CRISPRitz Installation and Usage

The two fastest way to use CRISPRitz is through the installation of Docker or Conda.
Here we summarize the steps to install CRISPRitz with Docker and Conda.

Installation (Phase 1):

Conda installation (Linux and MacOS):

- Open a terminal window
- Paste this command into the terminal (Linux):
  `curl https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh --output Miniconda3-latest-Linux-x86_64.sh`
- Paste this command into the terminal (MacOS):
  `curl https://repo.anaconda.com/miniconda/Miniconda3-latest-MacOSX-x86_64.sh --output Miniconda3-latest-MacOSX-x86_64.sh`
- If the file is correctly downloaded you now need to execute it to complete the installation, so paste this command into the terminal:
  `bash Miniconda3-latest-Linux-x86_64.sh` (Linux)
  `bash Miniconda3-latest-MacOSX-x86_64.sh`(MacOS):

- Press ENTER when requested and yes when an answer is requested, in this way you allow conda to set all the directories in your HOME path for an easy use
- After the complete installation you will receive this message “Thank you for installing Miniconda3!” to certify the correct installation.
- Now you need to close the terminal window you are using and open a new one, to allow the system to start conda.
- In the new terminal window you should see something like this

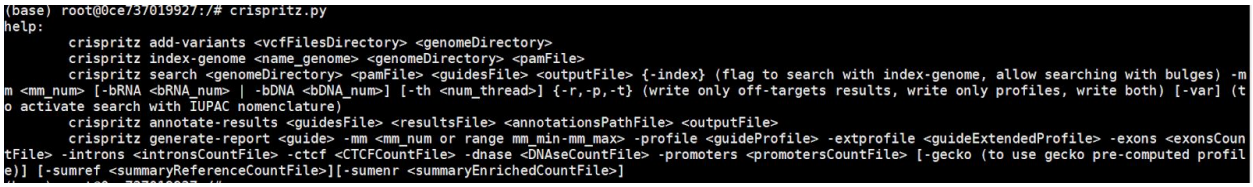
  “(base) user@nameofPC:~$” if you read the “(base)” like this, conda is loaded correctly and you can start using it.

- Now you need to set the channels to allow conda to access different repositories and set the default version of python to version 3.6, so paste these commands into the terminal you just opened:

  ```
  conda config --add channels defaults

  conda config --add channels bioconda

  conda config --add channels conda-forge

  conda install python=3.6
  ```

- Now, you can install CRISPRitz by typing the command:
  `conda install crispritz`
- To test your installation, type the command:
  `crispritz.py`

- After the execution of the command you should see a list of CRISPRitz tools.



Now the software is installed and ready to be used.

Docker installation:
Note: if you are using MasOS or Windows, you just need to download the installer file and follow the on screen instructions.
https://docs.docker.com/docker-for-windows/install/ (Windows)
https://docs.docker.com/docker-for-mac/install/ (MacOS)

Ubuntu installation guide:

- Open a terminal window
- Paste this command to update the index repository:
  ```
  sudo apt-get update
  ```
- Paste this command to allow package installation over HTTPS:
  ```
  sudo apt-get install \
      apt-transport-https \
      ca-certificates \
      curl \
      gnupg-agent \
      software-properties-common
  ```
- Paste this command to add the docker key:
  ```
  curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
  ```
- Paste this command to set the correct version of docker for your system:
  ```
  sudo add-apt-repository \
     "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
     $(lsb_release -cs) \
     Stable"
  ```
- Paste this command to update the index repository another time, to make sure everything is ready and set to install docker:
  ```
  sudo apt-get update
  ```
- Then paste this command to finally install docker:
  ```
  sudo apt-get install docker-ce docker-ce-cli containerd.io
  ```
- Paste this last command to check if the installation is complete and functional:
  ```
  sudo docker run hello-world
  ```

