227232	WASH7P	unprocessed_pseudogene	1351
ENST00000619216_MIR6859-1-201	ENST00000619216	MIR6859-1-201	ENSG00000278267	MIR6859-1	miRNA	68
ENST00000473358_MIR1302-2HG-202	ENST00000473358	MIR1302-2HG-202	ENSG00000243485	MIR1302-2HG	lincRNA	712
ENST00000469289_MIR1302-2HG-201	ENST00000469289	MIR1302-2HG-201	ENSG00000243485	MIR1302-2HG	lincRNA	535
ENST00000607096_MIR1302-2-201	ENST00000607096	MIR1302-2-201	ENSG00000284332	MIR1302-2	miRNA	138
ENST00000417324_FAM138A-201	ENST00000417324	FAM138A-201	ENSG00000237613	FAM138A	lincRNA	1187
ENST00000461467_FAM138A-202	ENST00000461467	FAM138A-202	ENSG00000237613	FAM138A	lincRNA	590
ENST00000606857_OR4G4P-201	ENST00000606857	OR4G4P-201	ENSG00000268020	OR4G4P	unprocessed_pseudogene	840
ENST00000642116_OR4G11P-202	ENST00000642116	OR4G11P-202	ENSG00000240361	OR4G11P	processed_transcript	1414
ENST00000492842_OR4G11P-201	ENST00000492842	OR4G11P-201	ENSG00000240361	OR4G11P	transcribed_unprocessed_pseudogene	939
ENST00000641515_OR4F5-202	ENST00000641515	OR4F5-202	ENSG00000186092	OR4F5	protein_coding	2618
ENST00000335137_OR4F5-201	ENST00000335137	OR4F5-201	ENSG00000186092	OR4F5	protein_coding	1054
ENST00000466430_AL627309.1-201	ENST00000466430	AL627309.1-201	ENSG00000238009	AL627309.1	lincRNA	2748
ENST00000477740_AL627309.1-202	ENST00000477740	AL627309.1-202	ENSG00000238009	AL627309.1	lincRNA	491
ENST00000471248_AL627309.1-203	ENST00000471248	AL627309.1-203	ENSG00000238009	AL627309.1	lincRNA	629
ENST00000610542_AL627309.1-205	ENST00000610542	AL627309.1-205	ENSG00000238009	AL627309.1	lincRNA	723
ENST00000453576_AL627309.1-204	ENST00000453576	AL627309.1-204	ENSG00000238009	AL627309.1	lincRNA	336

Figure 30. Example of a transcript info file.

2. Transcript Matrix Files

The transcript matrix file contains transcript counts for each barcode, with the transcripts in the rows and barcodes/cells in the columns. The file contains raw counts that can then be normalized and transformed using CogentDS. If there are not enough reads to estimate read distribution, the RSEM program will raise an error. The error results in a value of '0' for all transcripts in the barcode.

TranscriptID	CTGGTCTT	GTCGTTCT	CATAATG(	TATACGG/	CATACTC(	TGATTCC(	CCGTACG(	TGGATCA/	TAGCGAG	GCCTATT(	GCGTTAC(	TCTCCTAG
ENST00000456328_DDX11L1-202	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000450305_DDX11L1-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000488147_WASH7P-201	5.95	0	0	79.6	8.66	0	0	0	0	4	0	16.21
ENST00000619216_MIR6859-1-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000473358_MIR1302-2HG-202	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000469289_MIR1302-2HG-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000607096_MIR1302-2-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000417324_FAM138A-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000461467_FAM138A-202	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000606857_OR4G4P-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000642116_OR4G11P-202	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000492842_OR4G11P-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000641515_OR4F5-202	0	1	0	0	0	0	0	0	0	0	0	0
ENST00000335137_OR4F5-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000466430_AL627309.1-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000477740_AL627309.1-202	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000471248_AL627309.1-203	0	0	0	0	0	0	0	0	0	3	0	0
ENST00000610542_AL627309.1-205	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000453576_AL627309.1-204	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000495576_AL627309.3-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000442987_CICP27-201	0	0	0	0	0	0	0	0	0	0	0	0
ENST00000494149_AL627309.6-201	9.83	0	62.91	32.97	0	0	0	12	0	0	11.93	43.45

Figure 31. Example of a transcript matrix file.

