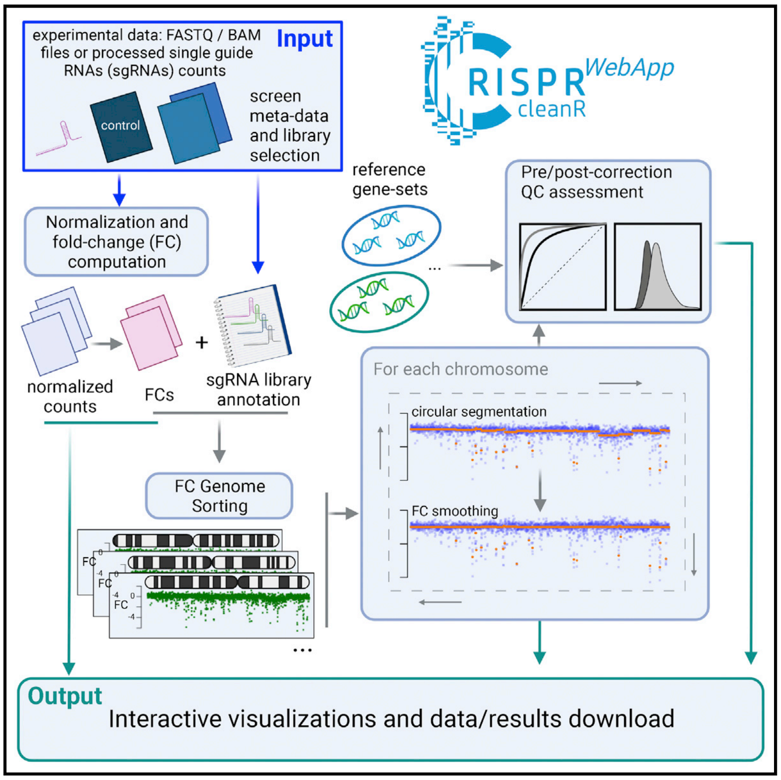
Cell Reports
Methods

Report

An interactive web application for processing, correcting, and visualizing genome-wide pooled CRISPR-Cas9 screens

Graphical abstract


experimental data: FASTQ / BAM files or processed single guide RNAs (sgRNAs) counts
Input
control
screen meta-data and library selection
CRISPR cleanR WebApp
reference gene-sets
Pre/post-correction QC assessment
Normalization and fold-change (FC) computation
normalized counts
FCs
+
sgRNA library annotation
For each chromosome
circular segmentation
FC smoothing
FC Genome Sorting
FC
...
Output
Interactive visualizations and data/results download

Highlights

- CRISPRcleanR corrects biases in CRISPR-Cas9 screens
- CRISPRcleanR^{WebApp} is a web-based, user-friendly front end for CRISPRcleanR
- CRISPRcleanR^{WebApp} processes raw sequencing files and produces interactive results

Authors

Alessandro Vinceti,
Riccardo Roberto De Lucia,
Paolo Cremaschi, ...,
Krzysztof Henryk Kluczynski,
Daniel Stephen Anderson,
Francesco Iorio

Correspondence

francesco.iorio@fht.org

In brief

A major problem affecting CRISPR-Cas9 screens is the high false-positive rate in detecting essential genes found in copy-number-amplified regions. Vinceti et al. present CRISPRcleanR^{WebApp}, a web application based on the CRISPRcleanR R package that corrects such biases in an unsupervised manner. CRISPRcleanR^{WebApp} provides an intuitive and user-friendly graphical interface.

Report

An interactive web application for processing, correcting, and visualizing genome-wide pooled CRISPR-Cas9 screens

Alessandro Vinceti,[1,4] Riccardo Roberto De Lucia,[1,4] Paolo Cremaschi,[1] Umberto Perron,[1] Emre Karakoc,[2] Luca Mauri,[3] Carlos Fernandez,[3] Krzysztof Henryk Kluczynski,[3] Daniel Stephen Anderson,[3] and Francesco Iorio[1,2,5,*]
[1]Computational Biology Research Centre, Human Technopole, Viale Rita Levi-Montalcini, 1, 20157 Milano, Italy
[2]Cancer Dependency Map Analytics, Wellcome Sanger Institute, Wellcome Genome Campus, Hinxton, Cambridge CB10 1SA, UK
[3]ICT and Digitalisation, Human Technopole, Viale Rita Levi-Montalcini, 1, 20157 Milano, Italy
[4]These authors contributed equally
[5]Lead contact
*Correspondence: francesco.iorio@fht.org

https://doi.org/10.1016/j.crmeth.2022.100373

MOTIVATION Several computational tools exist for correcting gene-independent responses to CRISPR-Cas9 targeting. Among these, CRISPRcleanR has been the first to correct such biases in an unsupervised manner (i.e., not requiring input copy-number data of the screened model). To broaden the CRISPRcleanR user base, we enriched it with original functionalities and developed CRISPRcleanR^{WebApp}, a web application front-end to the CRISPRcleanR R package. CRISPRcleanR^{WebApp} provides an intuitive graphical interface and automated workflows for complete CRISPR-Cas9 data processing, analysis, and visualization.

SUMMARY

A limitation of pooled CRISPR-Cas9 screens is the high false-positive rate in detecting essential genes arising from copy-number-amplified genomics regions. To solve this issue, we previously developed CRISPRcleanR: a computational method implemented as R/python package and in a dockerized version. CRISPRcleanR detects and corrects biased responses to CRISPR-Cas9 targeting in an unsupervised fashion, accurately reducing false-positive signals while maintaining sensitivity in identifying relevant genetic dependencies. Here, we present CRISPRcleanR^{WebApp}, a web application enabling access to CRISPRcleanR through an intuitive interface. CRISPRcleanR^{WebApp} removes the complexity of R/python language user interactions; provides user-friendly access to a complete analytical pipeline, not requiring any data pre-processing and generating gene-level summaries of essentiality with associated statistical scores; and offers a range of interactively explorable plots while supporting a more comprehensive range of CRISPR guide RNAs' libraries than the original package. CRISPRcleanR^{WebApp} is available at https://crisprcleanr-webapp.fht.org/.

