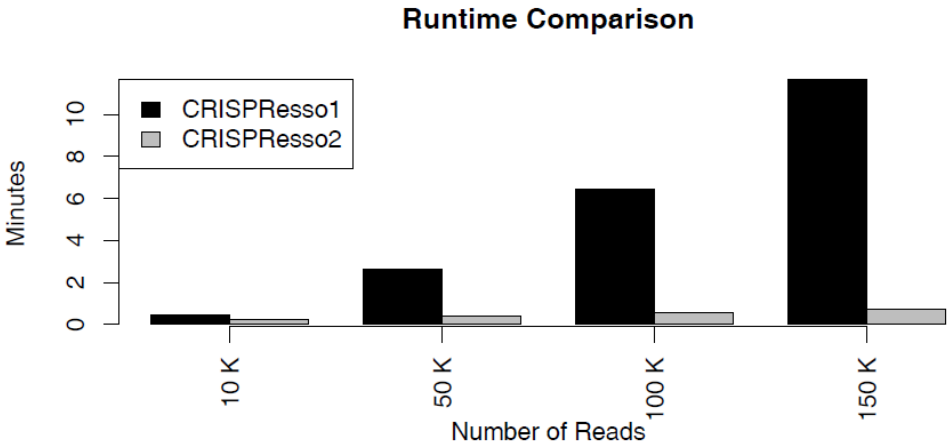
Supplementary Information



Supplementary Figure 1: Runtime comparison between CRISPResso (black) and CRISPResso2 (gray) on datasets of varying sizes.

Supplementary Note 1: Details on alignment algorithm improvements

Existing alignment algorithms find an optimal alignment of two sequences (in this case a sequenced read with the reference sequence) using a score made up of rewards for matching characters (nucleotides) and penalties for gaps. An affine gap penalty structure allows consecutive gaps to be penalized with a 'gap extension' score which is generally lower than the 'gap open' score. This tends to favor alignments with clusters of insertions or deletions reflecting the biological intuition that insertions and deletions are caused by a single nuclease cleavage event rather than multiple cleavages that create multiple single-base deletions or insertions.

For many alignments between a reference sequence and a sequencing read, one single alignment produces a highest alignment score, and that is assumed to be the correct alignment. However, particularly if the sequence is repetitive around the putative insertion or deletion, multiple alignments will share the same best score – meaning that the assignment of insertions and deletions across the reference sequence can occur at multiple locations and return the same score. Our understanding of nuclease activity suggests that most indels occur around a predicted cut site which is a physical characteristic of each nuclease. In order to select alignments with indels at the predicted cut site, we implemented a modified version of the Needleman-Wunsch (Needleman 1970) or Gotoh (Gotoh 1990) alignment that incentivizes indels at the predicted cut site while using an affine gap penalty scoring metric.

Given:

- The match/mismatch score between nucleotide at position *i* and *j* as S_{ij}
- Gap opening penalty as *GO*
- Gap extension penalty as *GE*
- The gap incentive at position *i* as GI_i

The objective is to find the optimal alignment between a read *A* and a reference sequence *B*. The matrices *I*, *J*, and *M* (each with dimensions length(*A*)xlength(*B*)) can be filled using dynamic programming. $M_{i,j}$ represents the score of the best alignment between *A[0:i]* (meaning the first *i* characters of A) and *B[0:j]* that ends with a match between A_i and B_j. $I_{i,j}$ represents the score of the best alignment between *A[0:i]* and *B[0:j]* that ends with a gap being paired with A_i. $J_{i,j}$ represents the score of the best alignment between *A[0:i]* and *B[0:j]* that ends with a gap being paired with B_j. After all three matrices are computed, a traceback procedure is used to recover the optimal alignment.

The Gotoh algorithm defines the score at each position in matrices *I*, *J*, and *M* recursively as:

$$I_{i,j} = max\begin{cases} GO + M_{i,j-1} \\ GE + I_{i,j-1} \\ GO + J_{i,j-1} \end{cases}$$

$$J_{i,j} = max\begin{cases} GO + M_{i-1,j} \\ GO + I_{i-1,j} \\ GE + J_{i-1,j} \end{cases}$$

$$M_{i,j} = max\begin{cases} M_{i-1,j-1} + S_{i,j} \\ I_{i-1,j-1} + S_{i,j} \\ J_{i-1,j-1} + S_{i,j} \end{cases}$$

Our modification to the algorithm incorporates the gap incentive vector GI, and also removes the possibility of following a gap in one sequence with a gap in the next sequence, which is the biological equivalent of observing an insertion and a deletion at the same basepair. These two modifications are reflected in our definitions below:

$$I_{i,j} = max\begin{cases} GO + M_{i,j-1} + GI_i \\ GE + I_{i,j-1} + GI_i \end{cases}$$

$$J_{i,j} = max\begin{cases} GO + M_{i-1,j} + GI_{i-1} \\ GE + J_{i-1,j} + GI_{i-1} \end{cases}$$

$$M_{i,j} = max\begin{cases} M_{i-1,j-1} + S_{i,j} \\ I_{i-1,j-1} + S_{i,j} \\ J_{i-1,j-1} + S_{i,j} \end{cases}$$

Supplementary Note 2: Comparison of alignment strategies from other amplicon-based genome editing software

Simulated data

The alignment of insertions and deletions can have an important effect on quantification of genome editing. We demonstrate this with a simulated reference and read produced by a 2bp insertion at the predicted Cas9 cut site.

Reference sequence:

GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA

Modified read sequence (2bp insertion):

GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA

Spacer sequence:

ATCAGTTGCTTCTATATCAC

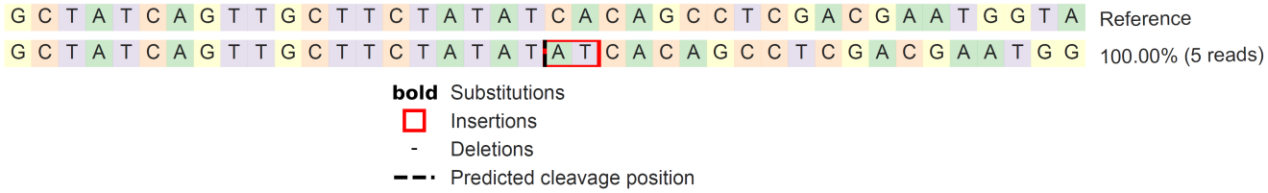
We created a simulated FASTQ file with five copies of the read sequence for testing analysis on different CRISPR analysis toolkits.

```
@M0000:100:000000000-ABCD:1:0001:0001:0001 1:N:0:CGAT
GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA
+
CCCCCCCFFFFF5GGGGGGGGGGHHHHHHHGGGGGGHHHHHHHHHHHHDGGGGEEGHGHHGGGGGGHHHHHHGHHHHHHGGGGGH
@M0000:100:000000000-ABCD:1:0001:0001:0002 1:N:0:CGAT
GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA
+
CCCCCCCFFFFF5GGGGGGGGGGHHHHHHHGGGGGGHHHHHHHHHHHHDGGGGEEGHGHHGGGGGGHHHHHHGHHHHHHGGGGGH
@M0000:100:000000000-ABCD:1:0001:0001:0003 1:N:0:CGAT
GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA
+
CCCCCCCFFFFF5GGGGGGGGGGHHHHHHHGGGGGGHHHHHHHHHHHHDGGGGEEGHGHHGGGGGGHHHHHHGHHHHHHGGGGGH
@M0000:100:000000000-ABCD:1:0001:0001:0004 1:N:0:CGAT
GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA
+
CCCCCCCFFFFF5GGGGGGGGGGHHHHHHHGGGGGGHHHHHHHHHHHHDGGGGEEGHGHHGGGGGGHHHHHHGHHHHHHGGGGGH
@M0000:100:000000000-ABCD:1:0001:0001:0005 1:N:0:CGAT
GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA
+
CCCCCCCFFFFF5GGGGGGGGGGHHHHHHHGGGGGGHHHHHHHHHHHHDGGGGEEGHGHHGGGGGGHHHHHHGHHHHHHGGGGGH
```

