AutoTCR V 1.0 Documentation

Ben K. Margetts
ucbpmar@ucl.ac.uk

August 30, 2017

This guide has been written to demonstrate how to use the AutoTCR R markdown script for the T cell receptor repertoire Decombinator bioinformatics pipeline.

AutoTCR takes a series of raw .fastq files as input alongside indexes, runs a series of modified Python (V 2.7) Decombinator scripts over the data, and produces 5 sets of output files. The script then selectively reads these output files in and produces summary analyses from the data.

AutoTCR prioritises efficiency over complex analysis and is designed for batch analysis of TCR sequencing data.

1 Prerequisites

To run the Decombinator scripts, a valid Python 2.7 installation is required alongside a number of specific packages. Details can be found on the Decombinator GitHub page. For ease of installation, I recommend installing the anaconda Python (V 2.7) distribution.

Secondly, to run the R markdown scripts, a 64 bit installation of R is required, alongside RStudio and a number of packages. At the time of writing, the most current versions of the following CRAN packages were used when writing AutoTCR: data.table, zoo, plyr, scales, ggplot2, ggthemes, dplyr, plotly, pracma, Hmisc, rpart, survminer, knitr, mailR, grid, sringr, ggrepel, ggjoy, RColorBrewer, strigdist, ape, dendextend, reshape2, gplots. More may be required in future versions.

To see an up to date list of required R packages, please visit the second chunk of the script. Most, if not all packages can be installed using the 'install.packages()' command, with the package given as a string. For example:

```
install.packages('data.table')
```

It can then be loaded with

```
library(data.table)
```

2 How AutoTCR works

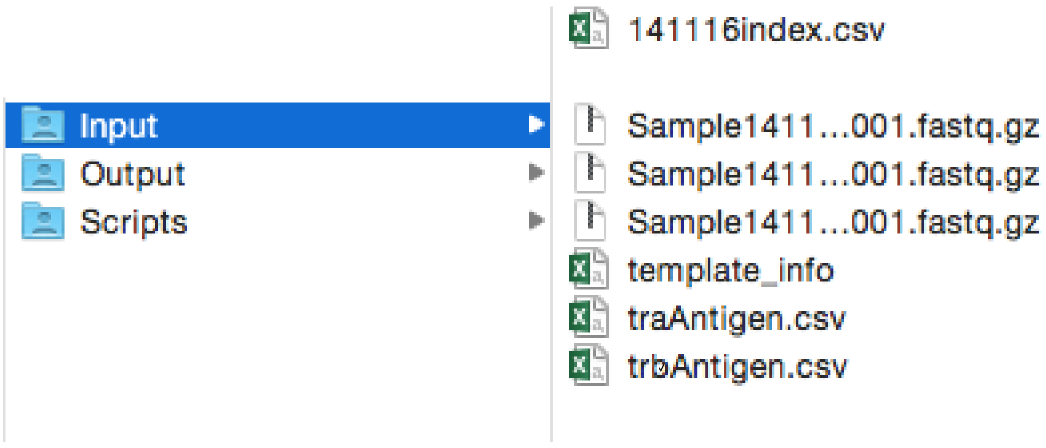
- The script first looks for an expected file structure and the necessary files. 3 files in a single directory: Input, Output, and Scripts.
- AutoTCR first calls Python and runs the Demultiplexor script over the input files (R1, R2, and I1).
- All Demultiplexor files are saved in the Output file.

- AutoTCR next runs Decombinator over the Demultiplexor output files and saves the Decombinator output separately. It does this once for alpha chain files and once for beta chain files
- AutoTCR then runs Collapsinator over all Decombinator output, saving the Collapsinator output seperately.
- Following this, AutoTCR translates the Collapsinator output into CDR3s using the CDR3Translator script. It does this once for alpha chains and once for beta chains, saving the output seperately.
- AutoTCR then repeats the following step, but this time produces a more detailed output file containing data on V and J gene usage, along with nonproductive sequences.
- With this output data generated, AutoTCR calls GeneNameTranslator and runs this over all detailed CDR3Translator output, saving the gene name translated files seperately.
- AutoTCR then reads some of these data back in, focussing on the CDR3 files, the non-productive sequence files, and the gene name files.
- AutoTCR pulls apart the file names from these output files to assign a chain, ID, and month to the sample.
- AutoTCR uses these data to build a number of datasets for further analysis
- AutoTCR builds a Plots folder in the Output directory, going through each analysis in-turn and saving the output.