INTRODUCTION

The advent of genome editing methods based on the clustered regularly interspaced short palindromic repeats (CRISPR) system has revolutionized the way molecular biology is investigated and potential therapeutic targets are discovered and prioritized.[1–6] One of the main applications of this technology has consisted of probing each gene's potential in selectively reducing the viability of cancer cells upon inactivation,[7–10] and large panels of immortalized tumor cell lines have been CRISPR screened to identify genomic-context-specific cancer vulnerabilities that might be exploited therapeutically.[6,11–14]

The efficiency of the CRISPR-Cas9 system originates from its mode of action: the induction of DNA double-strand breaks (DSBs) inflicted by the Cas9 enzyme on the genomic region matched by a given single guide RNA (sgRNA).[15] DSBs are repaired by non-homologous end joining, an error-prone mechanism causing small insertions and deletions, resulting in premature stop codons and, thus, efficient gene silencing.[16–21] A significant downside of this system is that, when used to target genomic copy-number (CN)-amplified regions, the Cas9 enzyme causes a large number of DSBs, resulting in highly cytotoxic effects independent from the targeted gene's function or expression, thus leading to false-positive essential gene calls.[1,10,22–26]

We and others have proposed computational methods to address this problem *in silico* by analyzing sgRNA counts and log fold changes (logFCs).[1,24,25,27] We developed CRISPRcleanR,[1] the first tool working in an unsupervised way, not requiring any information on the CN-alteration profiles of the screened models as input and not making any assumption on the properties of the genome to which the gene-independent responses to CRISPR-Cas9 targeting are due. CRISPRcleanR is implemented as R (https://github.com/francescojm/CRISPRcleanR) and Python package (https://github.com/cancerit/pyCRISPRcleanR), and it is available as an image for docker and cloud environments (https://dockstore.org/containers/quay.io/wtsicgp/dockstore-pycrisprcleanr).

To make CRISPRcleanR more accessible, we have enriched it with original capabilities and developed CRISPRcleanR^{WebApp}, a web-based, user-friendly, and interactive application that enables using all CRISPRcleanR functionalities through an intuitive web interface. CRISPRcleanR^{WebApp} wraps the native R package, avoiding low-level programming language interactions while providing the same processing and data analysis capabilities as the original package, plus original interactive data/result exploration modalities.

Here, we provide an overview of CRISPRcleanR^{WebApp} functionalities and data exploration modalities. Furthermore, we report results from comparing CRISPRcleanR^{WebApp} outcomes obtained on data from CRISPR-Cas9 screens of the same cell line performed using newly supported libraries. A high level of concordance across these outcomes indicates excellent compatibility of CRISPRcleanR^{WebApp} across supported libraries.

RESULTS

Overview

CRISPRcleanR^{WebApp} is a client-server web app (Figure 1A) using a recent release of the CRISPRcleanR R package (v.3.0.0, with v.0.5.0 presented in Iorio et al.[1]) through a user-friendly browser interface. CRISPRcleanR^{WebApp} provides a complete user experience offering the same native analytical possibilities of CRISPRcleanR. These encompass the workflows' usage and interactions plus original and interactive data exploration modalities and the possibility of processing low-level sequencing files (in FASTQ/BAM format). It consists of a web browser client single-page application (SPA) for user interaction and a back-end providing data storage and processing. It also includes a user authentication/authorization mechanism through a login system protecting submitted data and related results with a high level of privacy (STAR Methods). A set of video tutorials are included on the homepage of CRISPRcleanR^{WebApp}, guiding the user through every step: job submission and results identification, access, exploration, and download. In addition, constant screen responsiveness is ensured, allowing a pleasant and effective user interaction on various screen sizes, including tablets and smartphones. The application is served through a containerized architecture based at Human Technopole (the hosting research institute), making CRISPRcleanR^{WebApp} stable, easy to maintain, and scalable.

The core function of CRISPRcleanR^{WebApp} applies a circular binary segmentation algorithm[28,29] to patterns of sgRNA depletion logFCs on a per-chromosome basis. It identifies genomic regions containing sgRNA clusters with sufficiently similar depletion logFCs, which are, on average, significantly different from those in the flanking genomic regions. Since it is unlikely to observe the same fitness effect from targeting large numbers of contiguous genes, the logFCs in regions characterized by a stretch of essential genes (at least 3 in the default settings) are deemed as biased by some local features of the genomic segment (e.g., CN amplification). Subsequently, they are corrected via mean-centering. CRISPRcleanR^{WebApp} also includes a suite of tools to (1) measure, assess, and visualize the effect of said correction, (2) assemble gene-level summaries of essentiality and significance scores (collapsing sgRNA logFCs by averaging on a targeted gene basis), and (3) assess the performances of a depletion logFC rank-based classifier of prior known sets of essential/non-essential genes pre-/post-correction. CRISPRcleanR^{WebApp} also uses this latter classifier to identify and output genes significantly depleted at a fixed 5% false discovery rate of prior known non-essential genes using the approach we introduced and used in Pacini et al.[30,31] Finally, CRISPRcleanR^{WebApp} implements an inverse transformation function through which corrected sgRNA counts are derived from corrected depletion logFCs. These corrected sgRNAs are used by CRISPRcleanR^{WebApp} to compute other summaries of gene essentiality, and associated significance scores, via mean-variance modeling (using MAGeCK[32]).

These features make CRISPRcleanR^{WebApp} a one-stop tool for processing and analyzing CRISPR-Cas9 screens, producing results readily usable and interpretable by non-computational scientists.