- If this message is printed, everything is perfectly installed

```
(base) scancellieri@samuelecancellieri:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

- Now, we need to do some more steps to complete the settings. Paste this command to create a user group for docker user:
  `sudo groupadd docker`
- Paste this command to add your current user to the created group:
  `sudo usermod -aG docker $USER`
- Now you need to restart your machine or the virtual environment, to re-evaluate the user groups.
- One last command to test if the group is well configured. Paste this command:
  `docker run hello-world`
- If the previous “hello from docker” message is printed, everything is perfectly set.

Post installation test (Phase 2):

Conda:

- Download and run this script if you have installed CRISPRitz with Conda:
  `curl https://raw.githubusercontent.com/pinellolab/CRISPRitz/master/test_scripts/auto_test_crispritz_conda.sh --output auto_test_crispritz_conda.sh`
- Write this command to execute the script:
  `bash auto_test_crispritz_conda.sh`
- Wait until this confirmation message appears:
  "EVERY TEST PASSED!!! ENJOY CRISPRitz"

Docker:

- Download and run this script if you have installed CRISPRitz with Docker:
  `curl https://raw.githubusercontent.com/pinellolab/CRISPRitz/master/test_scripts/auto_test_crispritz_docker.sh --output auto_test_crispritz_docker.sh`
- Write this command to execute the script:
  `bash auto_test_crispritz_docker.sh`
- Wait until this confirmation message appears:
  "EVERY TEST PASSED!!! ENJOY CRISPRitz"

Usage (Phase 3):

Here is a brief guide to help use CRISPRitz, **if you already execute the post installation test (Phase 2), and you obtain a positive result, you have all the necessary file in the test_crispritz directory and you can skip this list of steps.**

If you did not execute the test, follow these few steps to download the necessary files to try CRISPRitz.

Download test files (ONE TIME STEP):

- The script will download the chr22 from UCSC (hg19), the correspondent VCF file from the 1000 Genome Project, a directory containing some test guides, a directory containing some PAM sequences and a directory of pre-computed genomic annotations for the hg19 genome.
- Download the script with this command:
  `curl https://raw.githubusercontent.com/pinellolab/CRISPRitz/master/test_scripts/download_test_files.sh --output download_test_files.sh`
- Write this command to execute the script:
  `bash download_test_files.sh`
- The script will download every necessary file to test the software, we download only one chromosome and one vcf file, to save time. All the examples can be run on an entire genome, if you want to use the entire hg19 genome, you only need to add chromosomes into the `hg19_ref` directory.
- Write this command to enter the test directory:
  `cd test_crispritz/`
- Now you are ready to execute the following example functions.

3.1 - CRISPRitz Add-Variant Tool

This tool is created to insert variants in a fasta genome.
Input:

- Directory containing a genome in fasta format, need to be separated into single chromosome files.
- Directory containing VCF files, need to be separated into single chromosome files (multi-sample files will be collapsed into one fake individual).

Output:

- Directory containing a duplicate of the original genome in fasta format, separated into single chromosome files with added SNPs in IUPAC notation.
- Directory containing a duplicate of the original genome in fasta format, separated into single chromosome files with added INDELs.

Example call:
`crispritz.py add-variants hg19_1000genomeproject_vcf/ hg19_ref/` (Conda)

`docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/crispritz crispritz.py add-variants hg19_1000genomeproject_vcf/ hg19_ref/` (Docker)

Detailed input:
`hg19_1000genomeproject_vcf/` is the directory containing the vcf files.
`hg19_ref/` is the directory containing the fasta files.

3.2 - CRISPRitz Index-Genome Tool

This tool is created to generate an index genome (similar to the bwa-index step). This step is time consuming (from 30 to 60 minutes) but helps to save a lot of execution time while searching with lot of guides and with the support of bulges (DNA and RNA). If do not need to search with bulges, skip this passage.

Input:

- Name of the genome to create (e.g. `hg19_ref`).
- Directory containing a genome in fasta format, need to be separated into single chromosome files.
- Text file containing the PAM (including a number of Ns equal to the guide length) and a space separated number indicating the length of the PAM sequence (e.g. Cas9 PAM is NNNNNNNNNNNNNNNNNNNNGG 3). The sequence is composed by 20 Ns and NGG, followed by number 3, representing the length of the PAM sequence.
- Number of bulges to include in the database to perform the following search (i.e. the max number bulges allowed for DNA and RNA when searching on the database)

Output:

- Directory containing an index genome in .bin format, separated into single chromosome files, containing all the candidate targets for a selected PAM, adding also characters to perform bulge search.

Example call:

`crispritz.py index-genome hg19_ref hg19_ref/ pam/pamNGG.txt -bMax 2` (Conda)

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/crispritz crispritz.py
index-genome hg19_ref hg19_ref/ pam/pamNGG.txt -bMax 2
```
(Docker)

Detailed input:

`hg19_ref` is the name of the output directory containing the index genome.
`hg19_ref/` is the directory containing the fasta files.
`pam/pamNGG.txt/` is a text file containing the PAM sequence.
`-bMax 2` is the max number of bulges to allow for following searches when creating the database (e.g if -bMax 2, all the following searches with the created index can be performed with max 2 RNA bulges and 2 DNA bulges)

3.3 - CRISPRitz Search Tool

This tool is created to search on a fasta genome or an index genome.
There are two kinds of searches permitted with CRISPRitz;
The first and simplest one uses a common fasta genome and it's developed to perform fast, on-the-fly searches with mismatches only.
The second search type, uses the before generated index genome (Phase 3.2), to perform searches with lot of guides and also with bulges support.