As stated, AutoTCR needs 3 folders inside a single directory. Input, Output, and Scripts. The next sections will go through these in-turn.

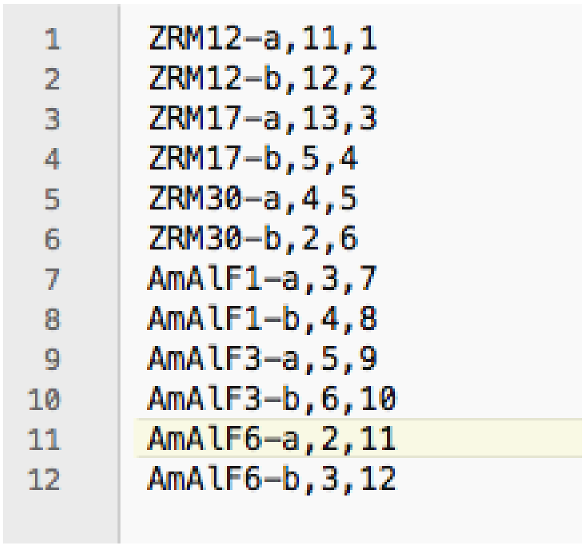
3 Input

This folder must be made by the user and will contain 3 .fastq files (R1, R2, and I1), a template info file (provided), 2 trAntigen files for VDJ specificity matching (provided), and an index file as demonstrated below.


Input
Output
Scripts
141116index.csv
Sample1411...001.fastq.gz
Sample1411...001.fastq.gz
Sample1411...001.fastq.gz
template_info
traAntigen.csv
trbAntigen.csv

The index file is the only file that requires significant user input. It is a .csv file and must be made by the user. It can be made in any text editing software (Excel is not recommended) and details the samples in the sequencing run (with a specific naming system required), and the forward and reverse indexes used for read identification.

The index file should look like this in the text editing software:



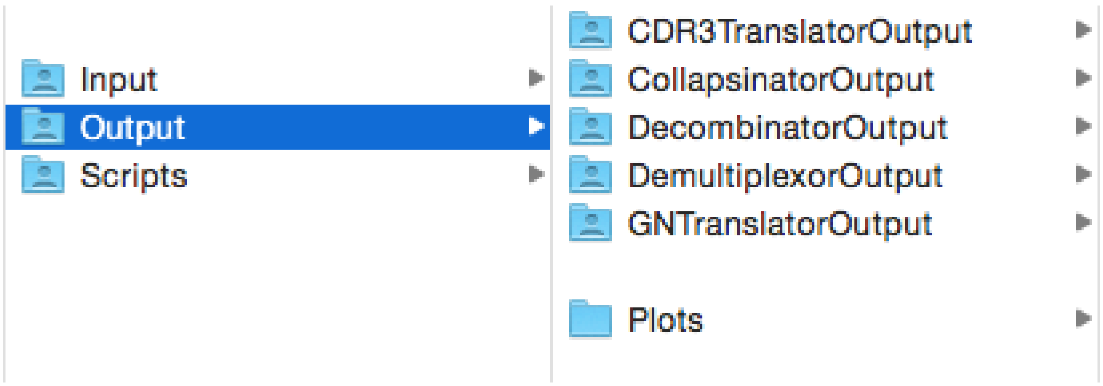
```
ZRM12-a,11,1
ZRM12-b,12,2
ZRM17-a,13,3
ZRM17-b,5,4
ZRM30-a,4,5
ZRM30-b,2,6
AmAlF1-a,3,7
AmAlF1-b,4,8
AmAlF3-a,5,9
AmAlF3-b,6,10
AmAlF6-a,2,11
AmAlF6-b,3,12
```

The first item will be a name i.e. ZRM12-a. This naming system is used by AutoTCR to analyse your data and it will not function without it. The naming system follows the following convention: 'IDmonth-chain' or 'ZRM12-a', where 'ZRM' is the patient ID, 12 is the month, and '-a' (alpha) is the chain.