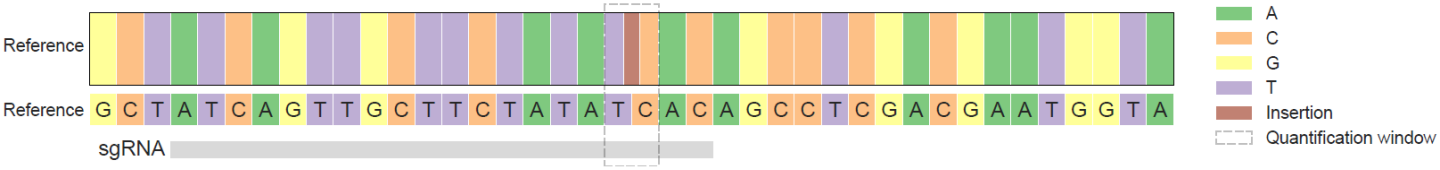
We performed analysis on this FASTQ file with the given amplicon reference and guide sequence using CRISPResso2 using the command:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso -r1 ../test.fq -a
GGGTGGGCTACAAGAGTGCAAGCTATCAGTTGCTTCTATATCACAGCCTCGACGAATGGTATGGCCTGTACCAGGGTCAATA -g
ATCAGTTGCTTCTATATCAC
```

This produced the following alignment:

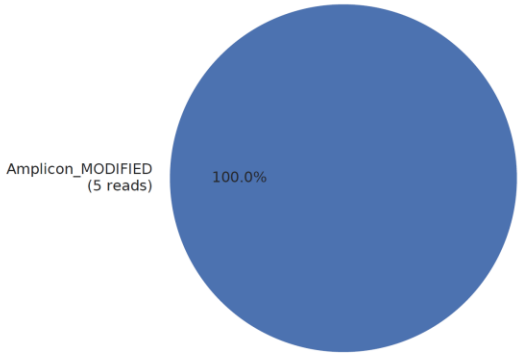


Supplementary Figure 2: CRISPResso2 output showing the location of the inserted 2bp (red) at the predicted cleavage position (dashed black line).



Supplementary Figure 3: CRISPResso2 output showing the position of the sgRNA, and the position where 100% of the reads have an insertion (brown vertical bar). The sgRNA position is shown with a gray bar. At each base, the proportion of each base (A: green, C: orange, G: yellow, and T: purple) are shown, as well as deletions (black). The proportion of reads with insertions are shown with a brown bar between two nucleotides.

Note that the insertion is correctly placed at the 3bp from the 5' end of the spacer sequence, at the predicted cleavage position, and that all reads are identified as 'modified' reads.



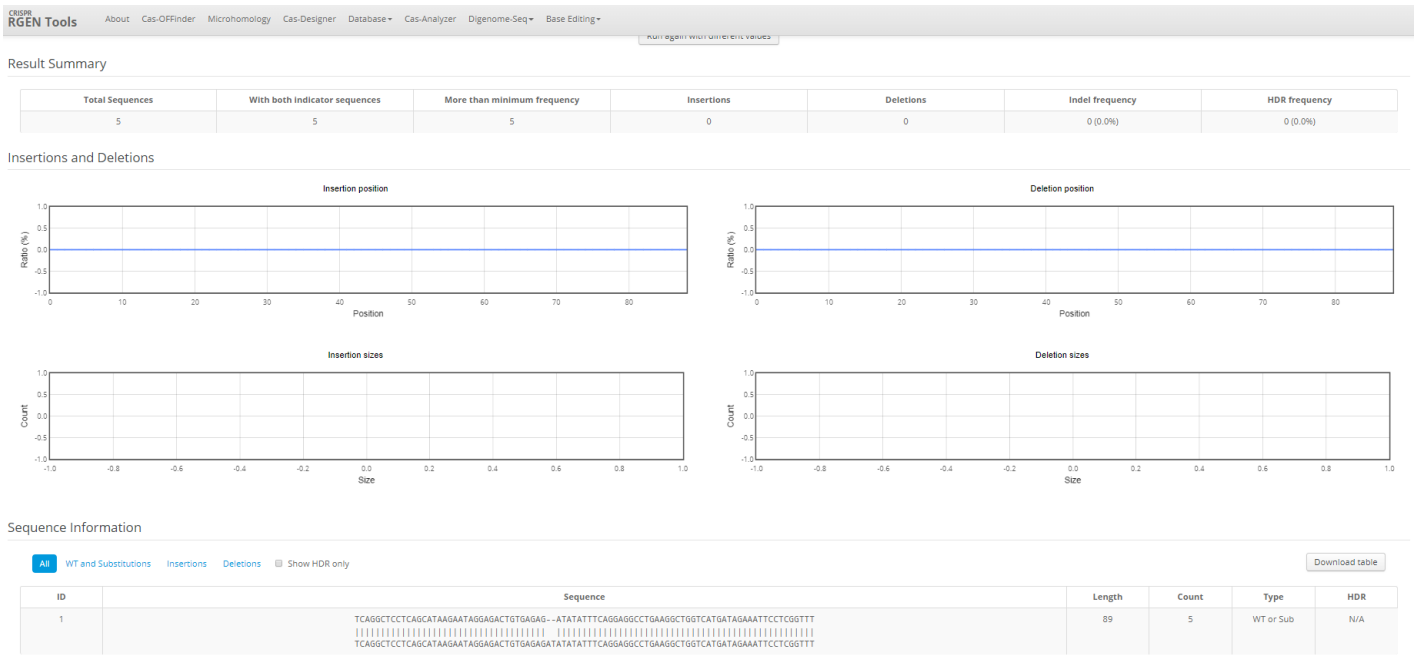
Supplementary Figure 4: CRISPResso2 quantification of edited reads identifying all 5 reads as modified.

Alignment strategies that are not aware of the predicted cutting at 3bp from the 5' end of the spacer sequence will produce alternate alignments. Below we show how currently available tools fail at correctly aligning this sequence.

Cas-Analyzer (Park 2017) aligns the indel to the left of the predicted cut site and not at the predicted cut site.

Supplementary Figure 5: annotated alignment of Cas-Analyzer output. The red arrow indicates the 2bp insertion aligned distal to the predicted cut site (black dashed line).

Cas-Analyzer excludes spurious indels that may be due to factors other than nuclease cleavage by only considering indel events that occur within a certain window (default 5bp) of the predicted cut site. Because the indel was aligned incorrectly, the indel is excluded from analysis.



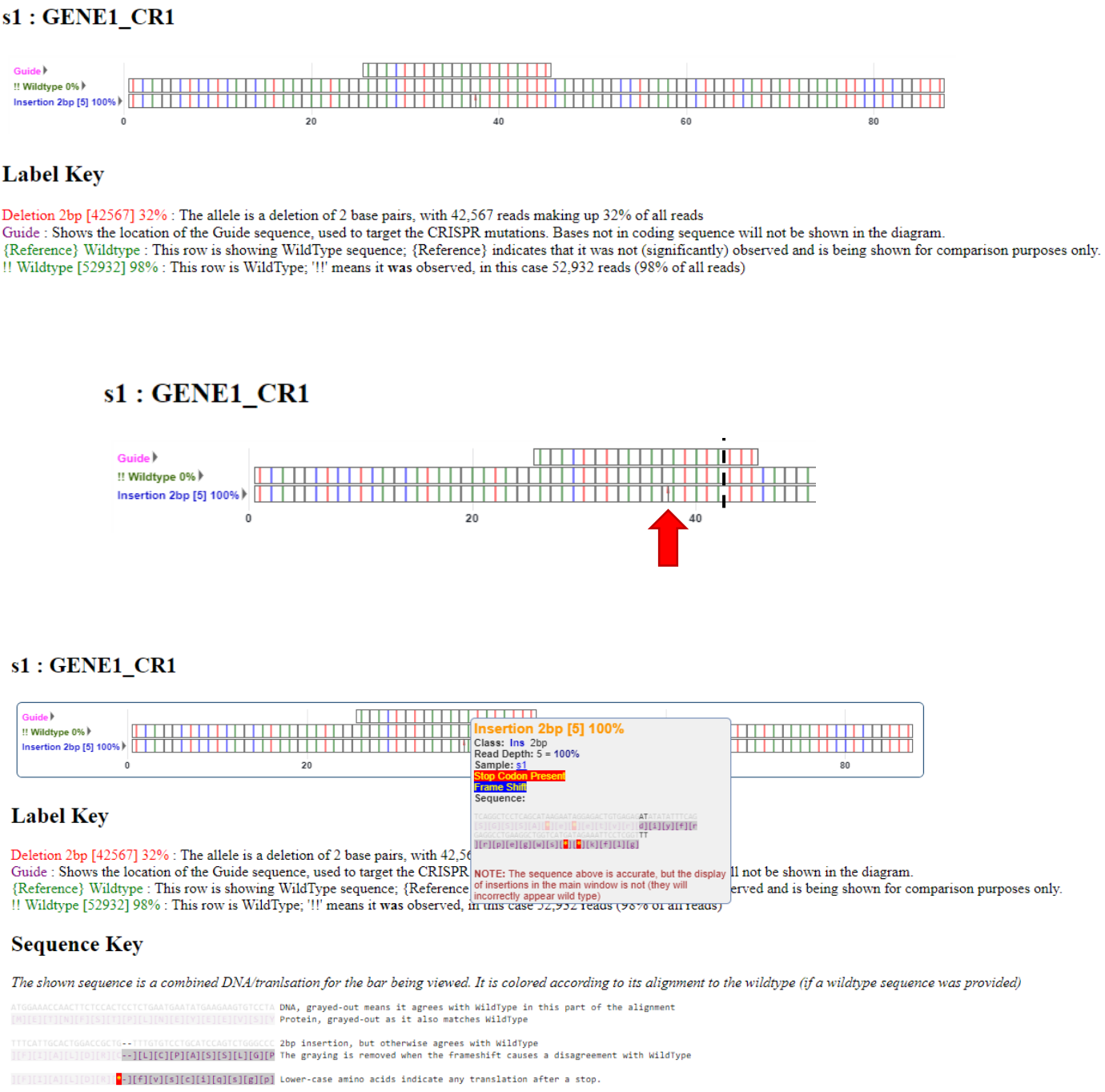
Supplementary Figure 6: output from Cas-Analyzer including the absence of identified indels (top) and the incorrect alignment (bottom).

Note that the two base pair insertion has been detected, but because it has been aligned outside of the predicted cut site, the insertion is not counted, and all reads are called unedited:

Total Sequences	With both indicator sequences	More than minimum frequency	Insertions	Deletions	Indel frequency	HDR frequency
5	5	5	0	0	0 (0.0%)	0 (0.0%)

CRISPR-DAV (Wang 2017) uses BWA (Li 2009) and ABRA (Mose 2014) to align reads to a reference. The resulting alignments also assign the insertion distal to the predicted cleavage site:

a



b

c

Supplementary Figure 7: (a) Output from CRISPR-DAV showing the position of the guide, wildtype reference, and modified reads. (b) Annotated and cropped output from CRISPR-DAV (part a) showing the location of the insertion (red arrow) and predicted cleavage site (black dashed line). (c) CRISPR-DAV output when the insertion is clicked.

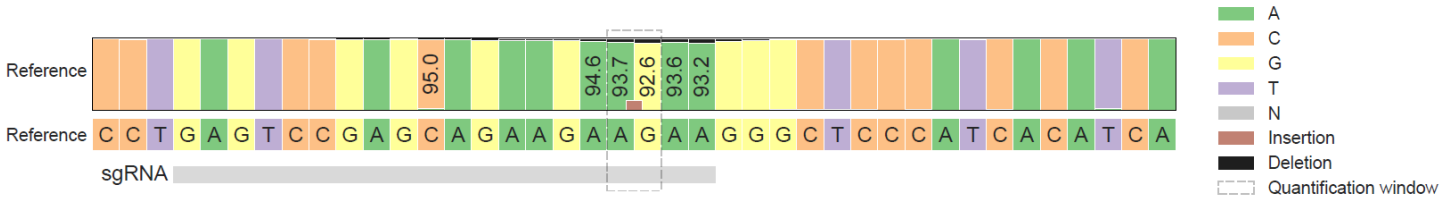
Experimental data

To demonstrate the effect of our improved alignment strategy on real sequencing data, we reanalyzed a published study (Rose 2017). Briefly, amplicon sequencing was used to profile mutations produced at the EMX1 site in HEK-293T cells 24 hours after transfection with ciCas9 and the EMX1 sgRNA. This data is accessible under the accession number SRR5694319. We compare the alignment of these reads between CRISPResso2, Cas-Analyzer (Park 2017) and CRISPR-DAV (Wang 2017) as performed with the simulated dataset above.

We ran CRISPResso2 using the following command:

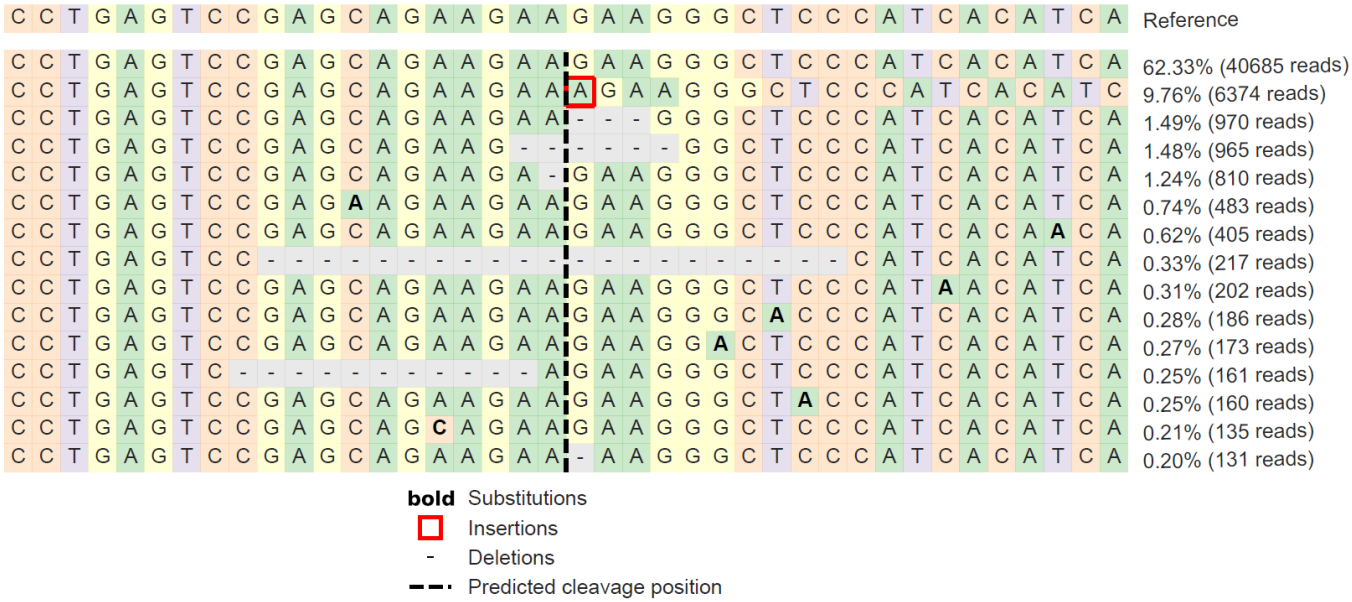
```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso -r1 SRR5694319.fastq -a
GGAGGACAAAGTACAAACGGCAGAAGCTGGAGGAGGAAGGGCCTGAGTCCGAGCAGAAGAAGAAGGGCTCCCATCACATCAACCGGTGGCGCATTGCC
ACGAAGCAGGCCAATGGGGAGGACATCGATGTCACCTCCAATGACTAGGGTG -g GAGTCCGAGCAGAAGAAGAA
```

This produced the following distribution of modifications. Note that the majority of insertions (shown in brown) and deletions (black) occur at the predicted position -3bp from the 3' end of the sgRNA.:



Supplementary Figure 8: CRISPResso2 plot of nucleotide distribution at the EMX1 locus. The sgRNA position is shown with a gray bar. At each base, the proportion of each base (A: green, C: orange, G: yellow, and T: purple) are shown, as well as deletions (black). The proportion of reads with insertions are shown with a brown bar between two nucleotides.

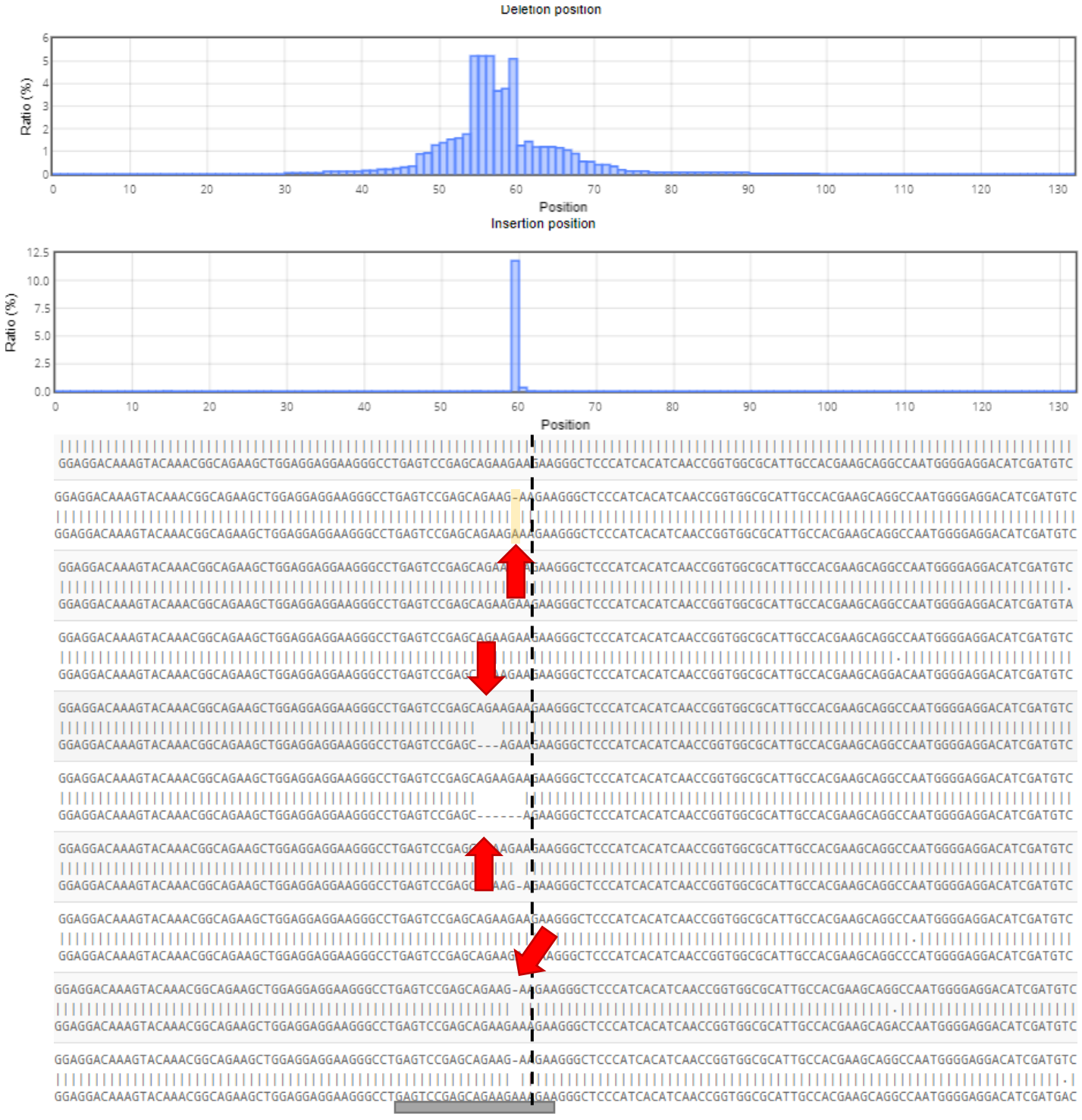
These alignments can be further visualized using CRISPResso2's alignment viewer:



Supplementary Figure 9: CRISPResso2 plot showing alignments of alleles. The reference allele is shown as the first ling, and alignments are shown in the lower portion of the plot. The predicted cut site is shown by the vertical black line. The percentage and total number of each alignment are shown to the right of each alignment.

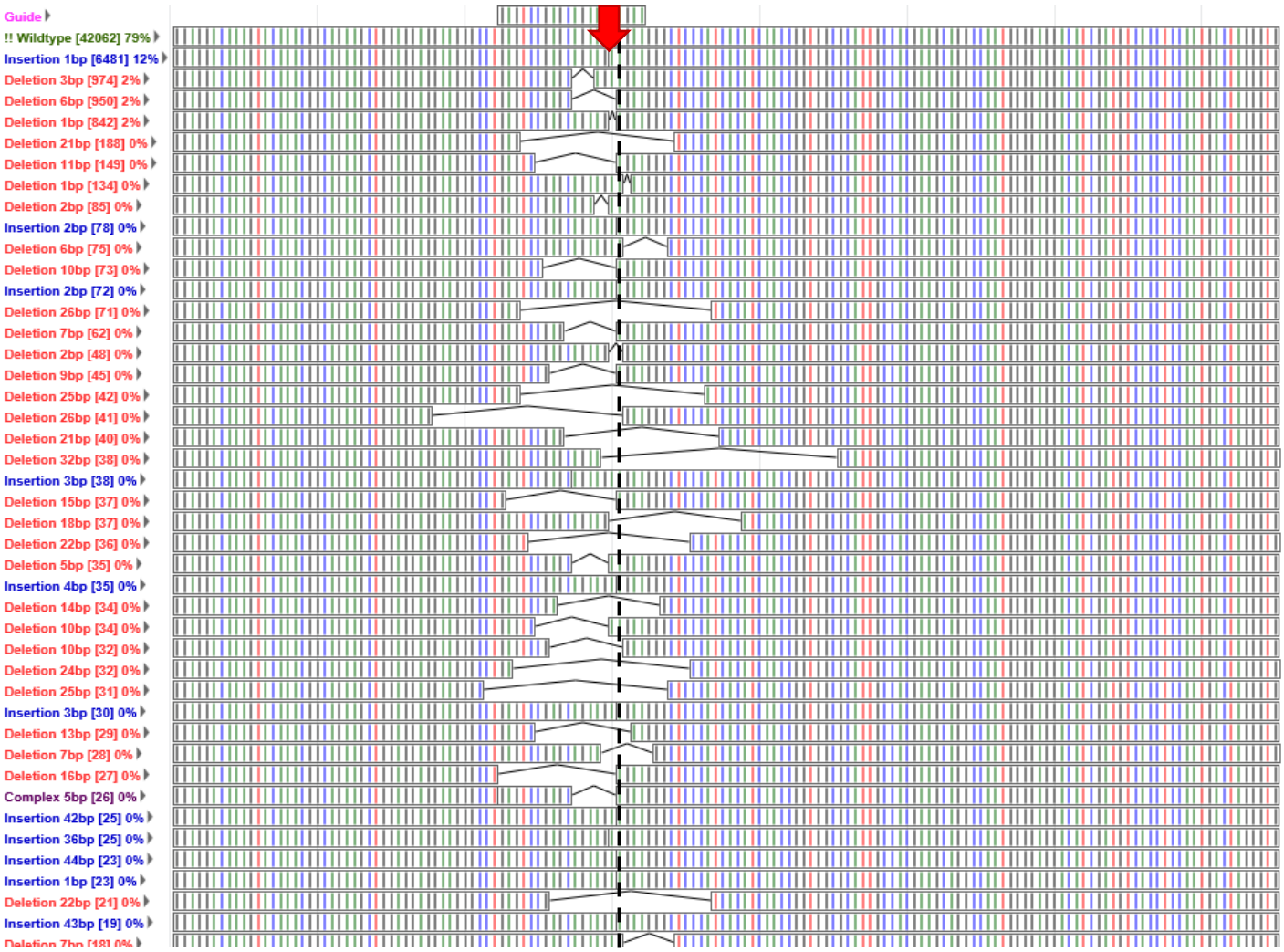
Note that the major editing event is the insertion of an A at the predicted cut site (9.76% of reads).

Cas-Analyzer (Park 2017) was used to align the same fastq file (SRR5694319) using the amplicon sequence and guide as used in the CRISPResso2 command. Cas-Analyzer aligns the dominant single-basepair insertion 2bp to the left of the predicted cut site and not at the predicted cut site. In fact, the majority of indels occur at regions distal to the cut site.



Supplementary Figure 10: annotated Cas-Analyzer alignment output. The top panel shows the position of all deletions and insertions, and the bottom panel shows the alignments reported by Cas-Analyzer. The red arrow annotations indicate insertions and deletions that are not at the predicted cut site and are aligned distal to the predicted cut site (black dashed line).The dominant insertion of the single A is highlighted with the orange box. The guide position is annotated using a gray bar.

CRISPR-DAV (Wang 2017) was used to align the same experimental fastq file to the same amplicon with the same guide. This method also aligns insertions and deletions at the left-most position, which does not overlap the cut site. Note that CRISPR-DAV does not count substitutions as modifications (even the ones that are perfectly overlapping the cleavage site and have been previously described as true repair outcomes), which leads to overquantification of wildtype allele as compared to CRISPResso2.



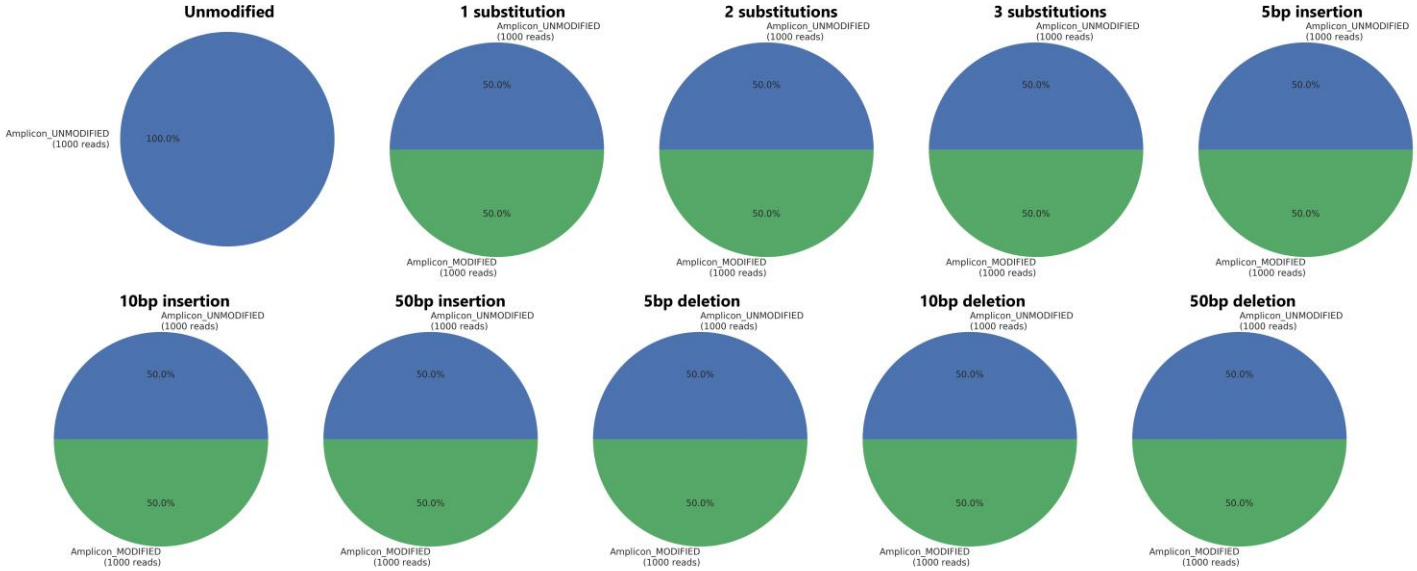
Supplementary Figure 11: Annotated CRISPR-DAV alignments. Each row represents an alignment. Insertions are shown with an additional black line. The position of the 1bp addition is annotated with a red arrow which occurs distal to the predicted cleavage position (shown with the dashed black line).