3.3.1 - Mismatches only search:

Input:

- Directory containing a genome in fasta format, need to be separated into single chromosome files.
- Text file containing the PAM sequence (including a number of Ns equal to the guide length) and a space separated number indicating the length of the PAM sequence (e.g. Cas9 PAM is NNNNNNNNNNNNNNNNNNNNGG 3). The sequence is composed by 20 Ns and NGG, followed by 3, representing the length of the PAM sequence.
- Text file containing one or more guides (including a number of Ns equal to the length of the PAM sequence) (e.g. TCACCCAGGCTGGAATACAGNNN, the last 3 Ns represents the space occupied by the PAM in the real sequence)
- Name of the output file (e.g. `emx1.hg19`)
- Number of allowed mismatches (e.g. `-mm 4`)
- Output type (-r off-targets list only, -p profile only, -t everything) (e.g `-t`)
- Scores (-scores followed by the directory of the fasta genome, to perform the score after the search with score calculation based on Doench 2016 and CFD, the two scoring methods supports only NGG PAM and 23 long guides) (e.g `-scores hg19_ref/`)

Output:

- Set of result files, including:
    - Targets file, containing all genomic targets for the guide set
    - Profile file, containing a matrix-like representation of guides behaviour (bp/mm, total on-/off- target, targets per mismatch threshold)
    - Extended profile file, containing the motif matrix for each guide and each mismatch threshold, useful to create visual analysis of the guides behaviour

Example call:

```
crispritz.py search hg19_ref/ pam/pamNGG.txt guides/EMX1.txt emx1.hg19 -mm 4 -t -scores hg19_ref/
```
(Conda)

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/crispritz crispritz.py search hg19_ref/ pam/pamNGG.txt guides/EMX1.txt emx1.hg19 -mm 4 -t -scores hg19_ref/
```
(Docker)

Detailed input:
`hg19_ref/` is the directory containing the fasta files.
`pam/pamNGG.txt/` is a text file containing the PAM sequence.
`guides/EMX1.txt` is a text file containing the EXM1 guide
`emx1.hg19` is the output file name
`-mm 4` to select the mismatch threshold
`-t` to select the output type
`-scores hg19_ref/` to activate the calculation of score (Doench 2016 and CFD)

3.3.2 - Mismatches + Bulges search:
Input:

- Directory containing an index genome in .bin format, separated into single chromosome files (Phase 3.2).
- Text file containing the PAM sequence (including a number of Ns equal to the guide length) and a space separated number indicating the length of the PAM sequence (e.g. Cas9 PAM is NNNNNNNNNNNNNNNNNNNNGG 3). The sequence is composed by 20 Ns and NGG, followed by 3, representing the length of the PAM sequence.
- Text file containing one or more guides (including a number of Ns equal to the length of the PAM sequence) (e.g. TCACCCAGGCTGGAATACAGNNN, the last 3 Ns represents the space occupied by the PAM in the real sequence)
- Name of output file (e.g. `emx1.hg19`)
- Tag to activate index search (`-index`)
- Number of allowed mismatches (e.g. `-mm 4`)
- Size of DNA bulges and/or RNA bulges (e.g. `-bDNA 1 -bRNA 1`)
- Output type (-r off-targets list only, -p profile only, -t everything) (e.g `-t`)
- Scores (-scores followed by the directory of the fasta genome, to perform the score after the search with score calculation based on Doench 2016 and CFD, the two scoring methods supports only NGG PAM and 23 long guides) (e.g `-scores hg19_ref/`)

Output:

- Set of result files, including:
    - Targets file, containing all genomic targets for the guides set
    - Profile file, containing a matrix-like representation of guides behaviour (bp/mm, total on-/off- target, targets per mismatch threshold)
    - Extended profile file, containing the motif matrix for each guide and each mismatch threshold, useful to create visual analysis of the guides behaviour

Example call:
`crispritz.py search genome_library/NGG_hg19_ref/ pam/pamNGG.txt guides/EMX1.txt emx1.hg19 -index -mm 4 -bDNA 1 -bRNA 1 -t -scores hg19_ref/` (Conda)

`docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/crispritz crispritz.py search genome_library/NGG_hg19_ref/ pam/pamNGG.txt guides/EMX1.txt emx1.hg19 -index -mm 4 -bDNA 1 -bRNA 1 -t -scores hg19_ref/` (Docker)

Detailed input:
`genome_library/NGG_hg19_ref/` is the directory containing the fasta files.
`pam/pamNGG.txt/` is a text file containing the PAM sequence.
`guides/EMX1.txt` is a text file containing the EXM1 guide
`emx1.hg19` is the output file name
`-index` tag to activate the index search
`-bDNA 1` DNA bulges threshold
`-bRNA 1` RNA bulges threshold
`-mm 4` Mismatches threshold
`-t` to select the output type
`-scores hg19_ref/` to activate the calculation of score (Doench 2016 and CFD)

3.4 - CRISPRitz Annotation Tool

This tool is created to perform genomic annotation on results obtained during the search phase.
Input:

- Text file containing one or more guides, the same file used during the search phase (Phase 3.3.1 / 3.3.2)
- Targets file, containing all genomic targets for the guides set (Phase 3.3.1 / 3.3.2)
- Text file containing the path to bed files with the annotations
- Name of output file

Output:

- Set of files, including:
    - Targets file with annotation (identical file as the targets file in input) with an added column containing the annotations).
    - One file per annotation (CTCF, DNase I, Promoter, Intron, Exon, etc) in a matrix-like format, counting the targets for each input guide and for each mismatch threshold.
    - One summary file, counting all the annotations per mismatch number.

Example call:

```
crispritz.py annotate-results guides/EMX1.txt emx1.hg19.targets.txt annotations_path.txt emx1.hg19.annotated
```
(Conda)

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/crispritz crispritz.py annotate-results guides/EMX1.txt emx1.hg19.targets.txt annotations_path.txt emx1.hg19.annotated
```
(Docker)

Detailed input:

`guides/EMX1.txt` is a text file containing the EMX1 guide
`emx1.hg19.targets.txt` is the text file containing targets from previous search
`chroms_bed/annotations_path.txt` is the text file containing the paths of bed files for the genomic annotations
`emx1.hg19.annotated` name of the output file

3.5 - CRISPRitz Generate-Report Tool

This tool is created to generate a visual representation of guide behaviour such as on-/off-target activity in specific genomic regions, total number of on-/off- targets in reference and variant genome and so on.

Input:

- A guide present in the analyzed set (phase 3.3.1 / 3.3.2) (e.g.`GAGTCCGAGCAGAAGAAGAANNN`)
- Number of mismatches to analyze (e.g. `-mm 4`)
- Profile files, profile and extended profile file (Phase 3.3.1 / 3.3.2)
- Count files for genomic annotations (Phase 3.4)
- Tag to activate gecko dataset comparison (e.g. `-gecko`)

Output:

- Pdf file containing the radar chart and motif logo for a guide, the radar chart shows how much the guide is similar, in terms of number of targets found, to all guides in its dataset (or the gecko dataset if selected).
- Barplot with a distribution of on-/off- targets in each annotation and a comparison between variant and reference genome, in terms of total targets found.

Example call:

```
crispritz.py generate-report GAGTCCGAGCAGAAGAAGAANNN -mm 4 -profile
emx1.hg19.profile.xls -extprofile emx1.hg19.extended_profile.xls -exons
emx1.hg19.annotated.ExonsCount.txt -introns
emx1.hg19.annotated.IntronsCount.txt -dnase emx1.hg19.annotated.DNAseCount.txt
-ctcf emx1.hg19.annotated.CTCFCount.txt -promoters
emx1.hg19.annotated.PromotersCount.txt -gecko
```
(Conda)