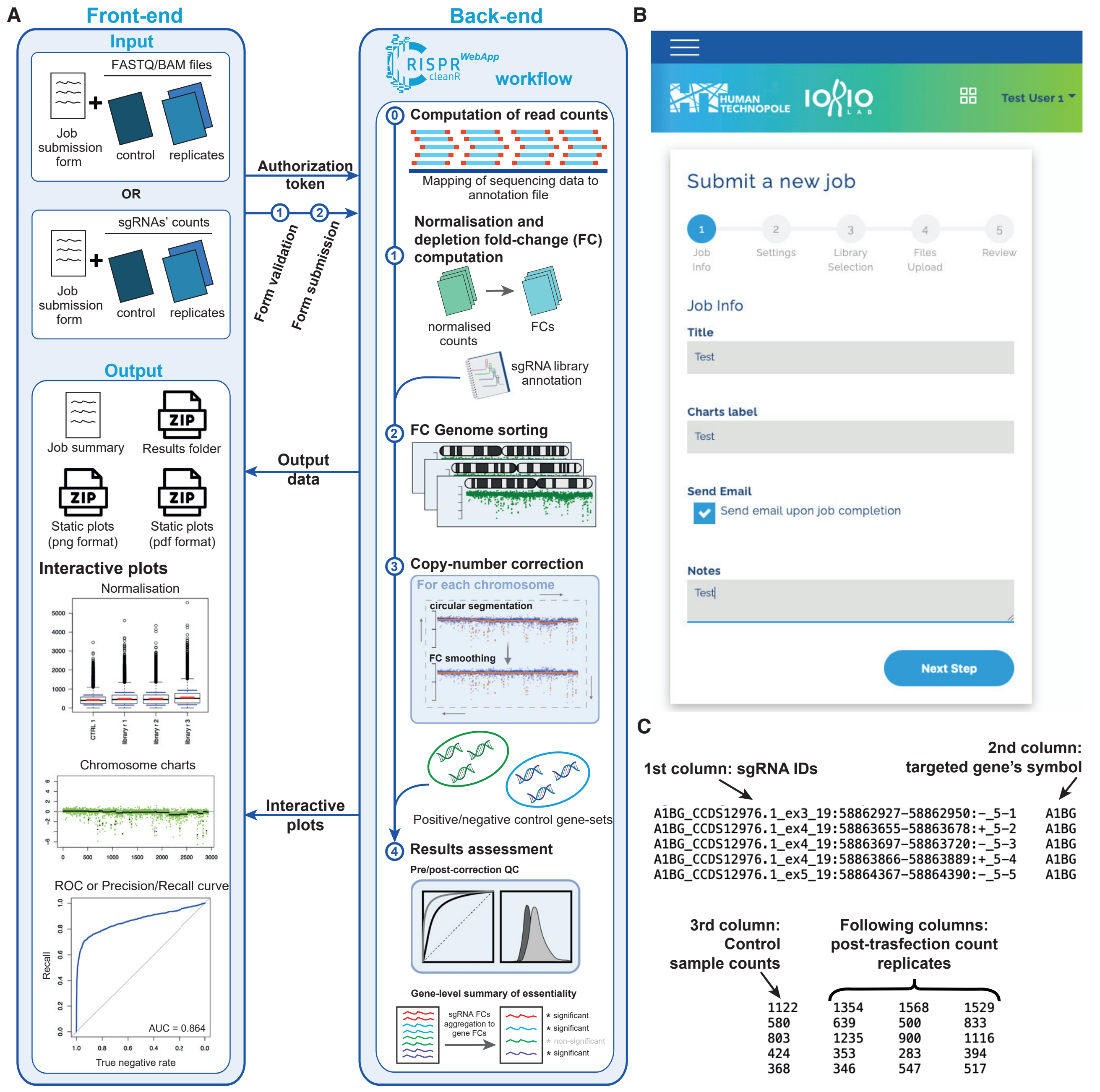
Interface and workflow setup

CRISPRcleanR^{WebApp} implements two workflows: the first one pre-processes input files (i.e., raw sgRNA counts as FASTQ, BAM, or pre-computed) as well as normalizes and corrects sgRNAs counts and depletion logFCs, and the second one implements a series of data quality control (QC) assessments and outputs interactive visualizations and result files. The first workflow encompasses calls to the complete set of CRISPRcleanR functions.

FASTQ files are converted to sgRNA counts using the mapping sequences included in the sgRNA library annotation (derived from one of the CRISPRcleanR built-in data objects or a plain text file provided by the user; STAR Methods). BAM format is also supported; in this case, the sequence identifiers are mapped to the guide identifiers to generate the sgRNA counts (STAR Methods). Users can optionally upload pre-computed sgRNA counts as plain text files.

For the subsequent normalization step, users can select different setups. For instance, sgRNA raw counts can be normalized either by scaling sample-wise based on the total number of reads or via the median ratios method.[33] In addition, users can specify the number of control samples should this be greater than one (default value). Another option is to specify the minimal value of read counts for a sgRNA in the control sample for that sgRNA to be included in the follow-up analyses (default value = 30, as in Behan et al.[6]).

The second workflow encompasses all steps needed to assess the quality of a CRISPR-Cas9 screen and visualize the results. For instance, common quality checks are based on the computation of the area under the receiver operating characteristic and precision-recall curves (AUROC and AUPRC, respectively). A profile of gene/sgRNA depletion logFCs is used as a



Submit a new job

1 Job Info — 2 Settings — 3 Library Selection — 4 Files Upload — 5 Review

Job Info

Title: Test

Charts label: Test

Send Email: ✓ Send email upon job completion

Notes: Test

Next Step

1st column: sgRNA IDs

2nd column: targeted gene's symbol

```
A1BG_CCDS12976.1_ex3_19:58862927-58862950:-_5-1    A1BG
A1BG_CCDS12976.1_ex4_19:58863655-58863678:+_5-2    A1BG
A1BG_CCDS12976.1_ex4_19:58863697-58863720:-_5-3    A1BG
A1BG_CCDS12976.1_ex4_19:58863866-58863889:+_5-4    A1BG
A1BG_CCDS12976.1_ex5_19:58864367-58864390:-_5-5    A1BG
```

3rd column: Control sample counts

Following columns: post-trasfection count replicates

```
1122   1354   1568   1529
580    639    500    833
803    1235   900    1116
424    353    283    394
368    346    547    517
```

Figure 1. Overview of CRISPRcleanR^{WebApp} design

(A) Schematic of the CRISPRcleanR^{WebApp} architecture. In the front-end, the user fills out a job submission form and uploads input files in FASTQ/BAM formats or pre-computed single-guide RNA read counts derived from a genome-wide CRISPR-Cas9 screen, alongside experiment metadata and library specification. The form is then validated and submitted to the back-end, where the data are processed following the CRISPRcleanR workflow. Results are then made available to the web interface and are explorable through a set of interactive plots on a dedicated results page.

(B) Step 1 of the CRISPRcleanR^{WebApp} job submission form: the entry point for starting any new job request after secure login. As illustrated, there are fields that the user is asked to fill out before submitting the job.

(C) Example of tab-separated file containing single-guide RNA pre-computed counts. This file is derived from screening the HT-29 cell line with the Sanger KY library. The first column contains sgRNA unique identifiers, the second one targeted gene symbols, then counts of the plasmidic DNA, followed by counts obtained 14 days post-transfection and selection of the library in three replicates.