C.   Gene Fusion Files

These files are only generated when the gene fusion option (Section V.D.2) is used.

Table 17. Raw data output files generated by CogentAP fusion analysis. Files can be found in the `gene_fusion/` subfolder of the output folder defined during the analysis run.

Interface	Referred to as	File name
GUI	Junction matrix	`analysis_junction_matrix.csv`
	Spanning matrix	`analysis_spanning_matrix.csv`
	Fusion_stats	`analysis_stats.csv`
CLI	Junction matrix	`analyze_junction_matrix.csv`
	Spanning matrix	`analyze_spanning_matrix.csv`
	Fusion_stats	`analyze_stats.csv`

1. Junction matrix file

The junction matrix file contains junction read counts for each barcode. In the table, each gene fusion is a row, with the index barcodes (i.e., a cell) in the columns. The table values represent the number of reads detected tagged with the specified barcode that contain the corresponding fusion.

GeneFusion	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AAGGTCTG	AAGGTCTG	AAGGTCTG	...
TPTE2P2--MALAT1	71	52	44	59	43	0	13	86	92	11	
TPTE2P2--RPL37	13	9	5	9	13	0	8	5	33	4	
MSH2--MRPS18A	9	0	0	0	0	0	0	0	0	0	
TPTE2P2--PPIAP29	7	3	1	1	2	0	0	2	0	2	
GPAT2--PFN1	9	0	0	1	1	0	0	0	0	1	
WDR35--PFN1	7	0	0	0	5	0	2	1	7	2	
RPS28--CHST2	5	0	0	0	0	0	0	0	0	0	
TPTE2P2--B2M	7	1	10	2	13	7	4	1	13	4	
CCNL1--YWHAQ	4	0	0	0	0	0	0	0	0	0	
SEPTIN9--MALAT1	2	0	0	0	0	0	0	0	0	0	
KNG1--RN7SL2	5	2	0	0	0	1	4	3	1	3	
KNG1--AL627171.4	5	2	0	0	0	1	4	3	1	3	
TPTE2P2--ACTG1	5	0	0	0	0	0	0	0	0	2	
STAT1--CHSY1	5	0	0	0	0	0	0	0	0	0	
PXN--ARAP2	3	0	0	0	0	0	0	0	0	0	
ACTB--MYL12A	4	0	0	0	0	0	0	0	0	0	
UCK2--MEAF6	3	0	0	0	0	0	0	0	0	0	
AL135905.2--LMAN2	4	0	0	0	0	0	0	0	0	0	
PLEK--OBSCN	6	0	0	0	0	0	0	0	0	0	
TMEM59--GRK6	3	0	0	0	0	0	0	0	0	0	
...											

Figure 32. Example of a junction matrix file.

2. Spanning matrix file

The spanning matrix file contains spanning read counts for each barcode, with the gene fusion in the rows and barcodes/cells in the columns. Each value is the number of paired-end reads containing the sequences of both genes that form the corresponding fusion.