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/crispritz crispritz.py
generate-report GAGTCCGAGCAGAAGAAGAANNN -mm 4 -profile emx1.hg19.profile.xls
-extprofile emx1.hg19.extended_profile.xls -exons
emx1.hg19.annotated.ExonsCount.txt -introns
emx1.hg19.annotated.IntronsCount.txt -dnase emx1.hg19.annotated.DNAseCount.txt
-ctcf emx1.hg19.annotated.CTCFCount.txt -promoters
emx1.hg19.annotated.PromotersCount.txt -gecko
```
(Docker)

Detailed input:

`GAGTCCGAGCAGAAGAAGAANNN` is a guide present in the result file, the one you want to analyze and print visualization files

`-mm 4` Mismatches threshold

`-profile emx1.hg19.profile.xls` is the xls file containing information for analyzed guides, as off-target per threshold count, bp/mm and so on

`-extprofile emx1.hg19.extended_profile.xls` is the xls file containing information detailed information about guides, used to construct the motif logo

`-exons emx1.hg19.annotated.ExonsCount.txt`
`-introns emx1.hg19.annotated.IntronsCount.txt`
`-dnase emx1.hg19.annotated.DNAseCount.txt`
`-ctcf emx1.hg19.annotated.CTCFCount.txt`

`-promoters emx1.hg19.annotated.PromotersCount.txt`
Set of files containing all the count for each genomic annotation, every file is a matrix-like text file with row representing the guides and column representing a mismatch threshold
`-gecko` tag to activate the gecko dataset comparison, the results of your test guide, will be compared with results from a previous computed analysis on gecko library.

Output examples:

- Targets file, containing all genomic targets for the guides set

65	CCATCATCTATGCCTTTGTCNNN	chr21	43291470	CCAcCAcCTcTGCCTcTGcCTGG	+	5
66	GTAGAGCGGAGGCAGGAGGCNNN	chr21	43304819	GcAGtGtGGgGaCAGGAGGCTGG	+	5
67	TCTCTGTCACCTGCATAGCTNNN	chr21	43332058	TCTCTGcCACCcaaAcAGCTGGG	+	5
68	AAGGAAAAACAGGTCAGAGANNN	chr21	43339677	AAGGAAAgAggaGTgAGAGAAGG	+	5
69	TATTCAGGCCAAAGAATTCCNNN	chr21	43346722	acTTggGGCCAAAaAATTCCAGG	+	5
70	GAGAAAATAAACAATCATGANNN	chr21	43377040	GgGAAAATAAggAATgATGcAGG	+	5
71	GCTGCCGCCCAGTGGGACTTNNN	chr21	43378200	GCTtCCcCCCAGcaGcACTTGGG	+	5
72	AAAACAGGTCAGAGATGGCCNNN	chr21	43382233	AAAACAGGaaAagGgTGGCCAGG	+	5
73	TGGTTTTGTGGGCAACATGCNNN	chr21	43382833	TGGaTTTtgGGGgAACATGaAGG	+	5
74	AAAGCCAGGACGGTCACCTTNNN	chr21	43394360	AAAGCCAGccCGGTCcCCTgGGG	+	4
75	TCTCTGTCACCTGCATAGCTNNN	chr21	43398579	TCTCaGcCtCCTGagTAGCTGGG	+	5
76	TGGTTTTGTGGGCAACATGCNNN	chr21	43415876	TGGTTTTGTGGGCcAggccCAGG	+	5
77	GGTGACAAGTGTGATCACTTNNN	chr21	43416479	GGaGtCAAGgGaGATCAtTTTGG	+	5
78	TCTCTGTCACCTGCATAGCTNNN	chr21	43424221	TCTCaGcCtCCTGagTAGCTGGG	+	5
79	CTGTGTTTGCTTTAAAAGCCNNN	chr21	43436931	tTGTtTTTGCTTTtAAAtCtTGG	+	5
80	GCAGCATAGTGAGCCCAGAANNN	chr21	43448974	GCAGCATAGgGAGgCtgGAgAGG	+	5