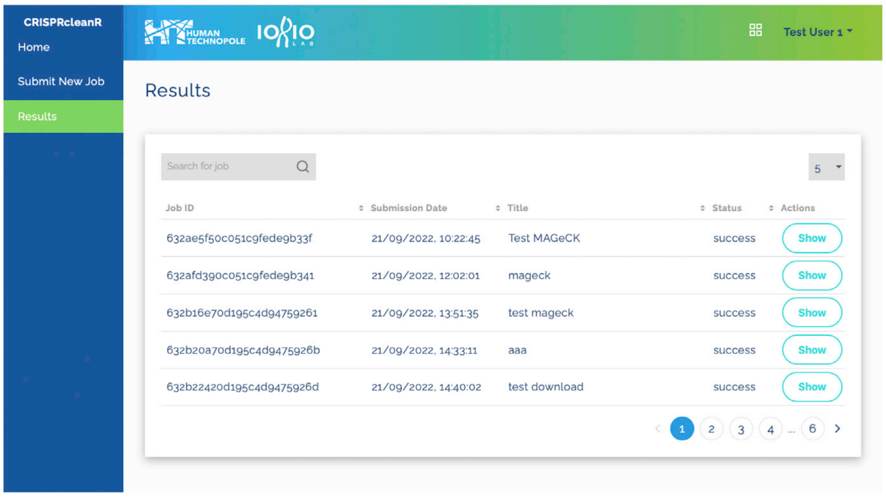
Table 1. Fields to be filled in by the user and buttons/checkboxes in the job submission form

Submission form step	Field	Type	Mandatory?	Description
1 (job info)	title	string	yes	used to identify the output in the results section once the job is completed
	charts label	string	yes	a string that will be used as a title for all the plots generated by CRISPRcleanR
	send e-mail	checkbox	optional	if selected, the user will receive a notification e-mail upon job completion
	notes	string	optional	additional notes describing the job
2 (settings)	minimal no. reads in the control sample	integer	yes (default value = 30)	minimal no. reads that each sgRNA should have in the control sample (or on average across control samples) in order to be included in the analysis
	normalization method	multiple selection	yes (default value = scaling by total numbers of reads)	normalization method for the read counts (scaling by total numbers of reads or median ratios)
3 (library selection)	library type	mutually exclusive radio buttons	yes (default value = built-in)	switch between one of the natively supported sgRNA libraries or other (external) library
	built-in library	multiple selection	yes if the selected library type is "built-in"; inactive otherwise	allows selecting one of the natively supported sgRNA libraries (AVANA, Brunello, GeCKO, KY v.1.0, KY v.1.1, MiniLibCas9, and Whitehead)
	library annotation file	file upload	yes if the selected library type is "other"; inactive otherwise	allows uploading a library annotation file
4 (files upload)	data type	mutually exclusive radio buttons	yes (default value = sgRNA counts)	allows switching between pre-computed sgRNA counts (in plain text format), FASTQ, or BAM files
	sgRNA counts file	file upload	yes if the selected data type is "sgRNA counts"; inactive otherwise	allows uploading pre-computed sgRNA counts as plain text file
	no. of controls	integer	yes with default value = 1 (if selected data type is "sgRNA counts"; inactive otherwise)	no. of control samples in the screen, i.e., columns in the sgRNA count file
	FASTQ controls	file upload	yes if the selected data type is "FASTQ"; inactive otherwise	allows uploading control sample(s) as FASTQ files
	FASTQ samples	file upload	yes if the selected data type is "FASTQ"; inactive otherwise	allows uploading screen replicates as FASTQ files
	BAM controls	file upload	yes if the selected data type is "BAM"; inactive otherwise	allows uploading control sample(s) as BAM files
	BAM samples	file upload	yes if the selected data type is "BAM"; inactive otherwise	allows uploading screen replicates as BAM files
5 (review)	submit	button	yes	job submission finalization

rank-based classifier of two built-in sets of prior known essential and non-essential genes (or their targeting sgRNAs). All the results are then summarized in data plots that can be queried through interactive visualizations or downloaded in static graphic formats (i.e., PNG or PDF).

These workflows are seamlessly integrated within CRISPR cleanR^{WebApp}. After secure login, the user only needs to fill out the initial parameters, including experimental metadata, in a job submission form (Figure 1B; Table 1) and upload properly formatted input files containing sequencing data (FASTQ/BAM files) or the sgRNA pre-computed counts (Figure 1C) to run the analyses.

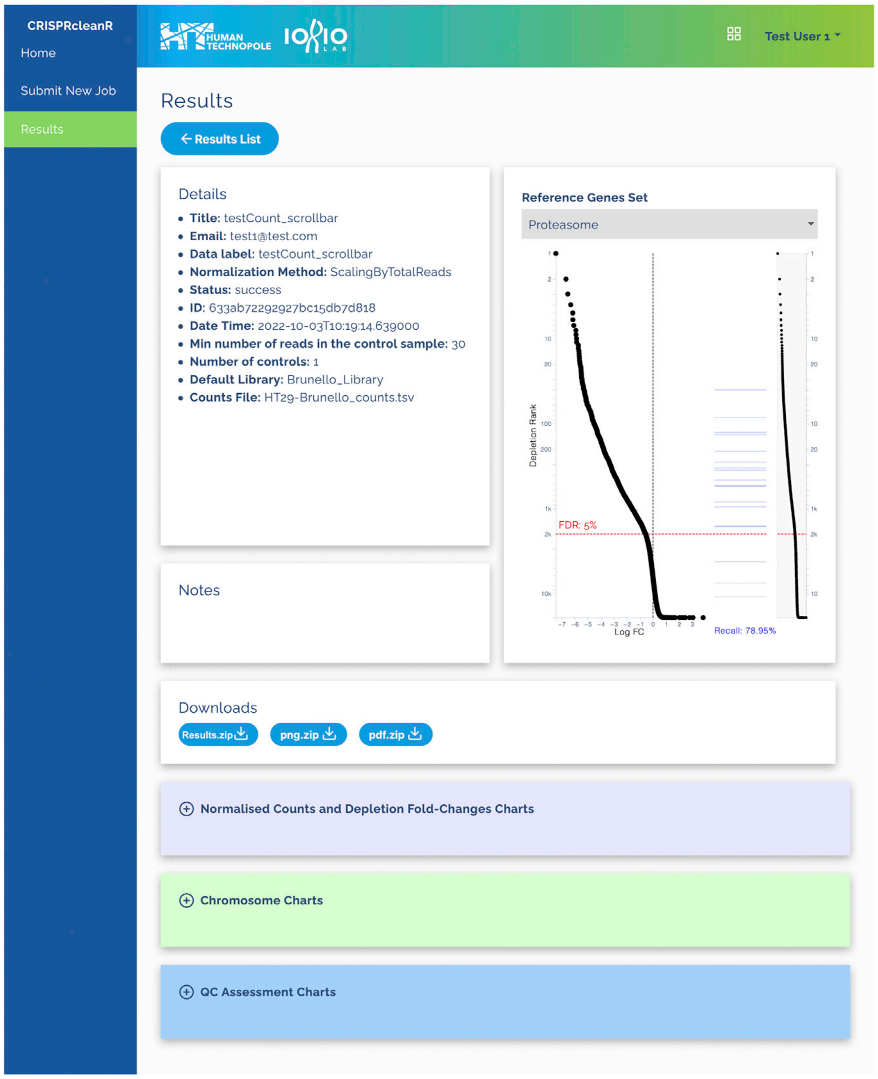
If the user opts to upload pre-computed read counts, these need to be in a comma- or tab-separated plain text with csv or tsv extensions. In this file, the first two columns must include unique sgRNA identifiers and their targeted gene symbol, A