GeneFusion	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AACCGGTT	AAGGTCTG	AAGGTCTG	AAGGTCTG	...
TPTE2P2--MALAT1	0	0	0	0	0	0	0	0	0	0	
TPTE2P2--RPL37	0	0	0	0	0	0	0	0	0	0	
MSH2--MRPS18A	16	0	0	0	0	0	0	0	0	0	
TPTE2P2--PPIAP29	0	0	0	0	0	0	0	0	0	0	
GPAT2--PFN1	0	0	0	0	0	0	0	0	0	0	
WDR35--PFN1	0	0	0	0	0	0	0	0	0	0	
RPS28--CHST2	8	0	0	0	0	0	0	0	0	0	
TPTE2P2--B2M	0	0	0	0	0	0	0	0	0	0	
CCNL1--YWHAQ	6	0	0	0	0	0	0	0	0	0	
SEPTIN9--MALAT1	14	0	0	0	0	0	0	0	0	0	
KNG1--RN7SL2	0	0	0	0	0	0	0	0	0	0	
KNG1--AL627171.4	0	0	0	0	0	0	0	0	0	0	
TPTE2P2--ACTG1	0	0	0	0	0	0	0	0	0	0	
STAT1--CHSY1	7	0	0	0	0	0	0	0	0	0	
PXN--ARAP2	11	0	0	0	0	0	0	0	0	0	
ACTB--MYL12A	5	0	0	0	0	0	0	0	0	0	
UCK2--MEAF6	9	0	0	0	0	0	0	0	0	0	
AL135905.2--LMAN2	0	0	0	0	0	0	0	0	0	0	
PLEK--OBSCN	4	0	0	0	0	0	0	0	0	0	
TMEM59--GRK6	4	0	0	0	0	0	0	0	0	0	
...											

Figure 33. Example of a spanning matrix file.

3. Stats file

When gene fusion analysis is run, CogentAP creates a second analysis_stats file in the `gene_fusion/` subfolder. Like the default stats file ([Appendix.A.1](#)), it contains barcode-level statistics: the number of reads and the number of fusions detected per barcode.

Table 18. Columns in the `gene fusion/*_analysis_stats.csv` output file.

Column name	Description
Barcode	Detected barcodes. This value will usually be the sample name from the well-list or well-list-like file, but there are three exceptions, documented in [Table 10](#).
Sample	Sample names described in sample description file.
Barcoded_Reads	Number of reads after demultiplexing.
No_of_Fusions	Number of detected fusions.

An example file screenshot is shown below.

Barcode	Sample	Barcoded_Reads	No_of_Fusions
ATGAATAGCAGGACCA	Sample	499560	168
GATGCGTTTATTGAAC	Sample	498992	296
AAGGTCTGTAGATGAC	Sample	498725	314
TCTAGGTTCGTTCCGA	Sample	498662	229
TCGAACGAATACTTGA	Sample	497330	187
GGTAGAGACAATAGTC	Sample	497173	365
CCGCAGTCCGTTGGTT	Sample	494496	77
TGCGCGTTACCATTAT	Sample	494008	170
CCTAACTATTGCGGAC	Sample	493394	86
GGAGAAGCCAATCTTG	Sample	493204	389
GCTGCTTCCAATGGAT	Sample	493001	296
GCAACTAGTTGCCGCT	Sample	492685	93
AGCGTCTTTGGCGGTT	Sample	492404	279
GAGATTCTTAGGCTCT	Sample	492042	236
ATACGTTCCAATTCGG	Sample	491834	395

Figure 34. Example of the `*_analysis_stats.csv` file output specific to gene fusion analysis.

D.   Immune Profiling Files

These files are only generated when the immune profiling option ([Section V.D.3](#)) is used.

Table 19. Raw data output files generated by CogentAP immune analysis. Files can be found in the **immune/** subfolder of the output folder defined during the analysis run.

Interface	Referred to as	File name
GUI	Clonotype matrix	`analysis_clonotype_matrix.csv`
	Metadata	`analysis_metadata.csv`
	Summary	`analysis_full_summary.csv`
	Top 3 clonotype matrix	`analysis_top3_matrix.csv`
	Top 3 metadata	`analysis_top3_meta.csv`
	Top 3 summary	`analysis_top3_summary.csv`
CLI	Clonotype matrix	`analyze_clonotype_matrix.csv`
	Metadata	`analyze_metadata.csv`
	Summary	`analyze_full_summary.csv`
CLI	Top 3 clonotype matrix	`analyze_top3_matrix.csv`
	Top 3 metadata	`analyze_top3_meta.csv`
	Top 3 summary	`analyze_top3_summary.csv`

1. clonotype_matrix

The clonotype_matrix file contains clonotype counts for each barcode, with the clonotype by rows and barcodes (i.e., cells) in the columns. The file contains raw counts that can then be normalized and transformed using CogentDS.