For accurate quantification of editing events, a narrow quantification window centered at the predicted cleavage position is used to avoid quantifying substitutions resulting from sequencing error. CRISPResso2 uses a 1bp window to only quantify indels and substitutions within 1bp or overlapping the predicted cleavage position. If indels are not aligned to the predicted cleavage site, these indels will be missed because they lie outside of the quantification window. On the other hand, if the quantification window is expanded to include additional bases, mutations resulting from sequencing errors will inflate the quantification of editing. Our analysis shows that for CRISPR-DAV and Cas-Analyzer ambiguous alignments involving insertions or deletions are reported at the left-most position and may not overlap the predicted cleavage site. Instead, our biologically-inspired aligner preferentially assigns insertions and deletions to positions that overlap with the predicted cleavage site. In summary, our improved alignment algorithm yields alignments that reflect our biological understanding of repair mechanisms, and improves quantification of editing events.

Supplementary Note 3: Measurement of calling accuracy by simulation studies

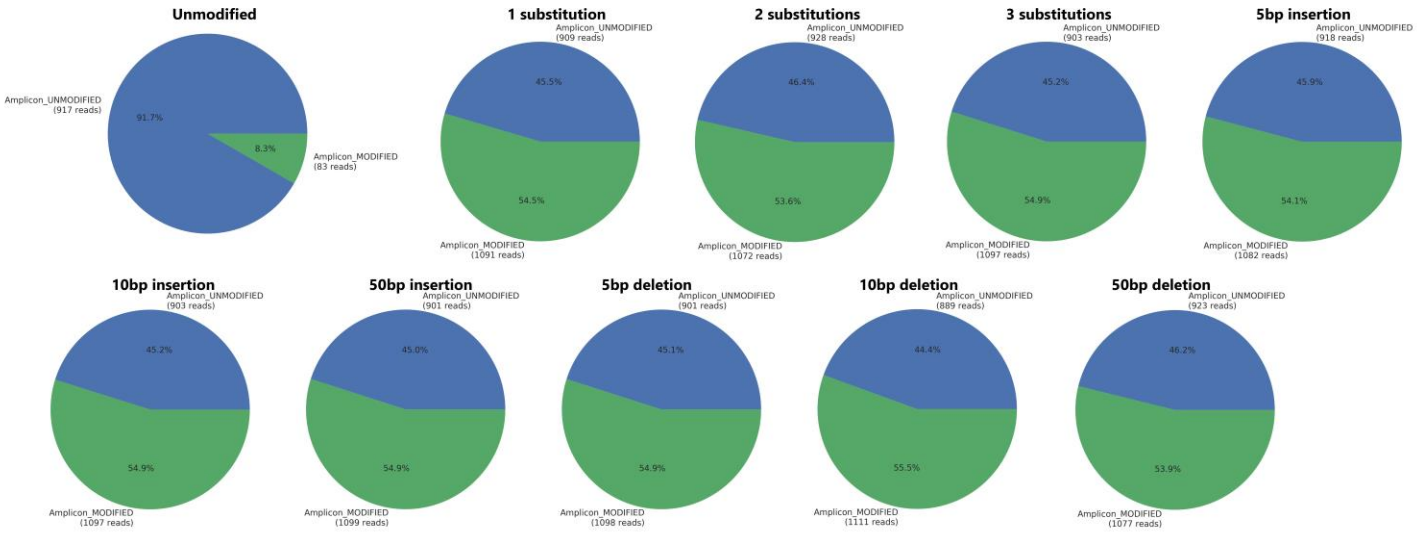
To measure the accuracy of indel calling and the robustness against sequencing errors, we simulated several datasets with known editing rates and measured the ability of CRISPResso2 to correctly recover and quantify the mutations present. This analysis was performed using the simulation methods detailed previously (Pinello 2016).

We first simulated 10 datasets with no sequencing error with 1000 unmodified reads. To 9 of the datasets, we added 1000 reads with the following modifications: substitutions (1, 2, 3bp), deletions (5, 10, 50bp), and insertions (5, 10, 50bp). We measured the number of modified reads called by CRISPResso2. As expected, CRISPResso2 recovers the simulated reads with 100% accuracy in every case.



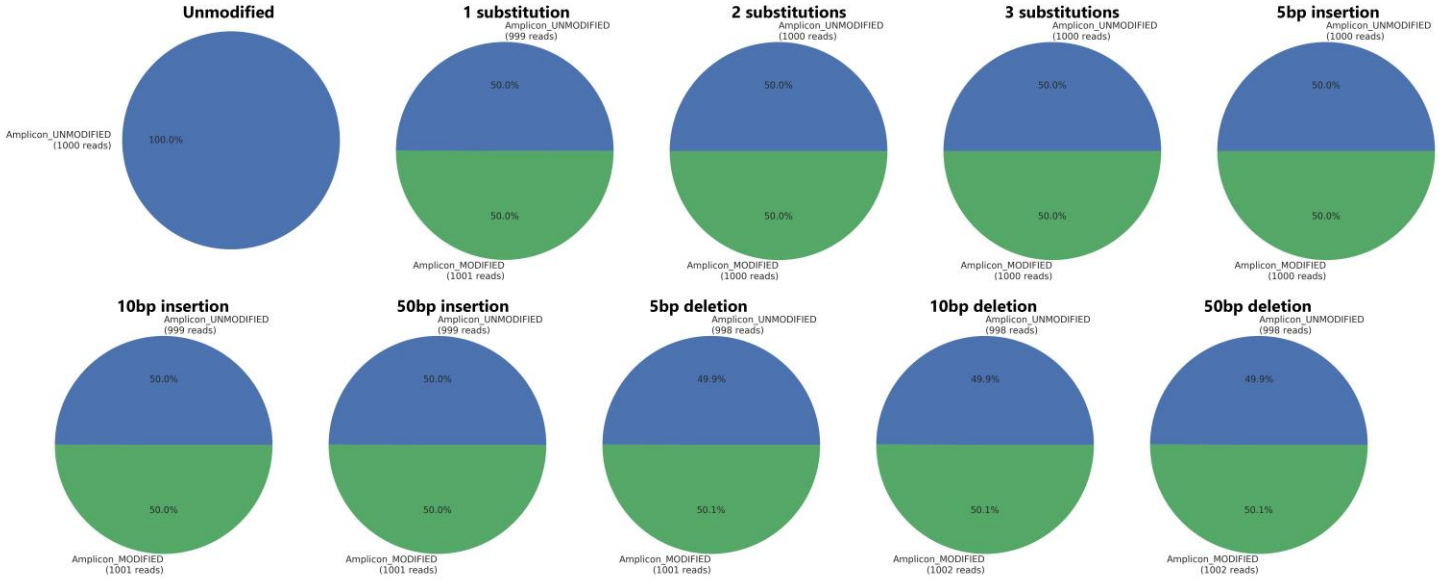
Supplementary Figure 12: CRISPResso2 calls on simulated populations without simulated sequencing error. Blue: proportion of unmodified reads, and Green: proportion of modified reads

We then performed a similar simulation, but this time introducing sequencing errors modeled after the Illumina Miseq platform using the ART tool (Huang 2012). We used CRISPResso2 to quantify the number of modified and unmodified alleles. In this case, sequencing errors in unmodified reads result in the reads being called as modified reads, and thus results in an overestimation of the number of modified reads (average 4.9% error).



Supplementary Figure 13: CRISPResso2 calls on simulated populations with sequencing errors.

We then ran CRISPResso2 on the simulated reads with simulated sequencing errors but set a 2bp window around the predicted cleavage site (1bp on each side) ignoring modifications outside of this window. This eliminates the effect of the simulated sequencing errors, and CRISPResso2 correctly identifies the percentage of modified reads as 50%.



Supplementary Figure 14: CRISPResso2 calls on simulated populations with sequencing errors using a 2bp quantification window (1bp on each side)

These three results show that by using the quantification window of CRISPResso2, accurate rates of allele modification can be recovered even in the presence of sequencing errors.

Supplementary Note 4: CRISPResso2 command line instructions and output

CRISPResso2

CRISPResso2 is a software pipeline designed to enable rapid and intuitive interpretation of genome editing experiments.

Briefly, CRISPResso2:

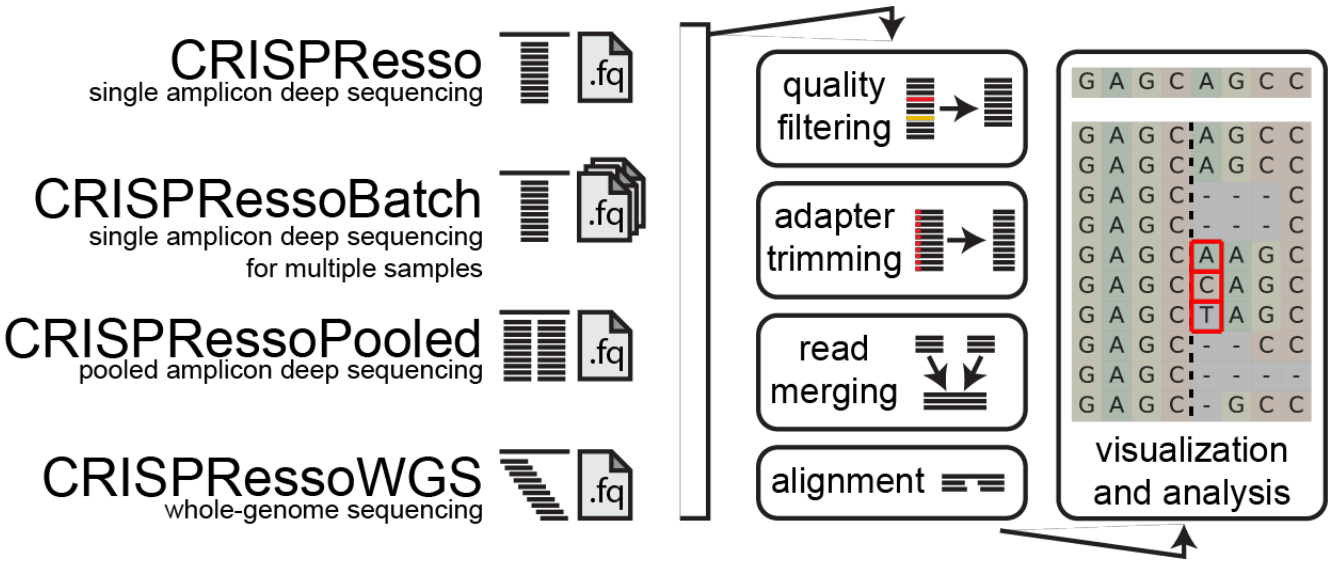
- aligns sequencing reads to a reference sequence
- quantifies insertions, mutations and deletions to determine whether a read is modified or unmodified by genome editing
- summarizes editing results in intuitive plots and datasets

What can I do with CRISPResso2?

CRISPResso2 can be used to analyze genome editing outcomes using cleaving nucleases (e.g. Cas9 or Cpf1) or noncleaving nucleases (e.g. base editors). The following operations can be automatically performed:

- filtering of low-quality reads
- adapter trimming
- alignment of reads to one or multiple reference sequences (in the case of multiple alleles)
- quantification of HDR and NHEJ outcomes (if the HDR sequence is provided)
- quantification frameshift/inframe mutations and identification affected splice sites (if an exon sequence is provided)
- visualization of the indel distribution and position (for cleaving nucleases)
- visualization of distribution and position of substitutions (for base editors)
- visualization of alleles and their frequencies

CRISPResso2 processing



Quality filtering

Input reads are first filtered based on the quality score (phred33) in order to remove potentially false positive indels. The filtering based on the phred33 quality score can be modulated by adjusting the optimal parameters (see additional notes below).

Adapter trimming

Next, adapters are trimmed from the reads. If no adapter are present, select 'No Trimming' under the 'Trimming adapter' heading in the optional parameters. If reads contain adapter sequences that need to be trimmed, select the adapters used for trimming under the 'Trimming adapter' heading in the optional parameters. Possible adapters include Nextera PE, TruSeq3 PE, TruSeq3 SE, TruSeq2 PE, and TruSeq2 SE. The adapters are trimmed from the reads using Trimmomatic.

Read merging

If paired-end reads are provided, reads are merged using FLASh . This produces a single read for alignment to the amplicon sequence, and reduces sequencing errors that may be present at the end of sequencing reads.

Alignment

The preprocessed reads are then aligned to the reference sequence with a global sequence alignment algorithm that takes into account our biological knowledge of nuclease function. If multiple alleles are present at the editing site, each allele can be passed to CRISPResso2 and sequenced reads will be assigned to the reference sequence or origin.

Visualization and analysis

Finally, after analyzing the aligned reads, a set of informative graphs are generated, allowing for the quantification and visualization of the position and type of outcomes within the amplicon sequence.

How is CRISPResso2 different from CRISPResso?

CRISPResso2 introduces four key innovations for the analysis of genome editing data:

1. Comprehensive analysis of sequencing data from base editors. We have added additional analysis and visualization capabilities especially for experiments using base editors.
2. Allele specific quantification of heterozygous references. If the targeted editing region has more than one allele, reads arising from each allele can be deconvoluted.
3. A novel biologically-informed alignment algorithm. This algorithm incorporates knowledge about the mutations produced by gene editing tools to create more biologically-likely alignments.
4. Ultra-fast processing time.

Installation

CRISPResso2 can be installed using the conda package manager Bioconda, or it can be run using the Docker containerization system.

Bioconda

To install CRISPResso2 using Bioconda, download and install Anaconda Python 2.7, following the instructions at: http://continuum.io/downloads.

Open a terminal and type:

```
conda config --add channels defaults
conda config --add channels bioconda
conda config --add channels conda-forge

conda install CRISPResso2
```

Verify that CRISPResso is installed using the command:

```
CRISPResso -h
```

Docker

CRISPResso2 can be used via the Docker containerization system. This system allows CRISPResso2 to run on your system without configuring and installing additional packages. To run CRISPResso2, first download and install docker: https://docs.docker.com/engine/installation/

Next, Docker must be configured to access your hard drive and to run with sufficient memory. These parameters can be found in the Docker settings menu. To allow Docker to access your hard drive, select 'Shared Drives' and make sure your drive name is selected. To adjust the memory allocation, select the 'Advanced' tab and allocate at least 4G of memory.

To run CRISPresso2, make sure Docker is running, then open a command prompt (Mac) or Powershell (Windows). Change directories to the location where your data is, and run the following command:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso -h
```

The first time you run this command, it will download the Docker image. The -v parameter mounts the current directory to be accessible by CRISPResso2, and the -w parameter sets the CRISPResso2 working directory. As long as you are running the command from the directory containing your data, you should not change the Docker -v or -w parameters. Additional parameters for CRISPResso2 as described below can be added to this command. For example,

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso -r1 sample.fastq.gz -a
ATTAACCAAG
```

CRISPResso2 usage

CRISPResso2 is designed be run on a single amplicon. For experiments involving multiple amplicons in the same fastq, see the instructions for CRISPRessoPooled or CRISPRessoWGS below.

CRISPresso2 requires only two parameters: input sequences in the form of fastq files (given by the --fastq_r1 and --fastq_r2) parameters, and the amplicon sequence to align to (given by the --amplicon_seq parameter). For example:

Using Bioconda:

```
CRISPResso --fastq_r1 reads.fastq.gz --amplicon_seq
AATGTCCCCCAATGGGAAGTTCATCTGGCACTGCCCACAGGTGAGGAGGTCATGATCCCCTTCTGGAGCTCCCAACGGGCCGTGGTCTGGTTCATCAT
CTGTAAGAATGGCTTCAAGAGGCTCGGCTGTGGTT
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso --fastq_r1 reads.fastq.gz
--amplicon_seq
AATGTCCCCCAATGGGAAGTTCATCTGGCACTGCCCACAGGTGAGGAGGTCATGATCCCCTTCTGGAGCTCCCAACGGGCCGTGGTCTGGTTCATCAT
CTGTAAGAATGGCTTCAAGAGGCTCGGCTGTGGTT
```

Example run: Non-homologous end joining (NHEJ)

Download the test datasets nhej.r1.fastq.gz and nhej.r2.fastq.gz to your current directory. This is the first 25,000 sequences from a paired-end sequencing experiment. To analyze this experiment, run the command:

Using Bioconda:

```
CRISPResso --fastq_r1 nhej.r1.fastq.gz --fastq_r2 nhej.r2.fastq.gz --amplicon_seq
AATGTCCCCCAATGGGAAGTTCATCTGGCACTGCCCACAGGTGAGGAGGTCATGATCCCCTTCTGGAGCTCCCAACGGGCCGTGGTCTGGTTCATCAT
CTGTAAGAATGGCTTCAAGAGGCTCGGCTGTGGTT -n nhej
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso --fastq_r1
nhej.r1.fastq.gz --fastq_r2 nhej.r2.fastq.gz --amplicon_seq
AATGTCCCCCAATGGGAAGTTCATCTGGCACTGCCCACAGGTGAGGAGGTCATGATCCCCTTCTGGAGCTCCCAACGGGCCGTGGTCTGGTTCATCAT
CTGTAAGAATGGCTTCAAGAGGCTCGGCTGTGGTT -n nhej
```

This should produce a folder called 'CRISPResso_on_nhej'. Open the file called CRISPResso_on_nhej/CRISPResso2_report.html in a web browser, and you should see an output like this: CRISPResso2_report.html.

Example run: Multiple alleles

Download the test dataset allele_specific.fastq.gz to your current directory. This is the first 25,000 sequences from a editing experiment targeting one allele. To analyze this experiment, run the following command:

Using Bioconda:

```
CRISPResso --fastq_r1 allele_specific.fastq.gz --amplicon_seq
CGAGAGCCGCAGCCATGAACGGCACAGAGGGCCCCAATTTTTATGTGCCCTTCTCCAACGTCACAGGCGTGGTGCGGAGCCACTTCGAGCAGCCGCAG
TACTACCTGGCGGAACCATGGCAGTTCTCCATGCTGGCAGCGTACATGTTCCTGCTCATCGTGCTGGG,CGAGAGCCGCAGCCATGAACGGCACAGAG
GGCCCCAATTTTTATGTGCCCTTCTCCAACGTCACAGGCGTGGTGCGGAGCCCCTTCGAGCAGCCGCAGTACTACCTGGCGGAACCATGGCAGTTCTC
CATGCTGGCAGCGTACATGTTCCTGCTCATCGTGCTGGG --amplicon_name P23H,WT --guide_seq GTGCGGAGCCACTTCGAGCAGC
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso --fastq_r1
allele_specific.fastq.gz --amplicon_seq
CGAGAGCCGCAGCCATGAACGGCACAGAGGGCCCCAATTTTTATGTGCCCTTCTCCAACGTCACAGGCGTGGTGCGGAGCCACTTCGAGCAGCCGCAG
TACTACCTGGCGGAACCATGGCAGTTCTCCATGCTGGCAGCGTACATGTTCCTGCTCATCGTGCTGGG,CGAGAGCCGCAGCCATGAACGGCACAGAG
GGCCCCAATTTTTATGTGCCCTTCTCCAACGTCACAGGCGTGGTGCGGAGCCCCTTCGAGCAGCCGCAGTACTACCTGGCGGAACCATGGCAGTTCTC
CATGCTGGCAGCGTACATGTTCCTGCTCATCGTGCTGGG --amplicon_name P23H,WT --guide_seq GTGCGGAGCCACTTCGAGCAGC
```

This should produce a folder called 'CRISPResso_on_allele_specific'. Open the file called CRISPResso_on_allele_specific/CRISPResso2_report.html in a web browser, and you should see an output like this: CRISPResso2_report.html.

Example run: Base editing experiment

Download the test dataset base_editor.fastq.gz to your current directory. This is the first 25,000 sequences from an editing experiment performed at the EMX1 locus. To analyze this experiment, run the following command:

Using Bioconda:

```
CRISPResso --fastq_r1 base_editor.fastq.gz --amplicon_seq
GGCCCCAGTGGCTGCTCTGGGGGCCTCCTGAGTTTCTCATCTGTGCCCCTCCCTCCCTGGCCCAGGTGAAGGTGTGGTTCCAGAACCGGAGGACAAAG
```