81	GTATGGAAAATGAGAGCTGCNNN	chr21	43462115	GTgTGGAggATGAGAGtTGtTGG	+	5
82	CCCATCATCTATGCCTTTGTNNN	chr21	43469674	CtCATCATCTATGaaggTGTGGG	+	5

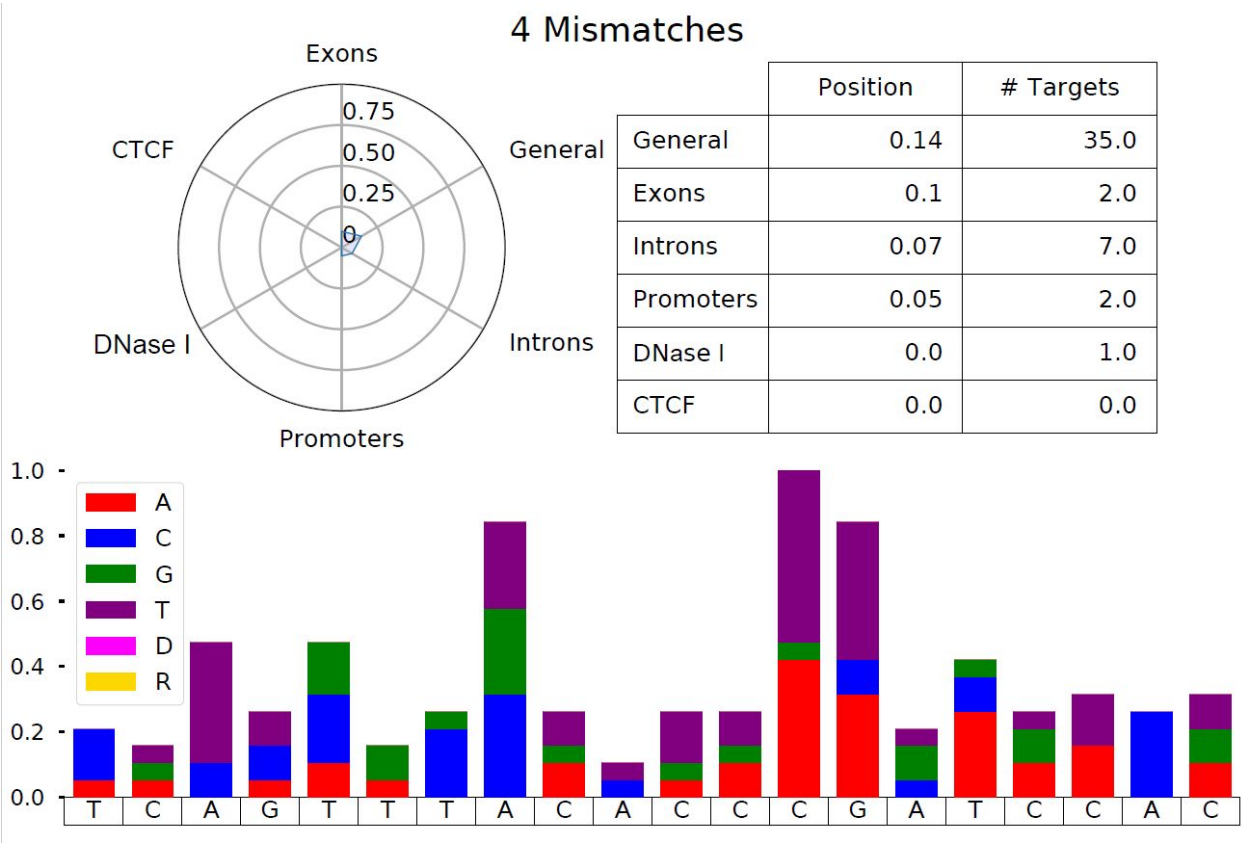
- Profile file, containing a matrix-like representation of guides behaviour (bp/mm, total on-/off- target, targets per mismatch threshold)

GUIDE	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	ONT	OFFT	OFFT/MM
AAAACAGGTCAGAGATGGCCNNN	836	798	649	1016	1712	508	976	1707	1252	1309	442	806	1494	918	1611	1087	1260	669	771	737		1	4255	4.831.492
AAAGATAGTCATCTTGGGGCNNN	352	332	244	708	335	763	431	615	593	625	393	526	681	408	400	464	369	294	577	458		1	1956	4.891.616
AAAGCCAGGACGGTCACCTTNNN	1346	333	177	165	1502	138	121	148	192	818	1760	380	1232	245	227	178	200	326	185	1501		2	2265	4.933.333
AAATGAGAAGAAGAGGCACANNN	1437	1472	1290	3087	1879	1207	1676	1123	1245	2111	1610	1308	1908	976	1921	2007	2997	1133	3147	1411		1	7191	4.859.547
AACACCAGTGAGTAGAGCGGNNN	252	306	375	202	454	180	147	175	378	177	166	185	453	210	152	187	273	598	437	190		1	1122	4.899.287
AACACCTTCCAGGAATTCTTNNN	408	398	599	370	560	479	262	342	435	412	275	386	388	390	446	402	358	375	342	468		2	1657	4.885.335
AAGACCTTCTTTTTGAGATCNNN	494	429	1120	679	1041	1067	394	400	1246	475	522	372	525	546	832	383	782	510	696	1225		1	2806	4.895.937
AAGCCAGGACGGTCACCTTTNNN	200	137	141	253	115	92	129	146	165	395	297	229	199	168	128	156	207	132	178	243		2	758	4.894.459
AAGGAAAAACAGGTCAGAGANNN	1473	1178	2122	2170	1217	1279	1339	1648	1652	3235	873	1756	1960	3243	2387	987	1527	1689	2064	1305		1	7223	4.860.030
AAGTGTGATCACTTGGGTGGNNN	433	345	455	375	382	446	400	438	499	448	387	447	452	387	297	368	388	355	334	453		1	1652	4.896.489
ACAATGTGTCAACTCTTGACNNN	236	371	177	190	190	270	160	256	258	317	188	291	317	189	305	230	204	275	272	324		1	1024	4.902.344
ACACAGCATGGACGACAGCCNNN	507	397	305	306	206	327	282	413	491	243	426	380	835	733	474	394	211	251	335	392		1	1621	4.878.470
ACAGCATTTGCAGAAGCGTTNNN	275	306	142	247	308	233	306	378	175	333	311	180	338	235	338	292	595	745	404	324		1	1322	4.890.318
ACATTAAAGATAGTCATCTTNNN	477	810	493	483	522	526	356	346	924	457	709	446	840	525	727	423	473	784	409	458		1	2294	4.877.071
AGAAGAGGCACAGGGCTGTGNNN	1122	755	1427	1196	882	1018	963	983	1330	1188	1317	871	715	863	971	1223	1173	996	1364	673		1	4326	4.861.304
AGAATTCCTGGAAGGTGTTCNNN	407	434	450	511	530	421	521	418	333	295	395	431	327	447	533	457	570	528	484	601		2	1864	4.878.219
AGATATTTCCTGCTCCCCAGNNN	355	590	521	532	537	444	359	383	369	391	314	900	875	383	562	643	549	311	392	749		1	2115	4.803.310
AGATGACTATCTTTAATGTCNNN	264	505	416	267	456	358	663	348	476	431	578	260	323	358	381	403	307	627	305	474		1	1684	4.869.359
AGCCAGGACGGTCACCTTTGNNN	197	184	236	146	117	153	184	199	462	377	227	274	229	220	245	197	177	287	206	165		2	916	4.893.013