The clonotype is defined as joining of V, D, and J genes, constant regions (C), and CDR3 amino acid (CDR3aa) sequences, connected by the dollar-sign ($) symbol.

```
<V gene>$<D gene>$<J gene>$<constant region><$CDR3 aa>
```

A period (`.`) is used in place of a segment that doesn't exist in the clonotype.

Examples:

```
TRBV20-1*01$TRBD2*02$TRBJZ-7*01$TRBC$CSAGSGRGGRAVEQYF

IGKV4-1*01$.$IGKJ4*01$IGKC$CQQYYSTPALTF
```

In the second clonotype, the D gene isn't present, so the period is used.

Table 20. Columns in the `*_clonotype_matrix.csv` output file.

Column name	Description
V-D-J-C-CDR3aa	The string of V, D, and J genes, constant region, and CDR3 amino acid segment details, concatenated by the $ symbol.
<barcode1> ... ... **<barcode*N*>**	Subsequent columns correspond to the joint clonotype segments identified for the barcode listed in the column header. The cell values are a count of the clonotype reads found for the V-D-J-C-CDR3aa combination.

An example file screenshot is shown below.

V-D-J-C-CDR3aa	AATGGTAAT	CATAATGGT	CGCGGTCGT	CGCGGTCGT	TTGTAATAG	CGTAATGGT	CGAAGTCGT	CGTTGTCGT
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDAEISTGYYLPYYFDYW	0	3	0	0	0	3	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDEFSTGYYLPYYFDYW	0	5	0	0	0	5	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDEISTGYYIPYYFDYW	0	10	0	0	0	10	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDEISTGYYLPYYFDYW	0	35521	0	0	0	0	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDENSTGYYLPYYFDYW	0	3	0	0	0	3	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFGDEISTGYYLPYYFDYW	0	5	0	0	0	5	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVLDDEISTGYYLPYYFDYW	0	5	0	0	0	5	0	288
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CGRVFDDEISTGYYLPYYFDYW	0	7	0	0	0	7	0	0
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$RARVFDDEISTGYYLPYYFDYW	0	16	0	0	0	16	0	0
IGKV1-12*01$.$IGKJ5*01$IGKC$CQQASSFPPVTF	3	0	0	0	3	0	0	0
IGKV4-1*01$.$IGKJ4*01$IGKC$CQQYYSTPALTF	0	0	0	166	0	0	0	0
IGKV4-1*01$.$IGKJ4*01$IGKC$CQ_YYSTPALTF	0	0	0	4	0	0	0	3
IGKV4-1*01$.$IGKJ4*01$IGKC$RQQYYSTPALTF	0	0	0	3	0	0	0	0
IGKV4-1*01$.$IGKJ4*01$IGKC$SQQYNSTPALTF	0	0	0	2	0	0	0	1
IGLV3-19*01$.$IGLJ2*01$IGLC$CNSRDTSDNHLVF	0	7731	0	0	0	532	0	0
TRAV8-2*01$.$TRAJ12*01$.$CVVSPMASSYKLIF	0	0	2	0	0	0	2	0
TRAV8-2*01$.$TRAJ12*01$TRAC$CVVSPMDSSYKLIF	0	0	5	0	0	0	5	0
TRBV20-1*01$TRBD2*02$TRBJ2-7*01$TRBC$CSAGSGRGGRAVEQYF	0	0	22	0	0	0	22	0

Figure 35. Example of a clonotype matrix file.

2. top3 _matrix

The top3_matrix file is identical in format to the clonotype matrix file, but only contains information based on the three clonotypes identified with the maximum reads in the dataset, which correspond to the top3 rows of an intermediate raw clonotype file.

3. metadata

The metadata file contains clonotypes and their V, D, J, and C segment details, corresponding CDR3 amino acid sequences, and two columns with a boolean value 'Y' or 'N' to mark if the clonotype is a light or heavy chain. Similar to the clonotype_matrix file above, the clonotype is defined as joining of V, D, and J genes, constant region (C), and the CDR3 amino acid sequences (CDR3aa), connected by the $ symbol.