Results

Job ID	Submission Date	Title	Status	Actions
632ae5f50c051c9fede9b33f	21/09/2022, 10:22:45	Test MAGeCK	success	Show
632afd390c051c9fede9b341	21/09/2022, 12:02:01	mageck	success	Show
632b16e70d195c4d94759261	21/09/2022, 13:51:35	test mageck	success	Show
632b20a70d195c4d9475926b	21/09/2022, 14:33:11	aaa	success	Show
632b22420d195c4d9475926d	21/09/2022, 14:40:02	test download	success	Show

B



Results

Details

- **Title:** testCount_scrollbar
- **Email:** test1@test.com
- **Data label:** testCount_scrollbar
- **Normalization Method:** ScalingByTotalReads
- **Status:** success
- **ID:** 633ab72292927bc15db7d818
- **Date Time:** 2022-10-03T10:19:14.639000
- **Min number of reads in the control sample:** 30
- **Number of controls:** 1
- **Default Library:** Brunello_Library
- **Counts File:** HT29-Brunello_counts.tsv

Notes

Downloads

Normalised Counts and Depletion Fold-Changes Charts

Chromosome Charts

QC Assessment Charts

(legend on next page)

followed by one or more (in case of multiple controls) columns containing plasmid/control read counts. The remaining columns must contain replicates of the sgRNA counts obtained post-selection and amplification in the CRISPR screen (Figure 1C). The entire form is checked for potentially incorrect file formats or missing parameter specifications to prevent incomplete or inconsistent input data. After job submission, the user is notified of the outcome of this process (i.e., successful submission or submission error).

The CRISPRcleanR^{WebApp} homepage contains a link to download example input files in a single compressed folder. This folder contains pre-computed sgRNA read counts obtained by screening the HT-29 cell line with the Sanger KY library,[7] derived from Behan et al.[6] It includes one control sample and three post-selection/amplification read count replicates. Read counts from a similar experiment are also included in FASTQ format (one for the plasmid/control DNA, i.e., test_plasmid.fq.gz, and two for the screen replicates, i.e., test_sample1.fq.gz and test_sample2.fq.gz, downsampled to reduce file size). Finally, the example data folder contains a text file with the annotation of the KY v.1.0 library and a README file describing the folder's content.

Results exploration and interactive plots

Following job submission, the CRISPRcleanR^{WebApp} back-end server starts an offline computation, sending an e-mail message to the user once finished (if this option was selected in the job submission form). A results entry is immediately listed on the results page (Figure 2A), where a data table offers a configurable pagination size for splitting large sets of entries into smaller chunks. In this table, each row refers to a job submitted by the logged user, and it shows main details such as submission date and time, job title, and job status (succeed, failed, or pending). Furthermore, the table allows job filtering and sorting according to any of the column fields. Through this page, the user can access and interactively explore the results obtained from their job submissions (Figure 2B). The job results page shows a detailed summary that recapitulates the parameters specified by the user in the job submission form and a series of sections allowing exploring and downloading all data and results and access to all the interactive plots outputted by CRISPRcleanR^{WebApp}. The downloads section contains links to all the results, which can be downloaded as zipped folders, and to all the plots, downloadable as static images in PDF or PNG format.

All the plots can also be visualized interactively and are equipped with a tooltip providing detailed information when hovering the pointer on a graphic component. The job results page also includes a summary gene-signature plot. The other plots are collected within image accordions, containing clickable thumbnails, and partitioned into three different sections (Figure 2B): normalized counts and depletion logFC charts, chromosome charts, and QC assessment charts. All charts include a zoom area, often showing a mini graph representation of the overall chart, where users can select an area to magnify in the main chart. The gene-signature plot interactively shows results from the normalization and the depletion logFCs and count correction, i.e., the chromosome plots. On the top left of the results page, a plot shows all screened genes with coordinates indicating depletion logFCs (x axis) and depletion rank position (y axis), respectively (Figure 2B). The user can select one of seven control gene sets known as essential or non-essential genes[1,34–36] and explore how they rank based on their depletion logFCs on the right part of the chart. A red line indicates the rank position, above which a false discovery rate (FDR) of non-essential genes is $<5\%$ (when considering all genes in the previous rank positions as positive predictions), and it is determined using the logFC distributions of the BAGEL essential and non-essential genes (STAR Methods).

The "normalized counts and depletion logFCs" section contains two interactive plots: the first one shows a comparison between raw and normalized sgRNA read counts (Figure 3A), whereas the second shows uncorrected depletion logFCs (Figure 3C) across samples in the input file.

The user can toggle between raw and normalized read counts by clicking on a switch button in the upper left corner of the corresponding interactive plot (Figure 3A). A tooltip provides information on a given point, i.e., guide ID, exon, gene, and raw or normalized read count, when moving the mouse over it. The same functionalities, except for the switch button, are accessible in the interactive plot showing the uncorrected logFCs (Figure 3C).