AGCGTTTGGCAATGTGCTTTNNN	229	227	492	543	222	178	155	265	227	309	217	285	241	245	141	245	277	145	194	223		1	1034	4.893.617
AGGAGAAGGACAATGTTGTANNN	420	562	599	374	646	429	351	603	669	453	1096	412	616	758	606	756	729	757	975	683		1	2556	4.888.106
AGTAGAGCGGAGGCAGGAGGNNN	1500	902	3718	1734	969	1572	891	3717	1990	873	1260	866	781	2479	1191	734	685	1013	884	1070		1	5922	4.868.119
AGTTTACACCCGATCCACTGNNN	60	169	104	99	135	201	92	75	149	113	246	230	170	175	141	72	102	77	68	73		1	518	4.924.710
ATAATTGCAGTAGCTCTAACNNN	235	256	279	237	220	212	387	298	118	347	285	247	390	321	179	388	217	289	257	397		1	1136	4.893.486
ATATCTGTGGGCTTGTGACANNN	438	737	572	360	927	270	393	528	652	544	496	884	400	284	449	426	793	439	560	290		1	2138	4.884.004
ATCTGGTAAAGATGATTCCTNNN	746	467	861	265	420	510	589	713	337	320	673	739	402	802	407	369	317	546	361	330		1	2093	4.860.965
ATGAACACCAGTGAGTAGAGNNN	547	457	453	427	357	676	372	739	471	277	369	470	331	340	520	864	382	420	332	425		1	1889	4.885.654
ATGCAGGTGACAGAGACTCTNNN	680	635	459	615	373	465	483	664	605	599	481	303	500	466	457	657	575	627	588	614		1	2220	4.885.586
ATTTCCAAAGTCCCACTGGGNNN	370	300	287	463	340	266	256	340	269	595	524	506	384	256	535	384	292	288	399	439		1	1539	4.868.746
CAATGTGTCAACTCTTGACANNN	390	211	300	214	339	166	353	296	370	226	359	425	223	411	242	266	386	324	339	176		1	1232	4.883.117
CACACTTGTCACCACCCCAANNN	377	381	321	428	394	463	261	437	399	343	373	342	225	248	410	297	340	328	343	490		2	1473	4.887.984
CACCCCAAAGGTGACCGTCCNNN	141	145	164	133	163	114	146	197	145	162	193	178	171	222	169	278	336	186	160	132		2	720	4.909.722
CACCCGATCCACTGGGAGCNNN	214	266	266	225	495	540	427	447	271	136	323	165	250	188	191	138	265	173	218	304		1	1126	4.886.323
CAGAATTGATACTGACTGTANNN	336	163	290	217	270	359	264	355	290	415	328	392	199	330	281	385	202	334	413	313		1	1258	4.877.583
CAGAGATGGCCAGGTTGAGCNNN	501	382	495	1021	574	731	1137	326	757	750	539	382	582	607	1416	903	502	684	635	1270		2	2917	4.832.823
CAGATGACCATGACAAGCAGNNN	623	246	507	650	523	351	484	655	617	505	530	514	519	666	325	466	396	477	254	544		1	2017	4.884.482
CAGCATAGTGAGCCCAGAAGNNN	442	251	327	546	370	825	465	423	572	344	341	373	583	481	318	278	218	419	420	479		1	1740	4.870.690
CAGGACGGTCACCTTTGGGGNNN	248	123	152	196	290	639	390	285	314	263	270	262	240	212	420	324	150	204	286	374		2	1150	4.906.087
CAGGAGAAGGACAATGTTGTNNN	910	471	490	525	370	592	525	515	588	645	486	1034	492	726	850	550	881	676	672	991		1	2664	4.875.751

- Extended profile file, containing the motif matrix for each guide and each mismatch threshold, useful to create visual analysis of the guides behaviour

69	>AAAGATAGTCATCTTGGGGCNNN																					