Table 21. Columns in the `*_metadata.csv` output file.

Column name	Description
V-D-J-C-CDR3aa	The string of V, D, and J genes, constant region, and CDR3 amino acid segment details, concatenated by the $ symbol.
V	V segment of the clonotype.
D	D segment of the clonotype.
J	J segment of the clonotype.
C	C segment of the clonotype.
CDR3aa	CDR3 amino acid segment of the clonotype.
Light Chain	Boolean value (Y or N). A 'Y' value designates the clonotype as a light chain.
Heavy Chain	Boolean value (Y or N). A 'Y' value designates the clonotype as a heavy chain.

An example file screenshot is shown below.

V-D-J-C-CDR3aa	V	D	J	C	CDR3aa	Light Chain	Heavy Chain
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDAEISTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVFDAEISTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDEFSTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVFDDEFSTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDEISTGYYIPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVFDDEISTGYYIPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDEISTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVFDDEISTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFDDENSTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVFDDENSTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVFGDEISTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVFGDEISTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CARVLDDEISTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CARVLDDEISTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$CGRVFDDEISTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	CGRVFDDEISTGYYLPYYFDYW	N	Y
IGHV4-61*01$IGHD3-9*01$IGHJ4*02$IGHG1$RARVFDDEISTGYYLPYYFDYW	IGHV4-61*01	IGHD3-9*01	IGHJ4*02	IGHG1	RARVFDDEISTGYYLPYYFDYW	N	Y
IGKV1-12*01$.$IGKJ5*01$IGKC$CQQASSFPPVTF	IGKV1-12*01	.	IGKJ5*01	IGKC	CQQASSFPPVTF	Y	N
IGKV4-1*01$.$IGKJ4*01$IGKC$CQQYYSTPALTF	IGKV4-1*01	.	IGKJ4*01	IGKC	CQQYYSTPALTF	Y	N
IGKV4-1*01$.$IGKJ4*01$IGKC$CQ_YYSTPALTF	IGKV4-1*01	.	IGKJ4*01	IGKC	CQ_YYSTPALTF	Y	N
IGKV4-1*01$.$IGKJ4*01$IGKC$RQQYYSTPALTF	IGKV4-1*01	.	IGKJ4*01	IGKC	RQQYYSTPALTF	Y	N
IGKV4-1*01$.$IGKJ4*01$IGKC$SQQYNSTPALTF	IGKV4-1*01	.	IGKJ4*01	IGKC	SQQYNSTPALTF	Y	N
IGLV3-19*01$.$IGLJ2*01$IGLC$CNSRDTSDNHLVF	IGLV3-19*01	.	IGLJ2*01	IGLC	CNSRDTSDNHLVF	Y	N
TRAV8-2*01$.$TRAJ12*01$.$CVVSPMASSYKLIF	TRAV8-2*01	.	TRAJ12*01	.	CVVSPMASSYKLIF	Y	N
TRAV8-2*01$.$TRAJ12*01$TRAC$CVVSPMDSSYKLIF	TRAV8-2*01	.	TRAJ12*01	TRAC	CVVSPMDSSYKLIF	Y	N
TRBV20-1*01$TRBD2*02$TRBJ2-7*01$TRBC$CSAGSGRGGRAVEQYF	TRBV20-1*01	TRBD2*02	TRBJ2-7*01	TRBC	CSAGSGRGGRAVEQYF	N	Y

Figure 36. Example of a clonotype metadata file.

4. **top3_meta**

The `top3_meta` file is identical in format to the `metadata` file, but only contains information based on the three clonotypes identified with the maximum reads in the dataset, which correspond to the top3 lines of raw clonotype output file.

5. **full_summary**

The full_summary file contains barcodes, total reads, and clonotypes identified per barcode. It has 62 columns, which includes four components: barcode, total reads, clonotype category marks, and segment call details.

- The first two columns in the table are the barcode and total reads.