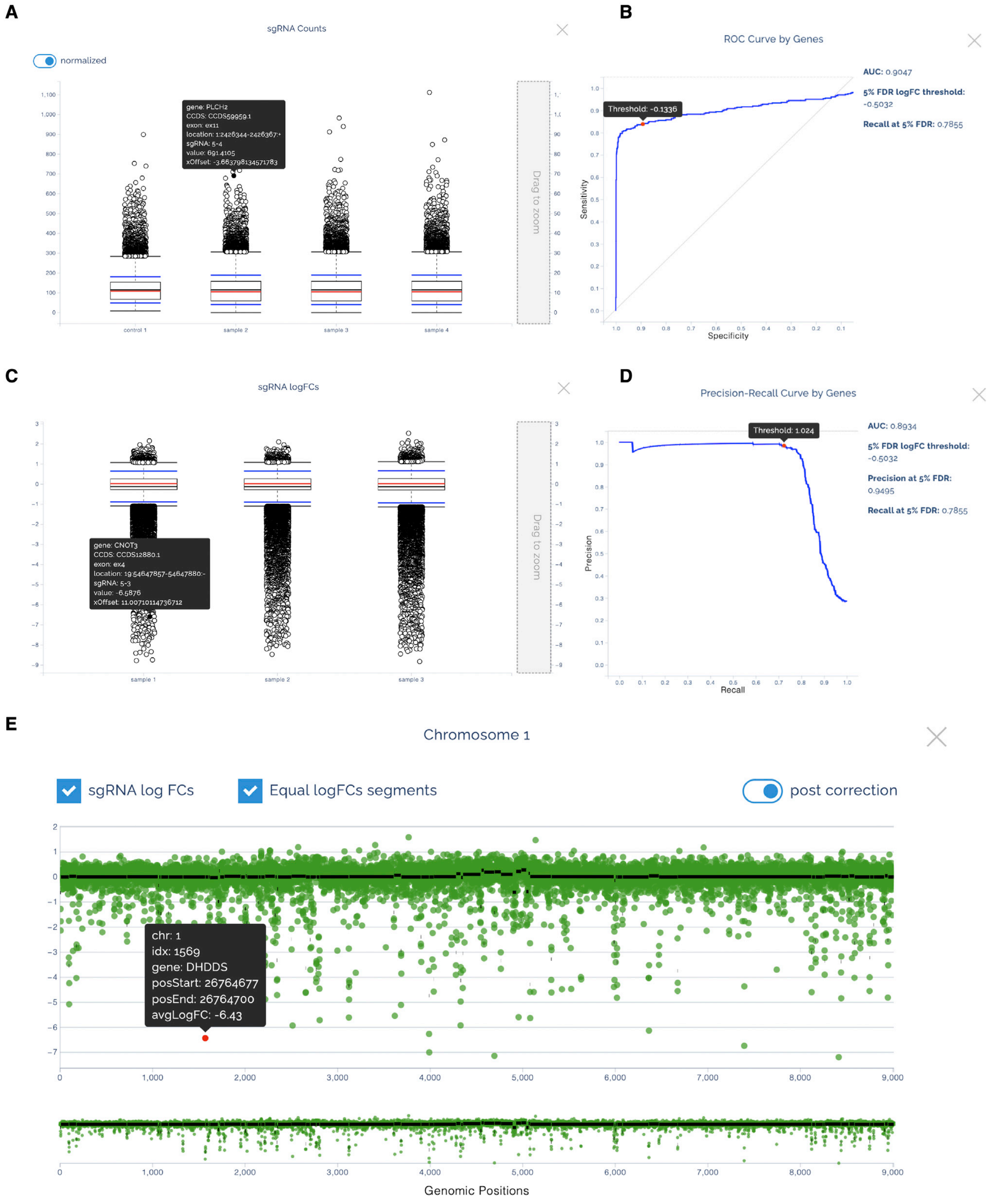
The "QC assessment" section contains a series of plots summarizing data quality checks performed on the sgRNA/gene-level depletion logFCs corrected by CRISPRcleanR^{WebApp} (Figures 3B and 3D). Particularly, they include visualizations of ROC and precision/recall curves (Figures 3B and 3D) computed on sgRNA- or gene-level-corrected depletion logFC profiles when considering rank-based classifiers of essential and non-essential genes. A tooltip provides information regarding the logFC threshold, below which a certain level of recall (for the AUROC plot) or precision (for the AUPRC plot) is achieved.

The "chromosome charts" section (Figure 3E) shows one plot per chromosome, with segments of sufficiently similar sgRNA depletion logFCs on the x axis and the corresponding average logFC pre-/post-CRISPCcleanR correction on the y axis (with the 23rd and 24th charts corresponding to X and Y/X chromosomes, respectively). Each chromosome chart has two checkboxes in the upper left corner that allow the user to focus on the segments (black lines) or the components of the sgRNA depletion logFC (green dots). A switch button in the upper right corner allows toggling between uncorrected and corrected logFCs. Also in this case, the segment tooltip shows relevant

Figure 2. Results section of CRISPRcleanR^{WebApp}

(A) The results list page: here, the user can find all results obtained from their submitted jobs.

(B) Job results are accessible by selecting a job ID in the results list page. The page is made of several sections including a detailed description of the job, an interactive summary plot where all the screened genes are ranked based on their depletion fold changes, with overlaid signatures of control genes. On the bottom, three image accordions contain clickable thumbnails for opening the related interactive plot rendered by the web application. Finally, the downloads section includes links to all plots as static images (in PDF or PNG format), as well as to both input and processed data and results (as unique zipped folder of plain text files).



(legend on next page)

information about the segment location on the genome and average depletion logFC of the mapped sgRNAs; the sgRNA tooltip provides information about the gene portion targeted by the sgRNA and depletion logFCs (uncorrected or corrected).

Extended sgRNA libraries' support

The original version of CRISPRcleanR (v.0.5.0) supported only the AVANA[37] and Sanger KY[6,7] CRISPR-Cas9 sgRNA libraries. CRISPRcleanR^{WebApp} builds upon and uses CRISPRcleanR v.3.0.0, which we have extended to support the following additional genome-wide libraries: Brunello,[38] GeCKOv2,[5,39] Whitehead,[9,10,40] and the recent MiniLibCas9[41] library of minimal size. Except for GeCKOv2, all these libraries' (publicly available) annotations include genomic coordinates of the sgRNAs' targeted sequences. CRISPRcleanR needs this information to genome sort sgRNA depletion logFCs before correction. For the GeCKOv2 library, this information was not available. We remapped the GeCKOv2 sgRNA sequences onto the human genome (GRCh38 - hg38) and assembled a CRISPRcleanR-compatible annotation object (STAR Methods) for this library. CRISPRcleanR^{WebApp} also supports any other genome-wide CRISPR library provided that the user uploads a custom library annotation file. In this case, the sgRNA sequences in the read count files are matched to the ones provided in the library annotation file, and the workflow proceeds in the case of successful retrieval of at least 80% of the guides (STAR Methods). This functionality further extends the applicability of CRISPRcleanR^{WebApp} to a larger pool of CRISPR screens performed on 2D cell lines or alternative cancer models (e.g., primary cultures, organoids, or patient-derived xenografts).