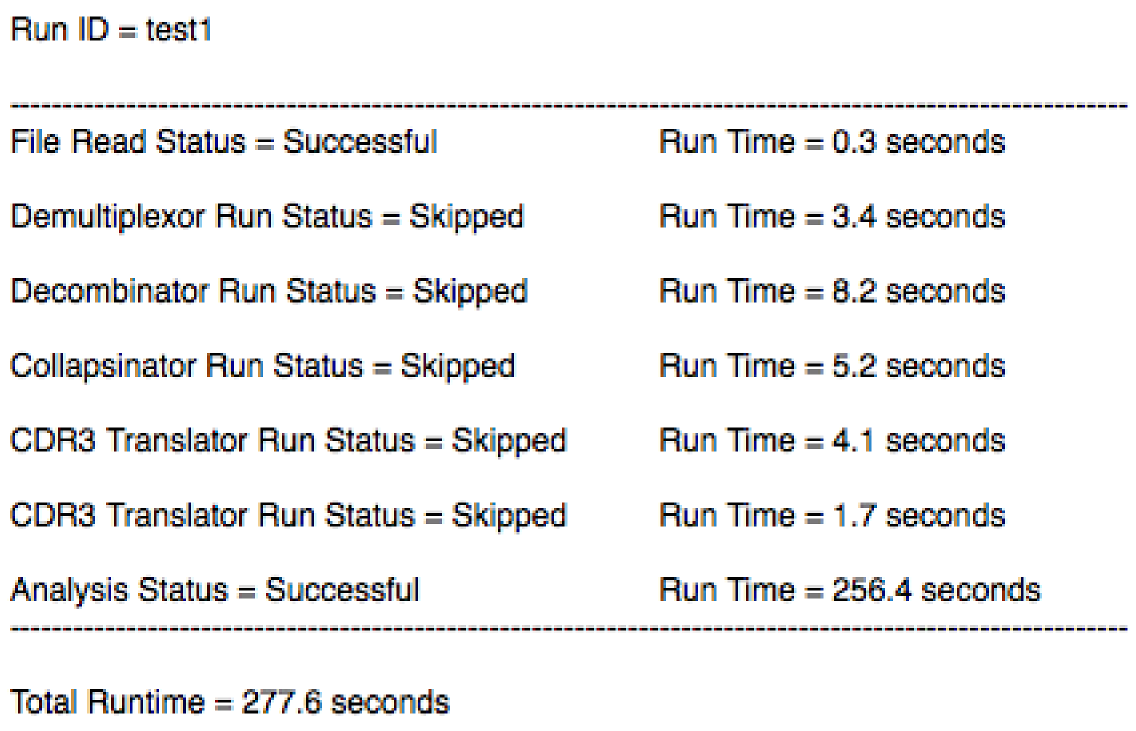
This index filename should then be appended with 'index.csv'.

4 Output

If AutoTCR has run successfully, it should produce the following output files and send you an email notifying you of this.



A successful run notification email will look like this:

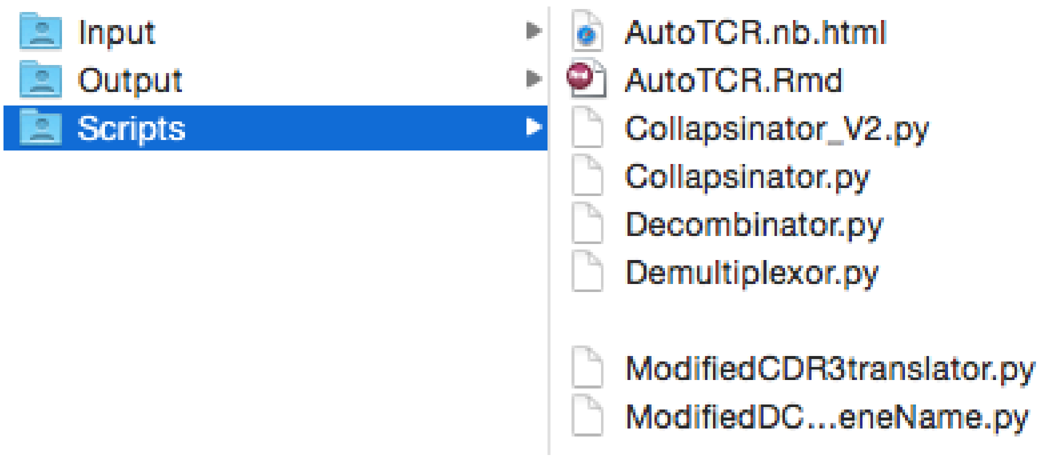


```
Run ID = test1

--------------------------------------------------------------------------------
File Read Status = Successful                Run Time = 0.3 seconds

Demultiplexor Run Status = Skipped           Run Time = 3.4 seconds

Decombinator Run Status = Skipped            Run Time = 8.2 seconds

Collapsinator Run Status = Skipped           Run Time = 5.2 seconds

CDR3 Translator Run Status = Skipped         Run Time = 4.1 seconds

CDR3 Translator Run Status = Skipped         Run Time = 1.7 seconds

Analysis Status = Successful                 Run Time = 256.4 seconds
--------------------------------------------------------------------------------

Total Runtime = 277.6 seconds
```

The email will notify you which steps ran, which failed, and which it skipped due to them having been run previously on the data.

5 Scripts

To run all of this, AutoTCR needs a file containing a number of original and modified scripts, along with the AutoTCR R Markdown file. These have all been provided. A functional Scripts folder should look like this:

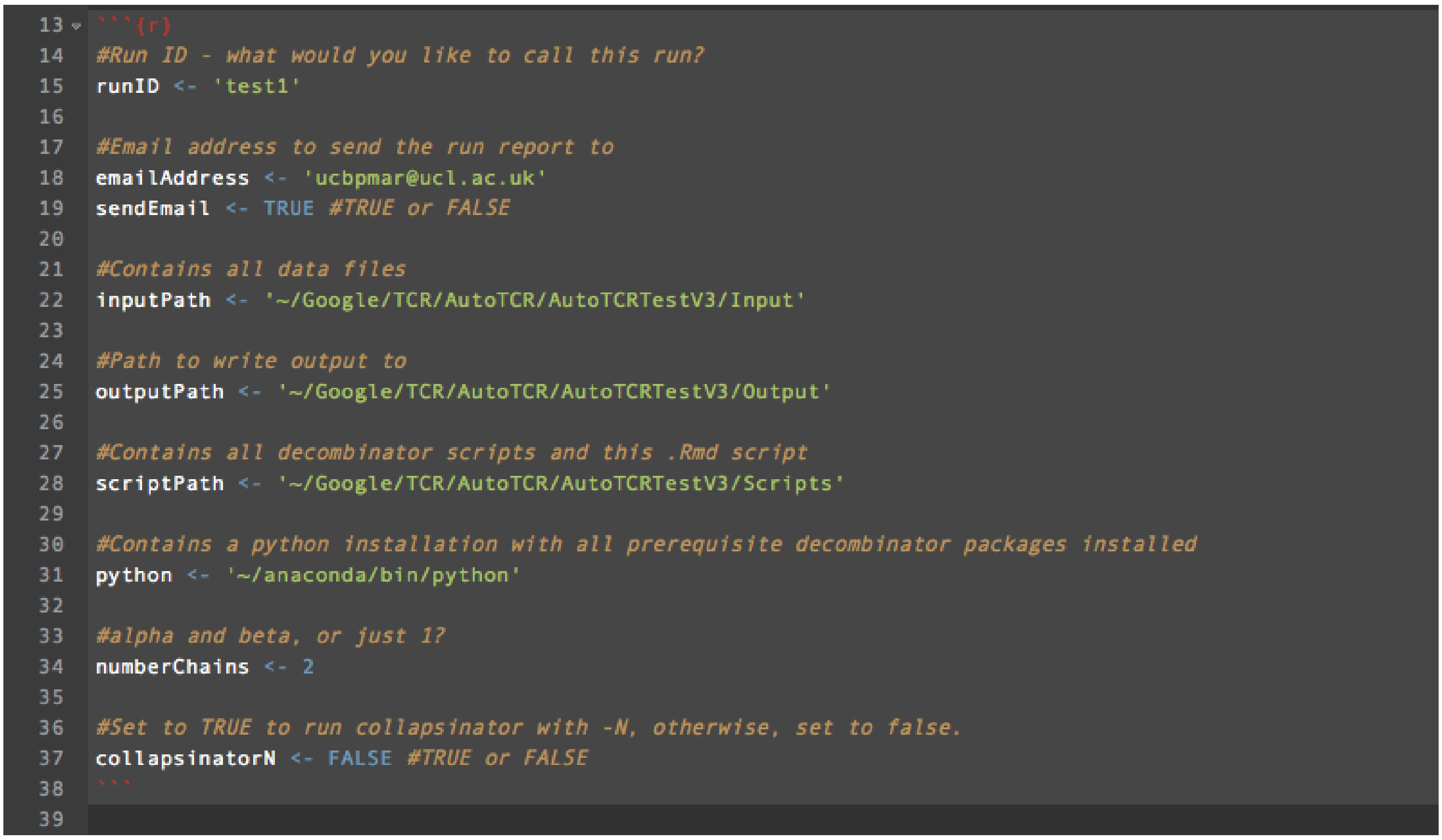


Input
Output
Scripts

AutoTCR.nb.html
AutoTCR.Rmd
Collapsinator_V2.py
Collapsinator.py
Decombinator.py
Demultiplexor.py

ModifiedCDR3translator.py
ModifiedDC...eneName.py

The only file that needs to be edited is the 'AutoTCR.Rmd' file. If you open it up with RStudio, you should see the following section, this is the only section that requires editing.



~~~r
```{r}
#Run ID - what would you like to call this run?
runID <- 'test1'

#Email address to send the run report to
emailAddress <- 'ucbpmar@ucl.ac.uk'
sendEmail <- TRUE #TRUE or FALSE

#Contains all data files
inputPath <- '~/Google/TCR/AutoTCR/AutoTCRTestV3/Input'

#Path to write output to
outputPath <- '~/Google/TCR/AutoTCR/AutoTCRTestV3/Output'

#Contains all decombinator scripts and this .Rmd script
scriptPath <- '~/Google/TCR/AutoTCR/AutoTCRTestV3/Scripts'

#Contains a python installation with all prerequisite decombinator packages installed
python <- '~/anaconda/bin/python'

#alpha and beta, or just 1?
numberChains <- 2

#Set to TRUE to run collapsinator with -N, otherwise, set to false.
collapsinatorN <- FALSE #TRUE or FALSE
~~~
```

- The **runID** variable gives the name that the run will be assigned. This will be used for notification emails. String expected.
- The **emailAddress** variable gives the email address to send the run notification email to. String expected.
- the **sendEmail** variable controls whether a notification email should be sent. TRUE will send an email, FALSE won't. Boolean expected.
- the **inputPath** variable gives the path containing the Input folder expected for AutoTCR to run. String expected.
- the **outputPath** variable gives the path containing the Output folder expected for AutoTCR to store output in. String expected.
- the **scriptPath** variable gives the path containing the Scripts folder expected for AutoTCR to run scripts out of. String expected.
- the **python** variable gives the path containing the python installation with the installed module prerequisites. String expected. If unsure of where this is installed, run: 'which python' in a terminal to display the path of the default python installation on your computer.
- the **numberChains** variable gives the number of chains sequenced. 1 for just alpha or beta, 2 for alpha and beta. Int expected.
- the **collapsinatorN** variable controls whether Collapsinator should allow undetermined bases in the CDR3 fuzzy matching algorithm. Boolean expected. Only set to TRUE if the CDR3Translator step is failing and crashing AutoTCR. Suggests poor quality output
```

should be expected.

After this has been configured, the R Markdown script can be left to run. It will generally take  3 - 4 hours per run, but this is highly dependent on the computer that it is run on and the size of the input files.