- Columns 3–14 are clonotype categories, which mark and summarize if a clonotype category was identified for the barcode, i.e.,
    o (For TCR): TRA, TRB, TRD, and TRG
    o (For BCR): IGG, IGD, IGA, IGM, IGE, IGH, IGK, and IGL
    If a category is not identified, the cell value is left blank.

- Columns 15–62 are listed V, D, and J genes and constant region details. For example: IGHV-1, IGHD3-1, IGHJ4 and IGHG1 for the IGH category (i.e., four columns per clonotype category).

Table 22. Columns in the `*_full_summary.csv` output file.

Row #	Column name	Description
1	**Barcode**	The barcode sequence of a single cell
2	**Total_Reads**	Number of reads with clonotype type identified for corresponding barcode.
3–6	**TRA, TRB, TRD, TRG**	Categories for T-cell (TCR) chain types. The cell value will match the column header if the chain type is detected with the given barcode. If it is not detected, the cell will be blank.
7–14	**IGG, IGD, IGA, IGM, IGE, IGH, IGK, IGL**	Categories for B-cell (BCR) chain types. The cell value will match the column header if the chain type is detected with the given barcode. If it is not detected, the cell will be blank.

Row #	Column name	Description
15–18	TRA_V, TRA_D, TRA_J, TRA_C	(TCR) V, D, and J genes and constant region identified within the TRA chain type.
19–22	TRB_V, TRB_D, TRB_J, TRB_C	(TCR) V, D, and J genes and constant region identified within the TRB chain type.
23–26	TRD_V, TRD_D, TRD_J, TRD_C	(TCR) V, D, and J genes and constant region identified within the TRD chain type.
27–30	TRG_V, TRG_D, TRG_J, TRG_C	(TCR) V, D, and J genes and constant region identified within the TRG chain type.
31–34	IGG_V, IGG_D, IGG_J, IGG_C	(BCR) V, D, and J genes and constant region identified within the IgG chain type.
35–38	IGD_V, IGD_D, IGD_J, IGD_C	(BCR) V, D, and J genes and constant region identified within the IgD chain type.
39–42	IGA_V, IGA_D, IGA_J, IGA_C	(BCR) V, D, and J genes and constant region identified within the IgA chain type.
43–46	IGM_V, IGM_D, IGM_J, IGM_C	(BCR) V, D, and J genes and constant region identified within the IgM chain type.
47–50	IGE_V, IGE_D, IGE_J, IGE_C	(BCR) V, D, and J genes and constant region identified within the IgE chain type.
51–54	IGH_V, IGH_D, IGH_J, IGH_C	(BCR) V, D, and J genes and constant region identified within the IgH chain type.
55–58	IGK_V, IGK_D, IGK_J, IGK_C	(BCR) V, D, and J genes and constant region identified within the IgK chain type.
59–62	IGL_V, IGL_D, IGL_J, IGL_C	(BCR) V, D, and J genes and constant region identified within the IgL chain type.

An example file screenshot is shown below.

Figure 37. Example of a clonotype summary file.

6. top3_summary

The top3_summary file is identical in format to the summary file, but only contains information based on the three clonotypes identified with the maximum reads in the dataset, which correspond to the top3 lines of raw clonotype output file.

Contact Us	
Customer Service/Ordering	**Technical Support**
tel: 800.662.2566 (toll-free)	tel: 800.662.2566 (toll-free)
fax: 800.424.1350 (toll-free)	fax: 800.424.1350 (toll-free)
web: takarabio.com/service	web: takarabio.com/support
e-mail: ordersUS@takarabio.com	e-mail: technical_support@takarabio.com

Notice to Purchaser

Our products are to be used for **Research Use Only**. They may not be used for any other purpose, including, but not limited to, use in humans, therapeutic or diagnostic use, or commercial use of any kind. Our products may not be transferred to third parties, resold, modified for resale, or used to manufacture commercial products or to provide a service to third parties without our prior written approval.

Your use of this product is also subject to compliance with any applicable licensing requirements described on the product's web page at takarabio.com. It is your responsibility to review, understand and adhere to any restrictions imposed by such statements

This document has been reviewed and approved by the Quality Department.