We tested the ability of CRISPRcleanR to support the extended set of libraries described above in terms of correction performances and results conservation across analyses of screens of the same cell line. We tested CRISPRcleanR on screens performed with different genome-wide CRISPR-Cas9 libraries on HT-29, a human cancer cell line derived from colorectal carcinoma that is frequently used to assess the sensitivity and specificity of CRISPR-Cas9 libraries, and with multiple public available CRISPR screen datasets.[6,37]

First, we compared the extent of CRISPRcleanR correction on the different screens. We contrasted gene depletion logFCs pre- versus post-correction and compared differences across libraries (Figure 4A). Before this comparison, we averaged sgRNA-level depletion logFCs on a targeted gene basis to obtain gene-level depletion logFCs. As the tested screens presented different depletion logFC ranges and phenotype penetrance due to inherent differences in the employed sgRNAs' sequences and for interscreen comparability, we concatenated screen-wise the pre- and post-correction logFC vectors. We applied a min-max normalization (STAR Methods) and then split each normalized vector back into the original components and computed differences between pre-/post-corrected logFCs.

We observed an average median of computed logFC differences across libraries equal to -0.012 (min = -0.002 for KY and max = -0.02 for GeCKOv2) and an average interquartile range equal to 0.0245 (min = 0.015 for MiniLibCas9 and max = 0.037 for AVANA; Figure 4A). These results show that the CRISPRcleanR correction has limited effects on the whole screen; it regards small sets of genes, and its minimal impact is consistent across screens of the same cell line performed with the different supported libraries.

Next, we checked the extent of CRISPRcleanR correction consistently affecting each gene's logFC across screens. We computed a normalized Shannon entropy (also known as efficiency) for each gene, quantifying the homogeneity of the correction effects on its logFC across screens. A low entropy value indicated that a gene's logFC was affected consistently across screens, whereas a high entropy value indicated the opposite (STAR Methods). We coded the gene-wise correction outcomes in a given screen as follows. A 0 indicated that the logFC of the gene under consideration was not corrected by CRISPRcleanR, i.e., the logFCs of its targeting sgRNAs were not detected as biased. A 1 indicated a positive correction, meaning that the sgRNAs targeting the gene under consideration were mapped onto a genomic segment detected as biased toward negative values by CRISPRcleanR and that their logFC increased. Following the same logic, a -1 indicated a negative correction. Applying this coding across all screens yielded for each gene a vector of 6 entries (one per each tested library) with values in $\{-1, 0, 1\}$, from which we computed the normalized Shannon entropy.

A summary of the results is provided in Figure 4B, showing the percentages of genes across efficiency values and concordance/discordance of correction effects for each value. For example, the first bar from the left (corresponding to a 0 efficiency) accounted for all the genes for which the correction outcomes were identical across screens. The second bar accounts for the corrections that agree across all the screens but one, and so on.

For over 95% (Figure 4B) of the genes, we observed the same correction effect in at least four screens out of six (efficiency < 0.63), with the absolute majority (42%) showing different outcomes in a 4:2 proportion. Finally, only 4.31% of the genes presented all three different correction outcomes across screens, and 0.39% presented them equally partitioned

Figure 3. Interactive plots rendered by CRISPRcleanR^{WebApp}

(A and C) Overview of the normalization plots: (A) boxplots for raw and normalized single-guide RNA (sgRNA) read counts across samples in the input file, and (C) CRISPRcleanR uncorrected log fold changes (logFCs). Both plots have a vertical scrollbar on the right that allows zooming in on specific portions of the plot. When hovering on each point, a tooltip shows information regarding the corresponding sgRNA, i.e., guide ID, exon, gene, and read count/depletion logFC. Box-and-whisker plots show interquartile ranges and 5th–95th percentiles; centers indicate medians.

(B and D) Quality control (QC) assessment plots. Examples of interactive receiver operating characteristic (ROC) and precision-recall curve (PRC) plots obtained from profiles of corrected depletion logFCs when considering them as rank-based classifiers of two sets of a negative/positive control genes, i.e., *a priori* known essential and non-essential genes.

(E) Interactive plot showing the sgRNA logFCs' (green dots) obtained targeting genes in the chromosome 1. The logFCs are clustered within segments of equal copy numbers (black lines), which are corrected by CRISPRcleanR. Each point is an sgRNA, and, when moving the mouse over it, a tooltip will show related information such as chromosome number, guide ID, start and end position, and depletion logFC when hovering the pointer on it.