70		BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	BP	TARGETS
71	0 MISMATCHES	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
72	NUCLEOTIDE A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
73	NUCLEOTIDE C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
74	NUCLEOTIDE G	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
75	NUCLEOTIDE T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
76																						
77	1 MISMATCHES	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
78	NUCLEOTIDE A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
79	NUCLEOTIDE C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
80	NUCLEOTIDE G	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
81	NUCLEOTIDE T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
82																						
83	2 MISMATCHES	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
84	NUCLEOTIDE A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
85	NUCLEOTIDE C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
86	NUCLEOTIDE G	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
87	NUCLEOTIDE T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
88																						
89	3 MISMATCHES	2	2	1	7	1	3	1	4	4	3	1	1	2	1	1	3	3	1	1	3	15
90	NUCLEOTIDE A	0	0	0	2	0	1	0	3	1	1	0	1	1	0	0	0	1	0	1	1	
91	NUCLEOTIDE C	0	1	0	0	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0	
92	NUCLEOTIDE G	0	0	0	0	0	1	1	0	2	0	0	0	1	1	1	0	0	0	0	2	
93	NUCLEOTIDE T	2	1	1	5	0	0	0	0	0	2	1	0	0	0	0	2	2	1	0	0	
94																						
95	4 MISMATCHES	24	24	10	60	23	53	37	61	41	49	28	39	50	25	30	34	31	28	43	38	182
96	NUCLEOTIDE A	0	0	0	32	0	31	0	31	22	17	0	15	10	9	9	14	14	19	5	9	
97	NUCLEOTIDE C	7	6	1	12	4	12	9	16	8	0	7	7	0	11	5	3	4	3	32	0	
98	NUCLEOTIDE G	6	12	4	0	9	10	16	0	11	18	4	17	17	5	16	0	0	0	0	11	
99	NUCLEOTIDE T	11	6	5	16	10	0	12	14	0	14	17	0	23	0	0	17	13	6	6	18	
100																						
101	5 MISMATCHES	326	306	233	641	311	707	393	550	548	573	364	486	629	382	369	427	335	265	533	417	1759
102	NUCLEOTIDE A	0	0	0	382	0	361	0	247	235	211	0	189	175	159	109	133	187	141	116	162	
103	NUCLEOTIDE C	106	90	32	104	73	190	60	112	135	0	122	123	0	160	53	66	40	23	302	0	
104	NUCLEOTIDE G	110	129	98	0	116	156	167	0	178	179	45	174	164	63	207	0	0	0	0	123	
105	NUCLEOTIDE T	110	87	103	155	122	0	166	191	0	183	197	0	290	0	0	228	108	101	115	132	

- Pdf file containing the radar chart and motif logo for a guide, the radar chart shows how much the guide is similar, in terms of number of targets found, to all guides in its dataset (or the gecko dataset if selected).



- Barplot with a distribution of on-/off- targets in each annotation and a comparison between variant and reference genome, in terms of total targets found.