```
TACAAACGGCAGAAGCTGGAGGAGGAAGGGCCTGAGTCCGAGCAGAAGAAGAAGGGCTCCCATCACATCAACCGGTGGCGCATTGCCACGAAGCAGGC
CAATGGGGAGGACATCGATGTCACCTCCAATGACTAGGGTGG --guide_seq GAGTCCGAGCAGAAGAAGAA --
quantification_window_size 20 --quantification_window_center -10 --base_editor_output
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPResso --fastq_r1
base_editor.fastq.gz --amplicon_seq
GGCCCCAGTGGCTGCTCTGGGGGCCTCCTGAGTTTCTCATCTGTGCCCCTCCCTCCCTGGCCCAGGTGAAGGTGTGGTTCCAGAACCGGAGGACAAAG
TACAAACGGCAGAAGCTGGAGGAGGAAGGGCCTGAGTCCGAGCAGAAGAAGAAGGGCTCCCATCACATCAACCGGTGGCGCATTGCCACGAAGCAGGC
CAATGGGGAGGACATCGATGTCACCTCCAATGACTAGGGTGG --guide_seq GAGTCCGAGCAGAAGAAGAA --
quantification_window_size 20 --quantification_window_center -10 --base_editor_output
```

This should produce a folder called 'CRISPResso_on_base_editor'. Open the file called CRISPResso_on_base_editor/CRISPResso2_report.html in a web browser, and you should see an output like this: CRISPResso2_report.html.

Parameter list

-h or --help: show a help message and exit.

-a or --amplicon_seq: The amplicon sequence used for the experiment.

-r1 or --fastq_r1: The first fastq file.

-r2 or --fastq_r2 FASTQ_R2: The second fastq file for paired end reads.

-an or --amplicon_name: A name for the reference amplicon can be given. If multiple amplicons are given, multiple names can be specified here. (default: Reference)

-amas or --amplicon_min_alignment_score: Amplicon Minimum Alignment Score; score between 0 and 100; sequences must have at least this homology score with the amplicon to be aligned (can be comma-separated list of multiple scores, corresponding to amplicon sequences given in --amplicon_seq) After reads are aligned to a reference sequence, the homology is calculated as the number of bp they have in common. If the aligned read has a homology less than this parameter, it is discarded. This is useful for filtering erroneous reads that do not align to the target amplicon, for example arising from alternate primer locations. (default: 60)

--default_min_aln_score or --min_identity_score: Default minimum homology score for a read to align to a reference amplicon (default: 60)

--expand_ambiguous_alignments: If more than one reference amplicon is given, reads that align to multiple reference amplicons will count equally toward each amplicon. Default behavior is to exclude ambiguous alignments. (default: False)

-g or --guide_seq: sgRNA sequence, if more than one, please separate by commas. Note that the sgRNA needs to be input as the guide RNA sequence (usually 20 nt) immediately adjacent to but not including the PAM sequence (5' of NGG for SpCas9). If the PAM is found on the opposite strand with respect to the Amplicon Sequence, ensure the sgRNA sequence is also found on the opposite strand. The CRISPResso convention is to depict the expected cleavage position using the value of the parameter '--quantification_window_center' nucleotides from the 3' end of the guide. In addition, the use of alternate nucleases besides SpCas9 is supported. For example, if using the Cpf1 system, enter the sequence (usually 20 nt) immediately 3' of the PAM sequence and explicitly set the '--cleavage_offset' parameter to 1, since the default setting of -3 is suitable only for SpCas9. (default:)

-e or --expected_hdr_amplicon_seq: Amplicon sequence expected after HDR. The expected HDR amplicon sequence can be provided to quantify the number of reads showing a successful HDR repair. Note that the entire amplicon sequence must be provided, not just the donor template. CRISPResso2 will quantify identified instances of NHEJ, HDR, or mixed editing events. (default:)

-c or --coding_seq: Subsequence/s of the amplicon sequence covering one or more coding sequences for frameshift analysis. Sequences of exons within the amplicon sequence can be provided to enable frameshift analysis and splice site analysis by CRISPResso2. If more than one (for example, split by intron/s), please separate by commas. Users should provide the subsequences of the reference amplicon sequence that correspond to coding sequences (not the whole exon sequence(s)!). (default:)

-q or --min_average_read_quality: Minimum average quality score (phred33) to keep a read (default: 0)

-s or --min_single_bp_quality: Minimum single bp score (phred33) to keep a read (default: 0)

--min_bp_quality_or_N: Bases with a quality score (phred33) less than this value will be set to "N" (default: 0)

--file_prefix: File prefix for output plots and tables (default:)

-n or --name: Output name of the report (default: the names is obtained from the filename of the fastq file/s used in input) (default:)

-o or --output_folder: Output folder to use for the analysis (default: current folder)

--split_paired_end: Splits a single fastq file containing paired end reads in two files before running CRISPResso (default: False)

--trim_sequences: Enable the trimming of Illumina adapters with Trimmomatic (default: False)

--trimmomatic_command: Command to run Trimmomatic. Alternate executables for Trimmomatic should be specified here. The default uses the conda-installed trimmomatic. (default: trimmomatic)

--trimmomatic_options_string: Override options for Trimmomatic (default:). This parameter can be used to specify different adaptor sequences used in the experiment if you need to trim them. For example: ILLUMINACLIP:NexteraPE-PE.fa:0:90:10:0:true, where NexteraPE-PE.fa is a file containing sequences of adapters to be trimmed.
--min_paired_end_reads_overlap: Parameter for the FLASH read merging step. Minimum required overlap length between two reads to provide a confident overlap. (default: None)

--max_paired_end_reads_overlap: Parameter for the FLASH merging step. Maximum overlap length expected in approximately 90% of read pairs. Please see the FLASH manual for more information. (default: None)

-w or --quantification_window_size or --window_around_sgrna: Defines the size of the quantification window(s) centered around the position specified by the "-- cleavage_offset" or "--quantification_window_center" parameter in relation to the provided guide RNA sequence (--sgRNA). Indels overlapping this quantification window are included in classifying reads as modified or unmodified. A value of 0 disables this window and indels in the entire amplicon are considered. (default: 1)

-wc or --quantification_window_center or --cleavage_offset: Center of quantification window to use within respect to the 3' end of the provided sgRNA sequence. Remember that the sgRNA sequence must be entered without the PAM. For cleaving nucleases, this is the predicted cleavage position. The default is -3 and is suitable for the Cas9 system. For alternate nucleases, other cleavage offsets may be appropriate, for example, if using Cpf1 this parameter would be set to 1. For base editors, this could be set to -17. (default: -3)

--exclude_bp_from_left: Exclude bp from the left side of the amplicon sequence for the quantification of the indels (default: 15)

--exclude_bp_from_right: Exclude bp from the right side of the amplicon sequence for the quantification of the indels (default: 15)

--ignore_substitutions: Ignore substitutions events for the quantification and visualization (default: False)

--ignore_insertions: Ignore insertions events for the quantification and visualization (default: False)

--ignore_deletions: Ignore deletions events for the quantification and visualization (default: False)

--discard_indel_reads: Discard reads with indels in the quantification window from analysis (default: False)

--needleman_wunsch_gap_open: Gap open option for Needleman-Wunsch alignment (default: -20)

--needleman_wunsch_gap_extend: Gap extend option for Needleman-Wunsch alignment (default: -2)

--needleman_wunsch_gap_incentive: Gap incentive value for inserting indels at cut sites (default: 1)

--needleman_wunsch_aln_matrix_loc: Location of the matrix specifying substitution scores in the NCBI format (see ftp://ftp.ncbi.nih.gov/blast/matrices/) (default: EDNAFULL)

--keep_intermediate: Keep all the intermediate files (default: False)

--dump: Dump numpy arrays and pandas dataframes to file for debugging purposes (default: False)

--plot_window_size or --offset_around_cut_to_plot: Window around quantification window center to plot. Plots alleles centered at each guide. (default: 40)

--min_frequency_alleles_around_cut_to_plot: Minimum % reads required to report an allele in the alleles table plot. (default: 0.2)

--max_rows_alleles_around_cut_to_plot: Maximum number of rows to report in the alleles table plot. (default: 50)

--conversion_nuc_from: For base editor plots, this is the nucleotide targeted by the base editor (default: C)

--conversion_nuc_to: For base editor plots, this is the nucleotide produced by the base editor (default: T)

--base_editor_output: Outputs plots and tables to aid in analysis of base editor studies. If base editor output is selected, plots showing the frequency of substitutions in the quantification window are generated. The target and result bases can also be set to measure the rate of on-target conversion at bases in the quantification window. (default: False)

-qwc or --quantification_window_coordinates: Bp positions in the amplicon sequence specifying the quantification window. This parameter overrides values of the "--quantification_window_center", "-- cleavage_offset", "--window_around_sgrna" or "-- window_around_sgrna" values. Any indels outside this window are excluded. Ranges are separted by the dash sign like "start-stop", and multiple ranges can be separated by the underscore (_). A value of 0 disables this filter. (can be comma-separated list of values, corresponding to amplicon sequences given in --amplicon_seq e.g. 5-10,5-10_20-30 would specify the 5th-10th bp in the first reference and the 5th-10th and 20th-30th bp in the second reference) (default: None)

--crispresso1_mode: Parameter usage as in CRISPResso 1. In particular, if this flag is set the plot window length will be twice the value 'plot_window_size' (whereas in CRISPResso2 the window length will be plot_window_size) and the old output files 'Mapping_statistics.txt', and 'Quantification_of_editing_frequency.txt' are created, and the new files 'nucleotide_frequency_table.txt' and 'substitution_frequency_table.txt' and figure 2a and 2b are suppressed, and the files 'selected_nucleotide_percentage_table.txt' are not produced when the flag --base_editor_output is set (default: False)
--auto: Infer amplicon sequence from most common reads (default: False)

--debug: Show debug messages (default: False)

--no_rerun: Don't rerun CRISPResso2 if a run using the same parameters has already been finished. (default: False)

--suppress_report: Suppress output report (default: False)

CRISPResso2 output

The output of CRISPResso2 consists of a set of informative graphs that allow for the quantification and visualization of the position and type of outcomes within an amplicon sequence.

Data file descriptions

CRISPResso2_report.html is a summary report that can be viewed in a web browser containing all of the output plots and summary statistics.

Alleles_frequency_table.txt is a tab-separated text file that shows all reads and alignments to references. The first column shows the aligned sequence of the sequenced read. The second column shows the aligned sequence of the reference sequence. Gaps in each of these columns represent insertions and deletions. The next column 'Reference_Name' shows the name of the reference that the read aligned to. The fourth column, 'Read_Status' shows whether the read was modified or unmodified. The fifth through seventh columns ('n_deleted', 'n_inserted', 'n_substituted') show the number of bases deleted, inserted, and substituted as compared to the reference sequence. The eighth column shows the number of reads having that sequence, and the ninth column shows the percentage of all reads having that sequence.

CRISPResso_mapping_statistics.txt is a tab-delimited text file showing the number of reads in the input ('READS IN INPUTS') the number of reads after filtering, trimming and merging (READS AFTER PREPROCESSING), the number of reads aligned (READS ALIGNED) and the number of reads for which the alignment had to be computed vs read from cache.

CRISPResso_quantification_of_editing_frequency.txt is a tab-delimited text file showing the number of reads aligning to each reference amplicon, as well as the status (modified/unmodified, number of insertions, deletions, and/or substitutions) of those reads.

CRISPResso_RUNNING_LOG.txt shows a log of the CRISPResso run.

The remainder of the files are produced for each amplicon, and each file is prefixed by the name of the amplicon.

Alleles_frequency_table_around_cut_site_for_NNNNN.txt is a tab-separated text file that shows alleles and alignments to the specified reference for a subsequence around the predicted cut site (here, shown by 'NNNNN'). This data report is produced for each amplicon when a guide is found in the amplicon sequence.

quantification_window_substitution_frequency_table.txt is a tab-separated text file that shows the frequency of substitutions in the amplicon sequence in the quantification window. The first row shows the reference sequence. The following rows show the number of substitutions to each base. For example, the first numeric value in the second row (marked ‘A’) shows the number of bases that have a substitution resulting in an A at the first basepair of the amplicon sequence. The number of unmodified bases at each position is now shown in this table (because they aren’t substitutions). Thus, if the first basepair of the amplicon sequence is an A, the first value in the first row will show 0.

substitution_frequency_table.txt is a tab-separated text file that shows the frequency of substitutions in the amplicon sequence across the entire amplicon. The first row shows the reference sequence. The following rows show the number of substitutions to each base. For example, the first numeric value in the second row (marked ‘A’) shows the number of bases that have a substitution resulting in an A at the first basepair of the amplicon sequence. The number of

unmodified bases at each position is now shown in this table (because they aren't substitutions). Thus, if the first basepair of the AMPLICON sequence is an A, the first value in the first row will show 0.

insertion_histogram.txt is a tab-separated text file that shows a histogram of the insertion sizes in the amplicon sequence in the quantification window. Insertions outside of the quantification window are not included. The ins_size column shows the insertion length, and the fq column shows the number of reads having that insertion size.

deletion_histogram.txt is a tab-separated text file that shows a histogram of the deletion sizes in the amplicon sequence in the quantification window. Deletions outside of the quantification window are not included. The del_size column shows length of the deletion, and the fq column shows the number of reads having that number of substitutions.

substitution_histogram.txt is a tab-separated text file that shows a histogram of the number of substitutions in the amplicon sequence in the quantification window. Substitutions outside of the quantification window are not included. The sub_count column shows the number of substitutions, and the fq column shows the number of reads having that number of substitutions.

effect_vector_insertion.txt is a tab-separated text file with a one-row header that shows the percentage of reads with an insertion at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a insertion at that location.

effect_vector_deletion.txt is a tab-separated text file with a one-row header that shows the percentage of reads with a deletion at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a deletion at that location.

effect_vector_substitution.txt is a tab-separated text file with a one-row header that shows the percentage of reads with a substitution at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a substitution at that location.

effect_vector_combined.txt is a tab-separated text file with a one-row header that shows the percentage of reads with any modification (insertion, deletion, or substitution) at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a modification at that location.

modification_count_vectors.txt is a tab-separated file showing the number of modifications for each position in the amplicon. The first row shows the amplicon sequence, and successive rows show the number of reads with insertions (row 2), insertions_left (row 3), deletions (row 4), substitutions (row 5) and the sum of all modifications (row 6). Additionally, the last row shows the number of reads aligned.

If an insertion occurs between bases 5 and 6, the insertions vector will be incremented at bases 5 and 6. However, the insertions_left vector will only be incremented at base 5 so the sum of the insertions_left row represents an accurate count of the number of insertions, whereas the sum of the insertions row will yield twice the number of insertions.

quantification_window_modification_count_vectors.txt is a tab-separated file showing the number of modifications for positions in the quantification window of the amplicon. The first row shows the amplicon sequence in the quantification window, and successive rows show the number of reads with insertions (row 2), insertions_left (row 3), deletions (row 4), substitutions (row 5) and the sum of all modifications (row 6). Additionally, the last row shows the number of reads aligned.

nucleotide_frequency_table.txt is a tab-separated file showing the number of each residue at each position in the amplicon. The first row shows the amplicon sequence, and successive rows show the number of reads with an A (row 2), C (row 3), G (row 4), T (row 5), N (row 6), or a deletion (-) (row 7) at each position.

quantification_window_nucleotide_frequency_table.txt is a tab-separated file showing the number of each residue at positions in the quantification window of the amplicon. The first row shows the amplicon sequence in the quantification

window, and successive rows show the number of reads with an A (row 2), C (row 3), G (row 4), T (row 5), N (row 6), or a deletion (-) (row 7) at each position.

nucleotide_percentage_table.txt is a tab-separated file showing the percentage of each residue at each position in the amplicon. The first row shows the amplicon sequence, and successive rows show the percentage of reads with an A (row 2), C (row 3), G (row 4), T (row 5), N (row 6), or a deletion (-) (row 7) at each position.

quantification_window_nucleotide_percentage_table.txt is a tab-separated file showing the percentage of each residue at positions in the quantification window of the amplicon. The first row shows the amplicon sequence in the quantification window, and successive rows show the percentage of reads with an A (row 2), C (row 3), G (row 4), T (row 5), N (row 6), or a deletion (-) (row 7) at each position.

The following report files are produced when the base editor mode is enabled:

quantification_window_selected_nucleotide_percentage_table.txt is a tab-separated text file that shows the percentage of each base at selected nucleotides in the quantification window. If the base editing experiment targets cytosines (as set by the --base_editor_from parameter), each C in the quantification window will be numbered (e.g. C5 represents the cytosine at the 5th position in the reference). The percentage of each base at these selected target cytosines is reported, with the first row showing the numbered cytosines, and the remainder of the rows showing the percentage of each nucleotide present at these locations.

quantification_window_selected_nucleotide_frequency_table.txt is a tab-separated text file that shows the frequency of each base at selected nucleotides in the quantification window. If the base editing experiment targets cytosines (as set by the --base_editor_from parameter), each C in the quantification window will be numbered (e.g. C5 represents the cytosine at the 5th position in the reference). The number of reads containing each base at these selected target cytosines is reported, with the first row showing the numbered cytosines, and the remainder of the rows showing the frequency of each nucleotide present at these locations.

The following report files are produced when the amplicon contains a coding sequence:

frameshift_analysis.txt is a text file describing the number of noncoding, in-frame, and frameshift mutations. This report file is produced when the amplicon contains a coding sequence.

splice_sites_analysis.txt is a text file describing the number of splicing sites that are unmodified and modified. This file report is produced when the amplicon contains a coding sequence.

effect_vector_insertion_noncoding.txt is a tab-separated text file with a one-row header that shows the percentage of reads with a noncoding insertion at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a noncoding insertion at that location. This report file is produced when amplicon contains a coding sequence.

effect_vector_deletion_noncoding.txt is a tab-separated text file with a one-row header that shows the percentage of reads with a noncoding deletion at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a noncoding deletion at that location. This report file is produced when amplicon contains a coding sequence.

effect_vector_substitution_noncoding.txt is a tab-separated text file with a one-row header that shows the percentage of reads with a noncoding substitution at each base in the reference sequence. The first column shows the 1-based position of the amplicon, and the second column shows the percentage of reads with a noncoding substitution at that location. This report file is produced when amplicon contains a coding sequence.

Troubleshooting

Please check that your input file(s) are in FASTQ format (compressed fastq.gz also accepted).

If you get an empty report, please double check that your amplicon sequence is correct and in the correct orientation. It can be helpful to inspect the first few lines of your FASTQ file - the start of the amplicon sequence should match the start of your sequences. If not, check to see if the files are trimmed (see point below).

It is important to determine whether your reads are trimmed or not. CRISPResso2 assumes that the reads ARE ALREADY TRIMMED! If reads are not already trimmed, select the adapters used for trimming under the 'Trimming Adapter' heading under the 'Optional Parameters'. This is FUNDAMENTAL to CRISPResso analysis. Failure to trim adaptors may result in false positives. This will result in a report where you will observe an unrealistic 100% modified alleles and a sharp peak at the edges of the reference amplicon in figure 4.

The quality filter assumes that your reads uses the Phred33 scale, and it should be adjusted for each user's specific application. A reasonable value for this parameter is 30.

If your amplicon sequence is longer than your sequenced read length, the R1 and R2 reads should overlap by at least 10bp. For example, if you sequence using 150bp reads, the maximum amplicon length should be 290 bp.

Especially in repetitive regions, multiple alignments may have the best score. If you want to investigate alternate best-scoring alignments, you can view all alignments using this tool: http://rna.informatik.uni-freiburg.de/Teaching/index.jsp?toolName=Gotoh. As input, sequences from the 'Alleles_frequency_table.txt' can be used. Specifically, for a given row, the value in the 'Aligned_Sequence' should be entered into the 'Sequence a' box after removing any dashes, and the value in the 'Reference_Sequence' should be entered into the 'Sequence b' box after removing any dashes. The alternate alignments can be selected in the 'Results' panel in the Output section.

Alternate running modes

CRISPResso2 can be run for many fastqs (CRISPRessoBatch), for many amplicons in the same fastq (CRISPRessoPooled), or for whole-genome sequencing (CRISPRessoWGS).

CRISPRessoBatch

CRISPRessoBatch allows users to specify input files and other command line arguments in a single file, and then to run CRISPResso2 analysis on each file in parallel. Samples for which the amplicon and guide sequences are the same will be compared between batches, producing useful summary tables and coomparison plots.

This flexible utility adds four additional parameters:

--batch_settings: This parameter specifies the tab-separated batch file. The batch file consists of a header line listing the parameters specified, and then one line for each sample describing the parameters for that sample. Each of the parameters for CRISPResso2 given above can be specified for each sample. When CRISPRessoBatch is run, additional parameters can be specified that will be applied to all of the samples listed in the batch file. An example batch file looks like:

```
name    fastq_r1
sample1 sample1.fq
sample2 sample2.fq
sample3 sample3.fq
```

--skip_failed: If any sample fails, CRISPRessoBatch will exit without completion. However, if this parameter is specified, CRISPressoBatch will continue and only summarize the statistics of the successfully-completed runs.

-p or --n_processes: This specifies the number of processes to use for quantification. (default: 1)

-bo or --batch_output_folder: Directory where batch analysis output will be stored.

CRISPRessoBatch outputs several summary files and plots:

CRISPRessoBatch_quantification_of_editing_frequency shows the number of reads that were modified for each amplicon in each sample.

CRISPRessoBatch_mapping_statistics.txt aggregates the read mapping data from each sample.

For each amplicon, the following files are produced with the name of the amplicon as the filename prefix:

NUCLEOTIDE_FREQUENCY_SUMMARY.txt and *NUCLEOTIDE_PERCENTAGE_SUMMARY.txt* aggregate the nucleotide counts and percentages at each position in the amplicon for each sample.

MODIFICATION_FREQUENCY_SUMMARY.txt and *MODIFICATION_PERCENTAGE_SUMMARY.txt* aggregate the modification frequency and percentage at each position in the amplicon for each sample.

Example run: Batch mode

Download the test dataset files SRR3305543.fastq.gz, SRR3305544.fastq.gz, SRR3305545.fastq.gz, and SRR3305546.fastq.gz to your current directory. These are files are the first 25,000 sequences from an editing experiment performed on several base editors. Also include a batch file that lists these files and the sample names: batch.batch To analyze this experiment, run the following command:

Using Bioconda:

```
CRISPRessoBatch --batch_settings batch.batch --amplicon_seq
CATTGCAGAGAGGCGTATCATTTCGCGGATGTTCCAATCAGTACGCAGAGAGTCGCCGTCTCCAAGGTGAAAGCGGAAGTAGGGCCTTCGCGCACCTC
ATGGAATCCCTTCTGCAGCACCTGGATCGCTTTTCCGAGCTTCTGGCGGTCTCAAGCACTACCTACGTCAGCACCTGGGACCCC -p 4 --
base_edit -g GGAATCCCTTCTGCAGCACC -wc -10 -w 20
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoBatch --batch_settings
batch.batch --amplicon_seq
CATTGCAGAGAGGCGTATCATTTCGCGGATGTTCCAATCAGTACGCAGAGAGTCGCCGTCTCCAAGGTGAAAGCGGAAGTAGGGCCTTCGCGCACCTC
ATGGAATCCCTTCTGCAGCACCTGGATCGCTTTTCCGAGCTTCTGGCGGTCTCAAGCACTACCTACGTCAGCACCTGGGACCCC -p 4 --
base_edit -g GGAATCCCTTCTGCAGCACC -wc -10 -w 20
```

This should produce a folder called 'CRISPRessoBatch_on_batch'. Open the file called CRISPRessoBatch_on_batch/CRISPResso2Batch_report.html in a web browser, and you should see an output like this: CRISPResso2Batch_report.html.

CRISPRessoPooled

CRISPRessoPooled is a utility to analyze and quantify targeted sequencing CRISPR/Cas9 experiments involving sequencing libraries with pooled amplicons. One common experimental strategy is to pool multiple amplicons (e.g. a single on-target site plus a set of potential off-target sites) into a single deep sequencing reaction (briefly, genomic DNA samples for pooled applications can be prepared by first amplifying the target regions for each gene/target of interest with regions of 150-400bp depending on the desired coverage. In a second round of PCR, with minimized cycle numbers, barcode and adaptors are added. With optimization, these two rounds of PCR can be merged into a single reaction. These reactions are then quantified, normalized, pooled, and undergo quality control before being sequenced). CRISPRessoPooled demultiplexes reads from multiple amplicons and runs the CRISPResso utility with appropriate reads for each amplicon separately.

Usage

This tool can run in 3 different modes:

Amplicons mode: Given a set of amplicon sequences, in this mode the tool demultiplexes the reads, aligning each read to the amplicon with best alignment, and creates separate compressed FASTQ files, one for each amplicon. Reads that do not align to any amplicon are discarded. After this preprocessing, CRISPResso is run for each FASTQ file, and separated reports are generated, one for each amplicon.

To run the tool in this mode the user must provide:

1. Paired-end reads (two files) or single-end reads (single file) in FASTQ format (fastq.gz files are also accepted)

2. A description file containing the amplicon sequences used to enrich regions in the genome and some additional information. In particular, this file, is a tab delimited text file with up to 5 columns (first 2 columns required):

- *AMPLICON_NAME*: an identifier for the amplicon (*must be unique*).

- *AMPLICON_SEQUENCE*: amplicon sequence used in the design of the experiment.

- *sgRNA_SEQUENCE (OPTIONAL)*: sgRNA sequence used for this amplicon *without the PAM sequence.* If not available, enter *NA.*

- *EXPECTED_AMPLICON_AFTER_HDR (OPTIONAL)*: expected amplicon sequence in case of HDR. If more than one, separate by commas *and not spaces*. If not available, enter *NA.*

- *CODING_SEQUENCE (OPTIONAL)*: Subsequence(s) of the amplicon corresponding to coding sequences. If more than one, separate by commas *and not spaces*. If not available, enter *NA.*

A file in the correct format should look like this:

Site1 CACACTGTGGCCCCTGTGCCCAGCCCTGGGCTCTCTGTACATGAAGCAAC CCCTGTGCCCAGCCC NA NA

Site2 GTCCTGGTTTTTGGTTTGGGAAATATAGTCATC NA GTCCTGGTTTTTGGTTTAAAAAAATATAGTCATC NA

Site 3 TTTCTGGTTTTTGGTTTGGGAAATATAGTCATC NA NA GGAAATATA

Note: *no column titles should be entered.* Also the colors here are used only for illustrative purposes and in a plain text file will be not be present and saved.

The user can easily create this file with *any text editor* or with spreadsheet software like Excel (Microsoft), Numbers (Apple) or Sheets (Google Docs) and then save it as tab delimited file.

Example:

Using Bioconda:

```
CRISPRessoPooled -r1 SRR1046762_1.fastq.gz -r2 SRR1046762_2.fastq.gz -f AMPLICONS_FILE.txt --name
ONLY_AMPLICONS_SRR1046762 --gene_annotations gencode_v19.gz
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoPooled -r1
SRR1046762_1.fastq.gz -r2 SRR1046762_2.fastq.gz -f AMPLICONS_FILE.txt --name
ONLY_AMPLICONS_SRR1046762 --gene_annotations gencode_v19.gz
```

The output of CRISPRessoPooled Amplicons mode consists of:

1. REPORT_READS_ALIGNED_TO_AMPLICONS.txt: this file contains the same information provided in the input description file, plus some additional columns:

   a. *Demultiplexed_fastq.gz_filename*: name of the files containing the raw reads for each amplicon.

   b. *n_reads*: number of reads recovered for each amplicon.

2. A set of fastq.gz files, one for each amplicon.

3. A set of folders, one for each amplicon, containing a full CRISPResso report.

4. SAMPLES_QUANTIFICATION_SUMMARY.txt: this file contains a summary of the quantification and the alignment statistics for each region analyzed (read counts and percentages for the various classes: Unmodified, NHEJ, point mutations, and HDR).

5. *CRISPRessoPooled_RUNNING_LOG.txt*: execution log and messages for the external utilities called.

Genome mode: In this mode the tool aligns each read to the best location in the genome. Then potential amplicons are discovered looking for regions with enough reads (the default setting is to have at least 1000 reads, but the parameter can be adjusted with the option *--min_reads_to_use_region*). If a gene annotation file from UCSC is provided, the tool also reports the overlapping gene/s to the region. In this way it is possible to check if the amplified regions map to expected genomic locations and/or also to pseudogenes or other problematic regions. Finally, CRISPResso is run in each region discovered.

To run the tool in this mode the user must provide:

1. Paired-end reads (two files) or single-end reads (single file) in FASTQ format (fastq.gz files are also accepted)

2. The full path of the reference genome in bowtie2 format (e.g. /genomes/human_hg19/hg19). Instructions on how to build a custom index or precomputed index for human and mouse genome assembly can be downloaded from the bowtie2 website: http://bowtie-bio.sourceforge.net/bowtie2/index.shtml.

3. Optionally the full path of a gene annotations file from UCSC. The user can download this file from the UCSC Genome Browser (http://genome.ucsc.edu/cgi-bin/hgTables?command=start) selecting as table "knownGene", as output format "all fields from selected table" and as file returned "gzip compressed". (e.g. like: /genomes/human_hg19/gencode_v19.gz)

Example:

Using Bioconda:

```
CRISPRessoPooled -r1 SRR1046762_1.fastq.gz -r2 SRR1046762_2.fastq.gz -x /GENOMES/hg19/hg19 --name
ONLY_GENOME_SRR1046762 --gene_annotations gencode_v19.gz
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoPooled -r1
SRR1046762_1.fastq.gz -r2 SRR1046762_2.fastq.gz -x /GENOMES/hg19/hg19 --name
ONLY_GENOME_SRR1046762 --gene_annotations gencode_v19.gz
```

The output of CRISPRessoPooled Genome mode consists of:

1. REPORT_READS_ALIGNED_TO_GENOME_ONLY.txt: this file contains the list of all the regions discovered, one per line with the following information:

- chr_id: chromosome of the region in the reference genome.
- bpstart: start coordinate of the region in the reference genome.
- bpend: end coordinate of the region in the reference genome.
- fastq_file: location of the fastq.gz file containing the reads mapped to the region.
- n_reads: number of reads mapped to the region.
- sequence: the sequence, on the reference genome for the region.

1. MAPPED_REGIONS (folder): this folder contains all the fastq.gz files for the discovered regions.
2. A set of folders with the CRISPResso report on the regions with enough reads.
3. SAMPLES_QUANTIFICATION_SUMMARY.txt: this file contains a summary of the quantification and the alignment statistics for each region analyzed (read counts and percentages for the various classes: Unmodified, NHEJ, point mutations, and HDR).
4. *CRISPRessoPooled_RUNNING_LOG.txt*: execution log and messages for the external utilities called.

   This running mode is particular useful to check if there are mapping artifacts or contaminations in the library. In an optimal experiment, the list of the regions discovered should contain only the regions for which amplicons were designed.

Mixed mode (Amplicons + Genome): in this mode, the tool first aligns reads to the genome and, as in the **Genome mode**, discovers aligning regions with reads exceeding a tunable threshold. Next it will align the amplicon sequences to the reference genome and will use only the reads that match both the amplicon locations and the discovered genomic locations, excluding spurious reads coming from other regions, or reads not properly trimmed. Finally, CRISPResso is run using each of the surviving regions.

To run the tool in this mode the user must provide:

- Paired-end reads (two files) or single-end reads (single file) in FASTQ format (fastq.gz files are also accepted)
- A description file containing the amplicon sequences used to enrich regions in the genome and some additional information (as described in the Amplicons mode section).
- The reference genome in bowtie2 format (as described in Genome mode section).
- Optionally the gene annotations from UCSC (as described in Genome mode section).

Example:

Using Bioconda:

```
CRISPRessoPooled -r1 SRR1046762_1.fastq.gz -r2 SRR1046762_2.fastq.gz -f AMPLICONS_FILE.txt -x
/GENOMES/hg19/hg19 --name AMPLICONS_AND_GENOME_SRR1046762 --gene_annotations gencode_v19.gz
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoPooled -r1
SRR1046762_1.fastq.gz -r2 SRR1046762_2.fastq.gz -f AMPLICONS_FILE.txt -x /GENOMES/hg19/hg19 --name
AMPLICONS_AND_GENOME_SRR1046762 --gene_annotations gencode_v19.gz
```

The output of CRISPRessoPooled Mixed Amplicons + Genome mode consists of these files:

1. REPORT_READS_ALIGNED_TO_GENOME_AND_AMPLICONS.txt: this file contains the same information provided in the input description file, plus some additional columns:

- Amplicon_Specific_fastq.gz_filename: name of the file containing the raw reads recovered for the amplicon.
- *n_reads*: number of reads recovered for the amplicon.
- *Gene_overlapping:* gene/s overlapping the amplicon region.
- chr_id: chromosome of the amplicon in the reference genome.
- bpstart: start coordinate of the amplicon in the reference genome.
- bpend: end coordinate of the amplicon in the reference genome.
- Reference_Sequence: sequence in the reference genome for the region mapped for the amplicon.

2. MAPPED_REGIONS (folder): this folder contains all the fastq.gz files for the discovered regions.
3. A set of folders with the CRISPResso report on the amplicons with enough reads.
4. SAMPLES_QUANTIFICATION_SUMMARY.txt: this file contains a summary of the quantification and the alignment statistics for each region analyzed (read counts and percentages for the various classes: Unmodified, NHEJ, point mutations, and HDR).
5. *CRISPRessoPooled_RUNNING_LOG.txt*: execution log and messages for the external utilities called.

The Mixed mode combines the benefits of the two previous running modes. In this mode it is possible to recover in an unbiased way all the genomic regions contained in the library, and hence discover contaminations or mapping artifacts. In addition, by knowing the location of the amplicon with respect to the reference genome, reads not properly trimmed or mapped to pseudogenes or other problematic regions will be automatically discarded, providing the cleanest set of reads to quantify the mutations in the target regions with CRISPResso.

If the focus of the analysis is to obtain the best quantification of editing efficiency for a set of amplicons, we suggest running the tool in the Mixed mode. The Genome mode is instead suggested to check problematic libraries, since a report is generated for each region discovered, even if the region is not mappable to any amplicon (however, his may be time consuming). Finally, the Amplicon mode is the fastest, although the least reliable in terms of quantification accuracy.

CRISPRessoWGS

CRISPRessoWGS is a utility for the analysis of genome editing experiment from whole genome sequencing (WGS) data. CRISPRessoWGS allows exploring any region of the genome to quantify targeted editing or potentially off-target effects.

Usage

To run CRISPRessoWGS you must provide:

1. A genome aligned *BAM* file. To align reads from a WGS experiment to the genome there are many options available, we suggest using either **Bowtie2 (**http://bowtie-bio.sourceforge.net/bowtie2/) or **BWA (**http://bio-bwa.sourceforge.net/**).**

2. A *FASTA* file containing the reference sequence used to align the reads and create the BAM file (the reference files for the most common organism can be download from UCSC: http://hgdownload.soe.ucsc.edu/downloads.html. *Download and uncompress only the file ending with .fa.gz*, for example for the last version of the human genome download and *uncompress* the file hg38.fa.gz)

3. Descriptions file containing the coordinates of the regions to analyze and some additional information. In particular, this file is a tab delimited text file with up to 7 columns (4 required):

- chr_id: chromosome of the region in the reference genome.
- bpstart: start coordinate of the region in the reference genome.
- bpend: end coordinate of the region in the reference genome.
- *REGION_NAME*: an identifier for the region (*must be unique*).
- *sgRNA_SEQUENCE (OPTIONAL)*: sgRNA sequence used for this genomic segment *without the PAM sequence.* If not available, enter *NA*.
- *EXPECTED_SEGMENT_AFTER_HDR (OPTIONAL)*: expected genomic segment sequence in case of HDR. If more than one, separate by commas *and not spaces*. If not available, enter *NA*.
- *CODING_SEQUENCE (OPTIONAL)*: Subsequence(s) of the genomic segment corresponding to coding sequences. If more than one, separate by commas *and not spaces*. If not available, enter *NA.*

A file in the correct format should look like this:

```
chr1 65118211 65118261 R1 CTACAGAGCCCCAGTCCTGG NA NA

chr6 51002798 51002820 R2 NA NA NA
```

Note: *no column titles should be entered.* As you may have noticed this file is just a *BED* file with extra columns. For this reason, a normal BED file with 4 columns, is also **accepted** by this utility.

4. Optionally the full path of a gene annotations file from UCSC. You can download the this file from the UCSC Genome Browser (http://genome.ucsc.edu/cgi-bin/hgTables?command=start) selecting as table "knownGene", as output format "all fields from selected table" and as file returned "gzip compressed". (something like: /genomes/human_hg19/gencode_v19.gz)

Example:

Using Bioconda:

```
CRISPRessoWGS -b WGS/50/50_sorted_rmdup_fixed_groups.bam -f WGS_TEST.txt -r /GENOMES/mm9/mm9.fa --
gene_annotations ensemble_mm9.txt.gz --name CRISPR_WGS_SRR1542350
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoWGS -b
WGS/50/50_sorted_rmdup_fixed_groups.bam -f WGS_TEST.txt -r /GENOMES/mm9/mm9.fa --gene_annotations
ensemble_mm9.txt.gz --name CRISPR_WGS_SRR1542350
```

The output from these files will consist of:

1. REPORT_READS_ALIGNED_TO_SELECTED_REGIONS_WGS.txt: this file contains the same information provided in the input description file, plus some additional columns:

   a. sequence: sequence in the reference genome for the region specified.

   b. *gene_overlapping:* gene/s overlapping the region specified.

   c. *n_reads*: number of reads recovered for the region.

   d. bam_file_with_reads_in_region: file containing only the subset of the reads that overlap, also partially, with the region. This file is indexed and can be easily loaded for example on IGV for visualization of single reads or for the comparison of two conditions. For example, in the figure below (fig X) we show reads mapped to a region inside the coding sequence of the gene Crygc subjected to NHEJ (CRISPR_WGS_SRR1542350) vs reads from a control experiment (CONTROL_WGS_SRR1542349).

   e. fastq.gz_file_trimmed_reads_in_region: file containing only the subset of reads fully covering the specified regions, and trimmed to match the sequence in that region. These reads are used for the subsequent analysis with CRISPResso.

2. ANALYZED_REGIONS (folder): this folder contains all the BAM and FASTQ files, one for each region analyzed.

3. A set of folders with the CRISPResso report on the regions provided in input with enough reads (the default setting is to have at least 10 reads, but the parameter can be adjusted with the option

   --min_reads_to_use_region).

4. *CRISPRessoPooled_RUNNING_LOG.txt*: execution log and messages for the external utilities called.

This utility is particularly useful to investigate and quantify mutation frequency in a list of potential target or off-target sites, coming for example from prediction tools, or from other orthogonal assays.

CRISPRessoCompare

CRISPRessoCompare is a utility for the comparison of a pair of CRISPResso analyses. CRISPRessoCompare produces a summary of differences between two conditions, for example a CRISPR treated and an untreated control sample (see figure below). Informative plots are generated showing the differences in editing rates and localization within the reference amplicon,

Usage

To run CRISPRessoCompare you must provide:

1. Two output folders generated with CRISPResso using the same reference amplicon and settings but on different datasets.
2. Optionally a name for each condition to use for the plots, and the name of the output folder

Example:

Using Bioconda:

```
CRISPRessoCompare -n1 "VEGFA CRISPR" -n2 "VEGFA CONTROL" -n VEGFA_Site_1_SRR10467_VS_SRR1046787
CRISPResso_on_VEGFA_Site_1_SRR1046762/ CRISPResso_on_VEGFA_Site_1_SRR1046787/
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoCompare -n1 "VEGFA CRISPR"
-n2 "VEGFA CONTROL"  -n VEGFA_Site_1_SRR10467_VS_SRR1046787 CRISPResso_on_VEGFA_Site_1_SRR1046762/
CRISPResso_on_VEGFA_Site_1_SRR1046787/
```

The output will consist of:

1. Comparison_Efficiency.pdf: a figure containing a comparison of the edit frequencies for each category (NHEJ, MIXED NHEJ-HDR and HDR) and as well the net effect subtracting the second sample (second folder in the command line) provided in the analysis from the first sample (first folder in the command line).
2. Comparison_Combined_Insertion_Deletion_Substitution_Locations.pdf: a figure showing the average profile for the mutations for the two samples in the same scale and their difference with the same convention used in the previous figure (first sample – second sample).
3. CRISPRessoCompare_RUNNING_LOG.txt: detailed execution log.

CRISPRessoPooledWGSCompare

CRISPRessoPooledWGSCompare is an extension of the CRIPRessoCompare utility allowing the user to run and summarize multiple CRISPRessoCompare analyses where several regions are analyzed in two different conditions, as in the case of the CRISPRessoPooled or CRISPRessoWGS utilities.

Usage

To run CRISPRessoPooledWGSCompare you must provide:

1. Two output folders generated with CRISPRessoPooled or CRISPRessoWGS using the same reference amplicon and settings but on different datasets.
2. Optionally a name for each condition to use for the plots, and the name of the output folder

Example:

Using Bioconda:

```
CRISPRessoPooledWGSCompare CRISPRessoPooled_on_AMPLICONS_AND_GENOME_SRR1046762/
CRISPRessoPooled_on_AMPLICONS_AND_GENOME_SRR1046787/ -n1 SRR1046762 -n2 SRR1046787 -n
AMPLICONS_AND_GENOME_SRR1046762_VS_SRR1046787
```

Using Docker:

```
docker run -v ${PWD}:/DATA -w /DATA -i pinellolab/CRISPResso2 CRISPRessoPooledWGSCompare
CRISPRessoPooled_on_AMPLICONS_AND_GENOME_SRR1046762/
CRISPRessoPooled_on_AMPLICONS_AND_GENOME_SRR1046787/ -n1 SRR1046762 -n2 SRR1046787 -n
AMPLICONS_AND_GENOME_SRR1046762_VS_SRR1046787
```

The output from these files will consist of:

1. COMPARISON_SAMPLES_QUANTIFICATION_SUMMARIES.txt: this file contains a summary of the quantification for each of the two conditions for each region and their difference (read counts and percentages for the various classes: Unmodified, NHEJ, MIXED NHEJ-HDR and HDR).
2. A set of folders with CRISPRessoCompare reports on the common regions with enough reads in both conditions.
3. CRISPRessoPooledWGSCompare_RUNNING_LOG.txt: detailed execution log.

An example output of CRISPResso is included as Supplementary figure 15 below:

CRISPResso2

Analysis of genome editing outcomes from deep sequencing data

CRISPResso2 run information

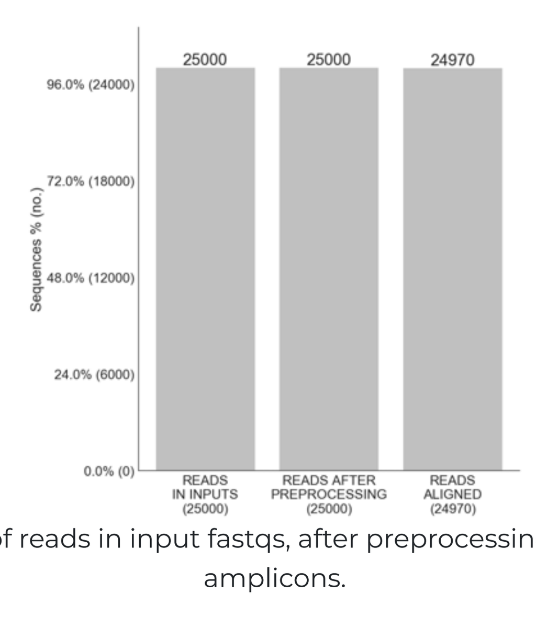


Figure 1a: The number of reads in input fastqs, after preprocessing, and after alignment to amplicons.

CRISPResso version: 2.0.14b

Run completed: 2018-10-03 17:27:49

Amplicon sequence:

```
GGCCCCAGTGGCTGCTCTGGGGGCCTCCTGAGTTTCTCATCTGTGCCCCTCCCTCCCTGGCCCAGGTGAAGGTGTGGTTCCAGAACCGGAGGACA
AAGTACAAACGGCAGAAGCTGGAGGAGGAAGGGCCTGAGTCCGAGCAGAAGAAGAAGGGCTCCCATCACATCAACCGGTGGCGCATTGCCACGAA
GCAGGCCAATGGGGAGGACATCGATGTCACCTCCAATGACTAGGGTGG
```

Guide sequence:

```
GAGTCCGAGCAGAAGAAGAA
```

Command used:

```
CRISPResso --fastq_r1 base_editor.fastq.gz --amplicon_seq
GGCCCCAGTGGCTGCTCTGGGGGCCTCCTGAGTTTCTCATCTGTGCCCCTCCCTCCCTGGCCCAGGTGAAGGTGTGGTTCCAGAACCGGAGGACA
AAGTACAAACGGCAGAAGCTGGAGGAGGAAGGGCCTGAGTCCGAGCAGAAGAAGAAGGGCTCCCATCACATCAACCGGTGGCGCATTGCCACGAA
GCAGGCCAATGGGGAGGACATCGATGTCACCTCCAATGACTAGGGTGG --guide_seq GAGTCCGAGCAGAAGAAGAA --
quantification_window_size 20 --quantification_window_center -10 --base_editor_output --
write_cleaned_report
```

Parameters:

```
aln_seed_count: 5
aln_seed_len: 10
aln_seed_min: 2
amplicon_min_alignment_score: 60
amplicon_name: Reference
amplicon_seq:
GGCCCCAGTGGCTGCTCTGGGGGCCTCCTGAGTTTCTCATCTGTGCCCCTCCCTCCCTGGCCCAGGTGAAGGTGTGGTTCCAGAACCGGAGGACA
AAGTACAAACGGCAGAAGCTGGAGGAGGAAGGGCCTGAGTCCGAGCAGAAGAAGAAGGGCTCCCATCACATCAACCGGTGGCGCATTGCCACGAA
GCAGGCCAATGGGGAGGACATCGATGTCACCTCCAATGACTAGGGTGG
auto: False
base_editor_output: True
coding_seq:
conversion_nuc_from: C
conversion_nuc_to: T
crispresso1_mode: False
debug: False
```

Allele assignments

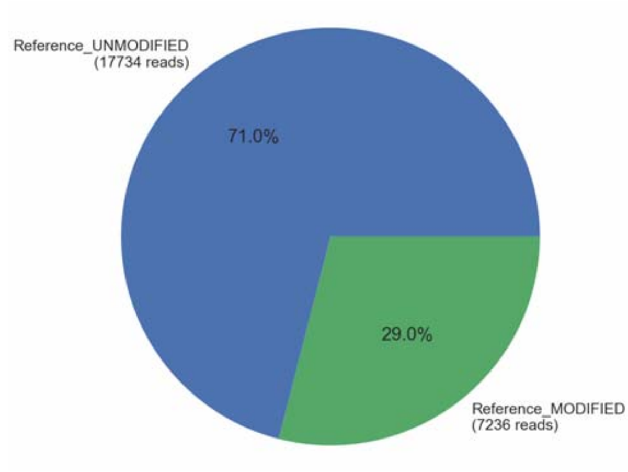


Figure 1b: Alignment and editing frequency of reads as determined by the percentage and number of sequence reads showing unmodified and modified alleles.

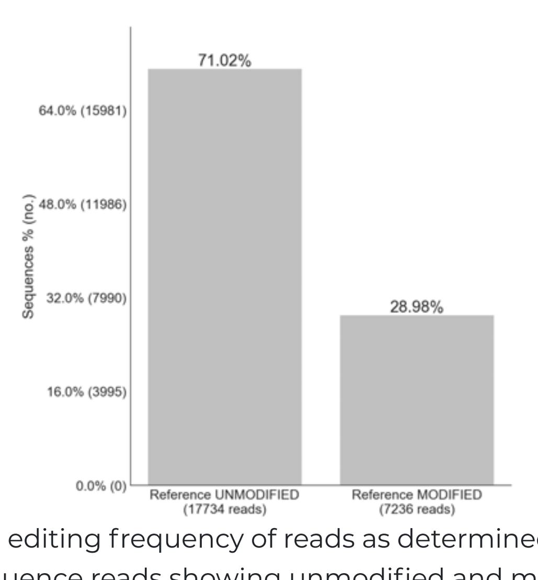


Figure 1c: Alignment and editing frequency of reads as determined by the percentage and number of sequence reads showing unmodified and modified alleles.

Nucleotide composition

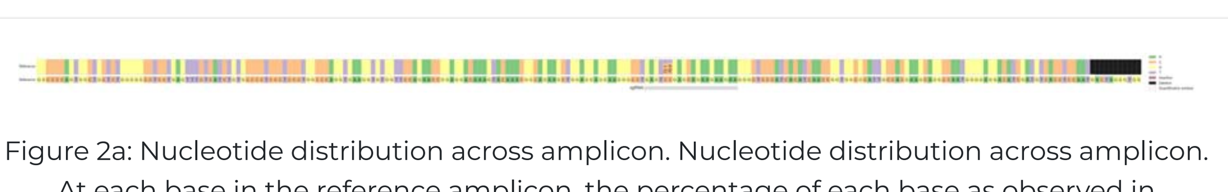


Figure 2a: Nucleotide distribution across amplicon. Nucleotide distribution across amplicon. At each base in the reference amplicon, the percentage of each base as observed in sequencing reads is shown (A = green; C = orange; G = yellow; T = purple). Black bars show the percentage of reads for which that base was deleted. Brown bars between bases show the percentage of reads having an insertion at that position.

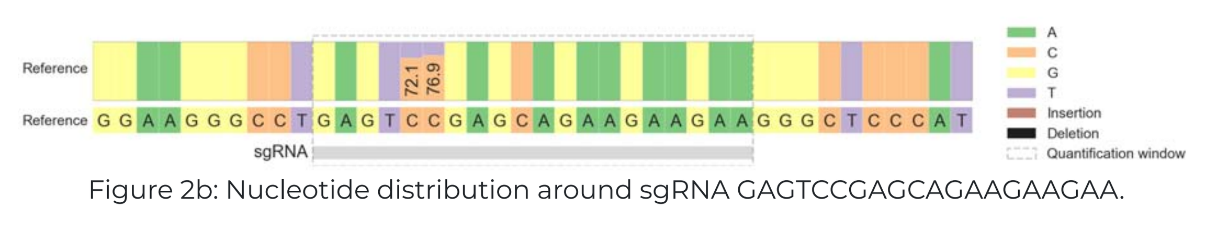


Figure 2b: Nucleotide distribution around sgRNA GAGTCCGAGCAGAAGAAGAA.

Modification lengths

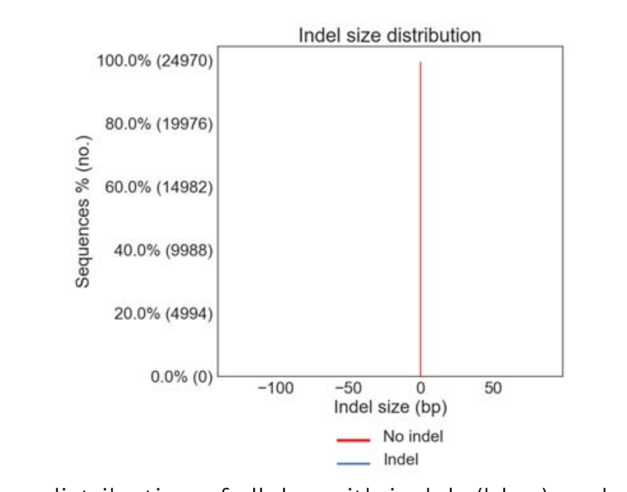


Figure 3a: Frequency distribution of alleles with indels (blue) and without indels (red).

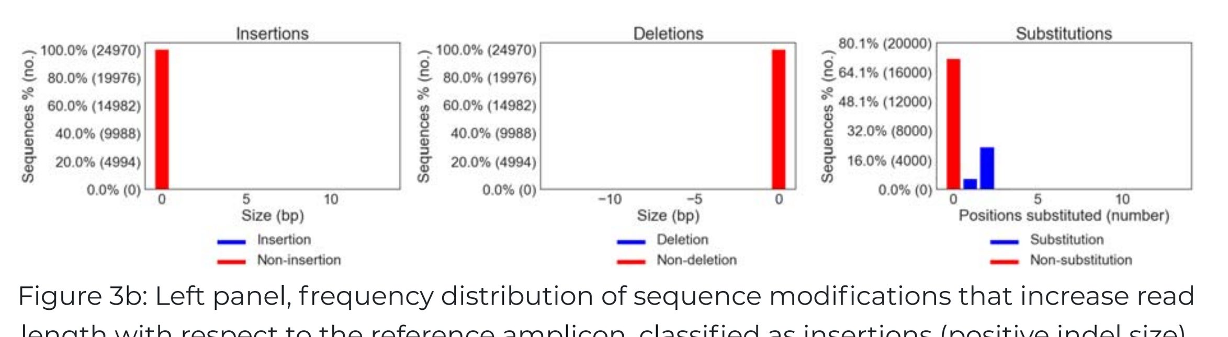


Figure 3b: Left panel, frequency distribution of sequence modifications that increase read length with respect to the reference amplicon, classified as insertions (positive indel size). Middle panel, frequency distribution of sequence modifications that reduce read length with respect to the reference amplicon, classified as deletions (negative indel size). Right panel, frequency distribution of sequence modifications that do not alter read length with respect to the reference amplicon, which are classified as substitutions (number of substituted positions shown).

Indel characterization

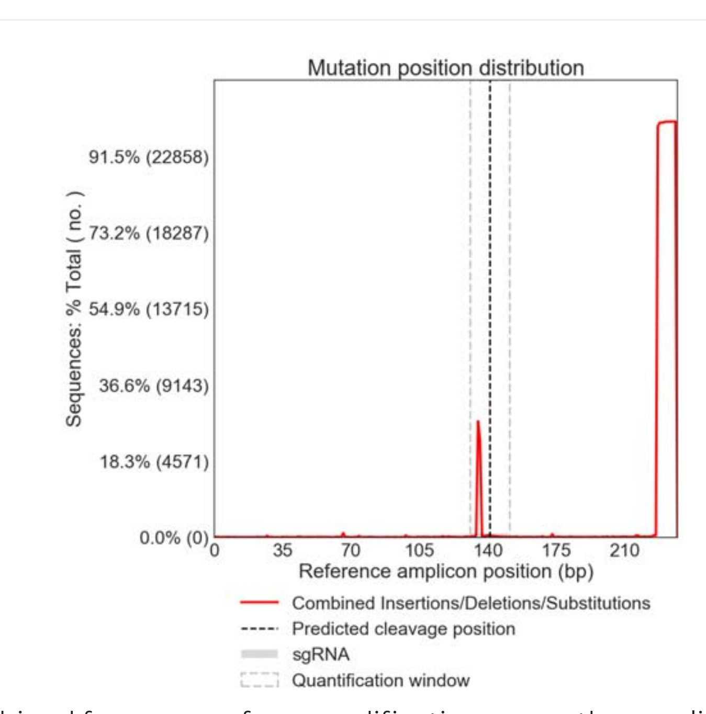


Figure 4a: Combined frequency of any modification across the amplicon. Modifications outside of the quantification window are also shown.

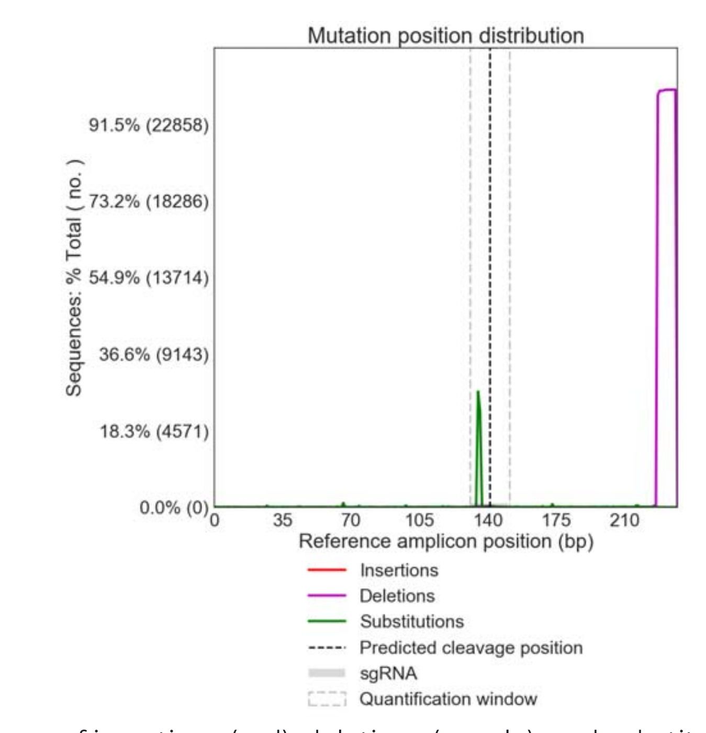


Figure 4b: Frequency of insertions (red), deletions (purple), and substitutions (green) across the entire amplicon, including modifications outside of the quantification window.

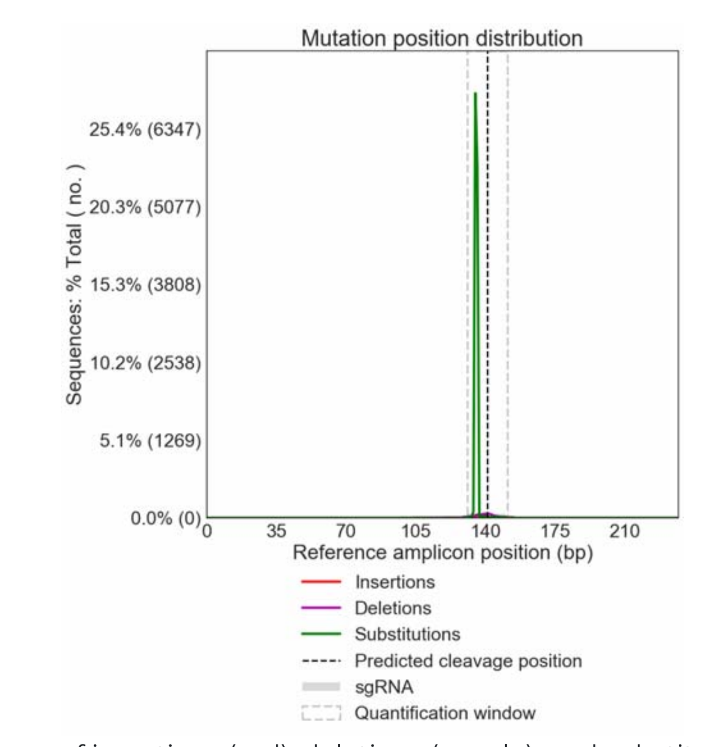


Figure 4c: Frequency of insertions (red), deletions (purple), and substitutions (green) across the entire amplicon, considering only modifications that overlap with the quantification window.

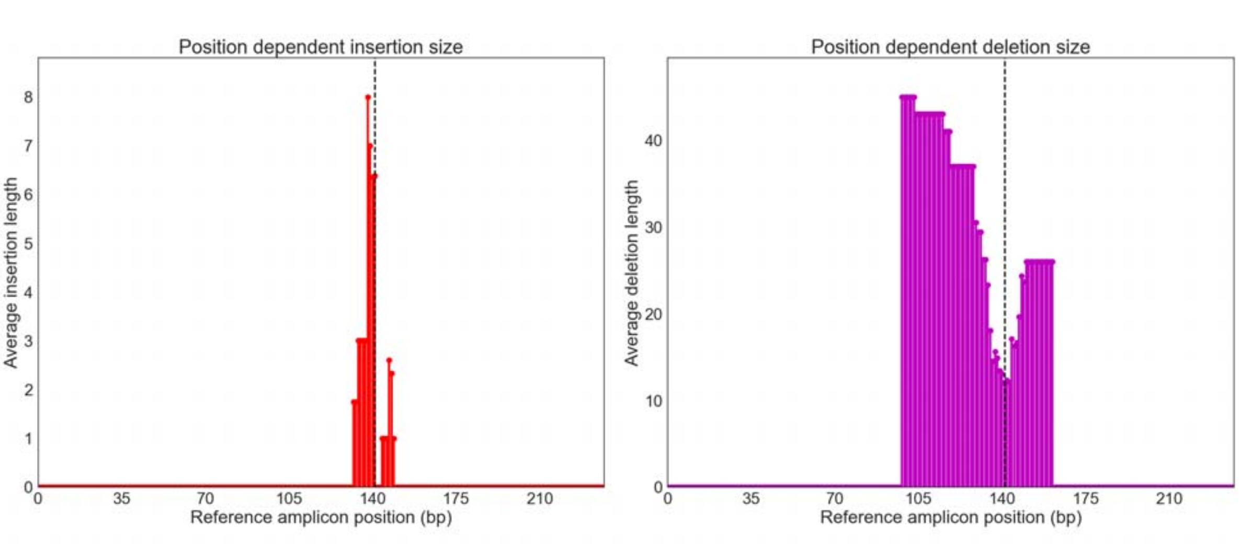


Figure 4d: Position dependent insertion size(left) and deletion size (right), including only modifications that overlap with the quantification window.

Allele plots

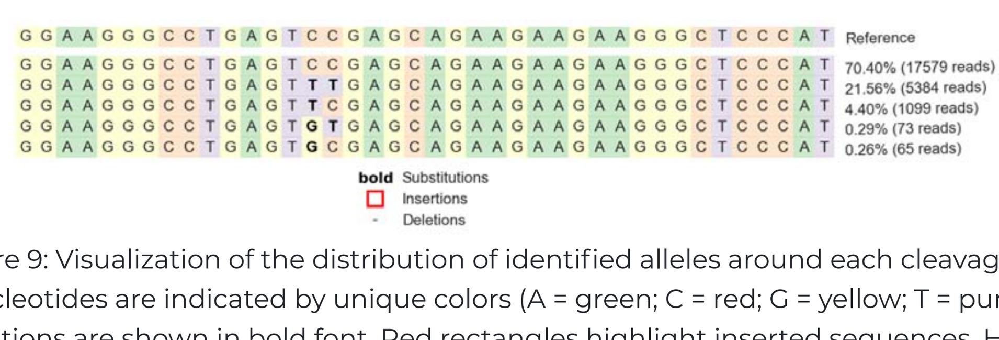


Figure 9: Visualization of the distribution of identified alleles around each cleavage site. Nucleotides are indicated by unique colors (A = green; C = red; G = yellow; T = purple). Substitutions are shown in bold font. Red rectangles highlight inserted sequences. Horizontal dashed lines indicate deleted sequences. The vertical dash line indicates the predicted cleavage site.

Base editing

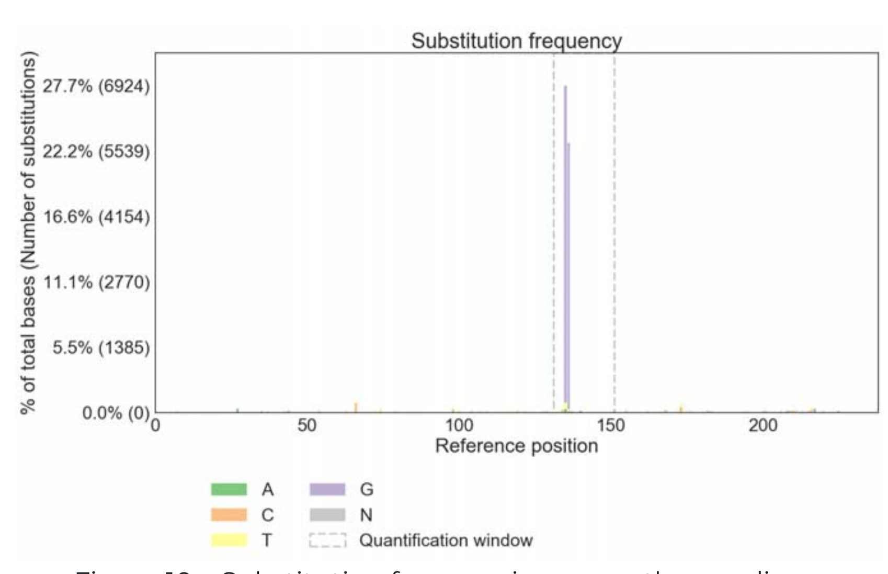


Figure 10a: Substitution frequencies across the amplicon.

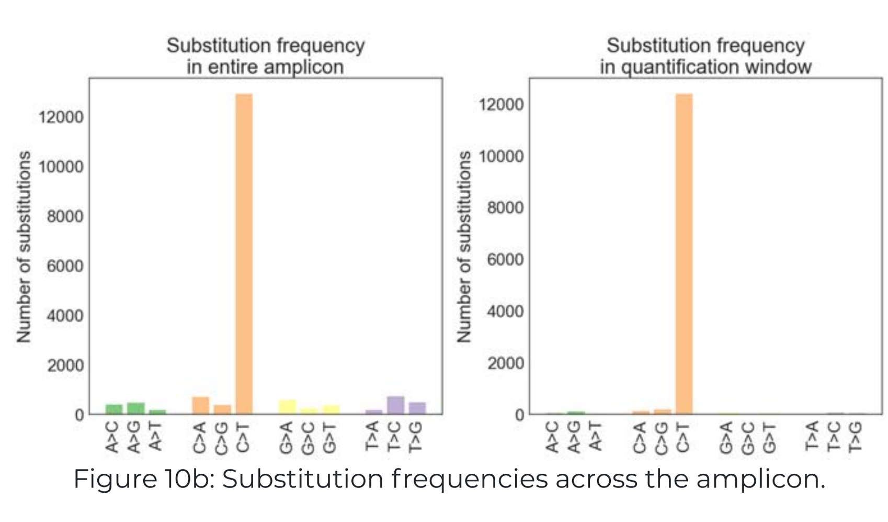


Figure 10b: Substitution frequencies across the amplicon.

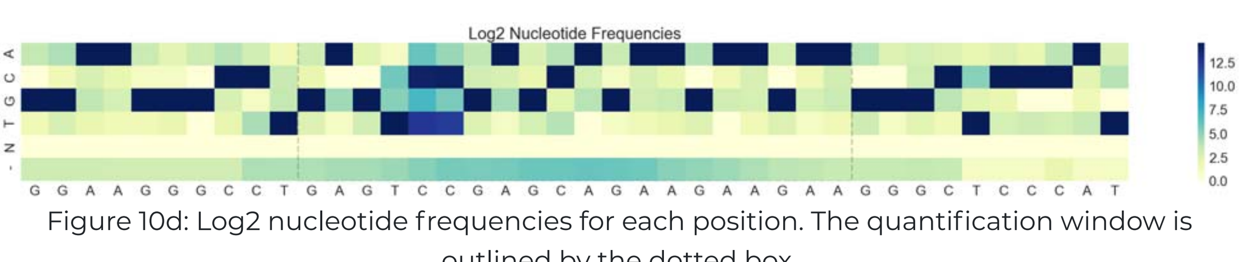


Figure 10d: Log2 nucleotide frequencies for each position. The quantification window is outlined by the dotted box.

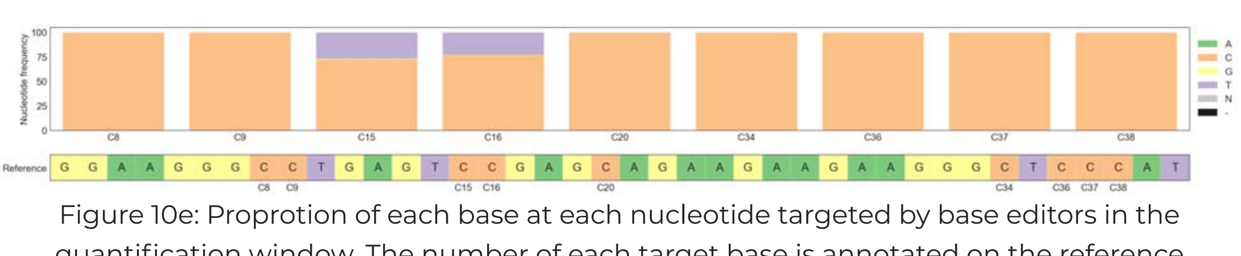


Figure 10e: Proprotion of each base at each nucleotide targeted by base editors in the quantification window. The number of each target base is annotated on the reference sequence at the bottom of the plot.

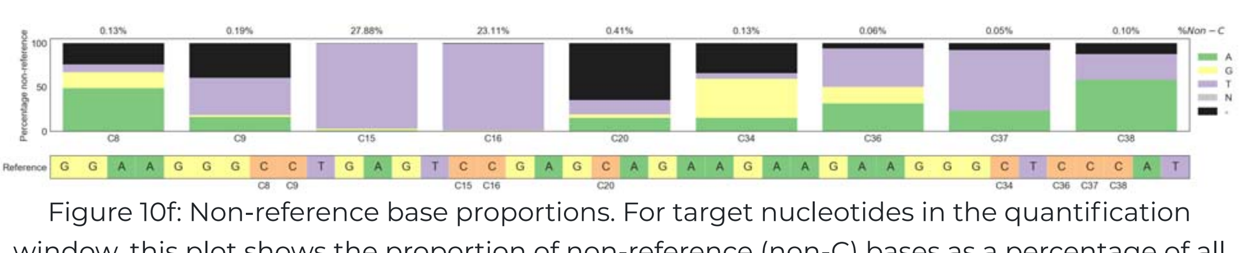


Figure 10f: Non-reference base proportions. For target nucleotides in the quantification window, this plot shows the proportion of non-reference (non-C) bases as a percentage of all non-reference sequences. The number of each target base is annotated on the reference sequence at the bottom of the plot.

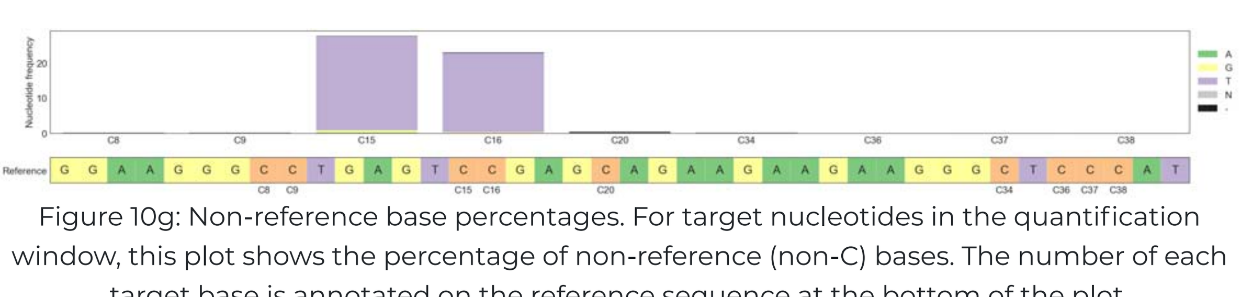


Figure 10g: Non-reference base percentages. For target nucleotides in the quantification window, this plot shows the percentage of non-reference (non-C) bases. The number of each target base is annotated on the reference sequence at the bottom of the plot.

Supplementary Figure 15: CRISPResso2 output on base editor data including reports of alignment, editing rates, indel and substitution locations, and base editing conversion efficiencies.

References

Needleman, S. B. & Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. 48, 443–453.

Gotoh, O. (1990) An improved algorithm for matching biological sequences. Journal of Molecular Biology 162(3), 705–708.

Huang, W., Li, L., Myers, J. R. & Marth, G. T. (2011) ART: a next-generation sequencing read simulator. Bioinformatics 28, 593–594.

Li H. et al. (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics, 25, 1754–1760.

Mose L.E. et al. (2014) ABRA: improved coding indel detection via assembly-based realignment. Bioinformatics, 30, 2813–2815.

Park J. et al. (2017) Cas-analyzer: an online tool for assessing genome editing results using NGS data. Bioinformatics, 33, 286–288.

Pinello, L. et al. (2016) Analyzing CRISPR genome-editing experiments with CRISPResso. Nature Biotechnology 34, 695–697

Rose, J. C. et al. (2017) Rapidly inducible Cas9 and DSB-ddPCR to probe editing kinetics. Nature Methods 14, 891–896.

Wang X, et al. (2017) CRISPR-DAV: CRISPR NGS data analysis and visualization pipeline. Bioinformatics, 33, 3811-3812