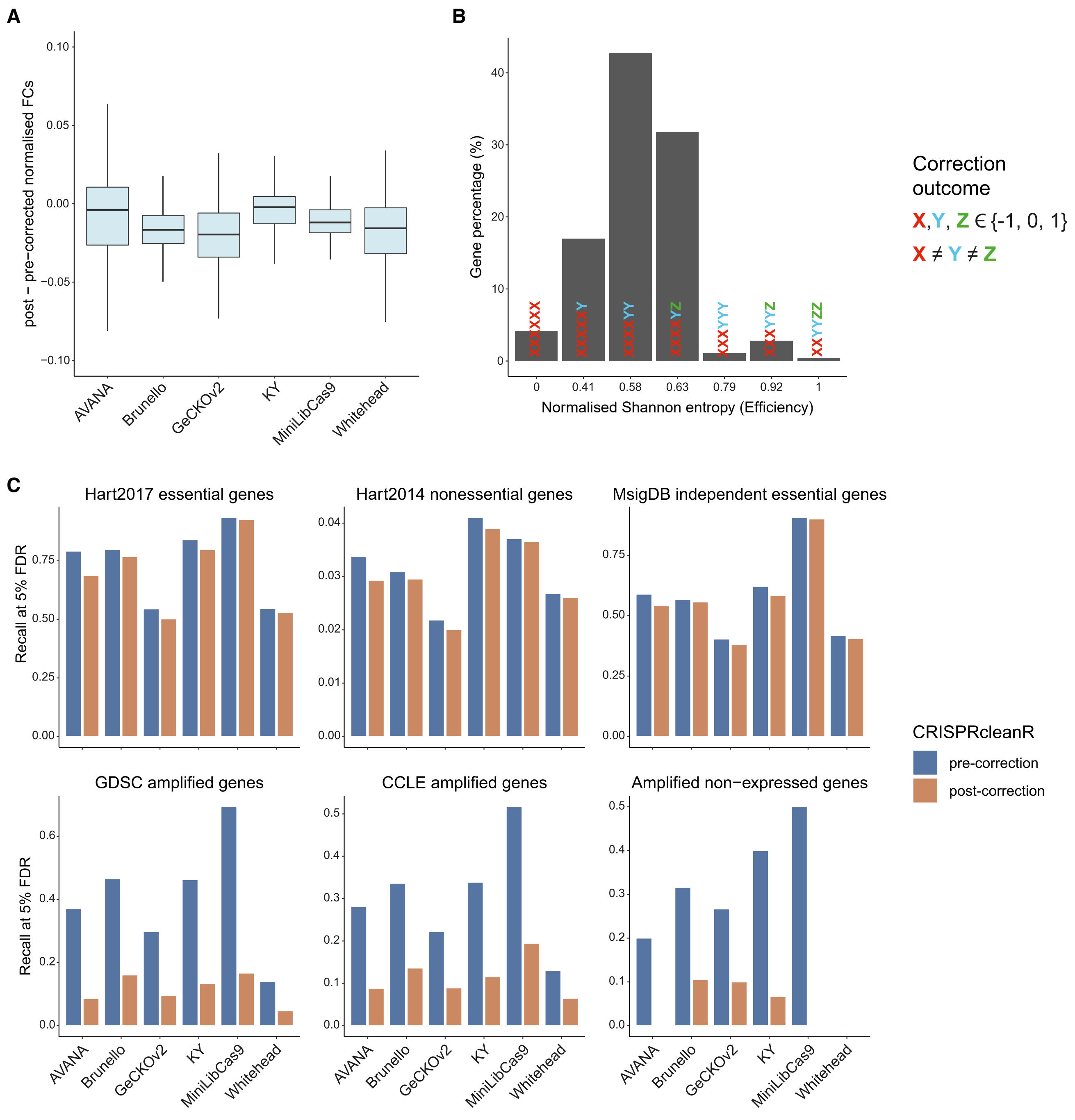


Figure 4. Assessment of CRISPRcleanR correction across different supported CRISPR-Cas9 libraries on the HT-29 cell line
(A) Comparison of CRISPRcleanR pre- and post-correction depletion fold changes (FCs) across screens. Box-and-whisker plots show interquartile ranges and 5th–95th percentiles; centers indicate medians.
(B) Normalized Shannon entropy (efficiency) of gene-wise correction outcomes (−1, 0, 1) across screens. Only genes shared across libraries were considered.
(C) Recall at 5% FDR of six prior known gene sets observed when considering pre- and post-correction (as indicated by the different colors) sgRNA logFCs' profiles as rank-based classifiers.

(i.e., two −1s, two 0s, and two 1s). These results show that the CRISPRcleanR correction affected individual genes' logFCs homogeneously across screens of the same cell line performed with the different supported libraries.

Finally, we measured the extent to which the CRISPRcleanR correction tendency in reducing false-positive gene-essentiality calls while maintaining true-positive calls was conserved across processed screens. To this aim, we considered six control sets of genes (Table S1): the Hart2017 essentials[34] and Hart2014 non-essentials,[35] respectively, as positive and negative controls, plus known essential gene sets derived from the MSigDB[30,36] accounting for cellular housekeeping processes and two final sets of HT-29-specific putative negative controls, i.e., CN-amplified genes (from two different public resources),[42–44] as well as amplified and non-expressed genes (FPKM < 0.1 in HT-29; STAR Methods).

For each gene set, we computed the recall at 5% FDR (STAR Methods) for pre- and post-corrected logFCs across the six screens (Figure 4C). For the negative controls (expected to be biased), we observed consistent reductions in recall across screens (median = 29.48%, 19.65%, and 20.53%, respectively). In contrast, for the positive controls, we observed negligible recall reductions across screens (median = 3.61%, 0.16%, and 1.74%, respectively). Thus, CRISPRcleanR can effectively correct the logFCs of amplified gene sets with high specificity without compromising the logFCs of other genes regardless of their phenotype intensity and employed supported library.

DISCUSSION

We presented CRISPRcleanR^{WebApp}, a web-based application integrating the complete suite of functionalities of the CRISPR cleanR R/python package.[1] Differently from other methods, CRISPRcleanR works in an unsupervised way, not requiring any input related to the gene CN profiles of the processed/screened model. In addition, CRISPRcleanR corrects screens on a single-sample basis, preserving the overall heterogeneity of the data and making this tool especially suited for identifying context-specific dependencies and biomarkers.[30]

We showed that CRISPRcleanR^{WebApp} does not require any prior knowledge of programming languages like R and python and offers a user-friendly interface giving full workflow control and fully customizable execution of the correction procedure on user-provided data.

The homepage of CRISPRcleanR^{WebApp} includes comprehensive video tutorials and toy datasets for testing. The job submission form can take in input FASTQ/BAM files (subject to quality checks and mapped to the library annotation file to obtain sgRNA counts) or pre-computed sgRNA counts in a plain text format. A results page allows users to download the output data and explore results through a portfolio of interactive plots.

We also enabled users to upload data from screens performed with custom libraries: in this case, sgRNA IDs are checked for consistency with respect to the library annotation file. We believe that this feature will extend the services of CRISPRcleanR^{WebApp} to a much larger audience, allowing the analysis of CRISPR-Cas9 screens performed in various models besides 2D cell lines (e.g., primary cultures, organoids, or patient-derived xenografts). These screens are amenable to CRISPRcleanR correction provided they are executed using libraries with sufficient sgRNA density and related annotation.

Here, we have provided an overview of the CRISPRcleanR^{WebApp} implementation, design, and interface and demonstrated that it yields consistent results across different technical settings and supported libraries. The features provide an easy-to-use framework for pre-processing and correcting data derived from CRISPR-Cas9 screens, which might significantly widen the CRISPRcleanR user community.

Limitations of the study

While CRISPRcleanR^{WebApp} is a user-friendly web application accessible to non-computational scientists, its current version still lacks some functionality of the original package, which we will include in subsequent versions. For instance, we are planning to extend the number of input parameters upon job submission as experienced users may require more advanced setups.

In addition, CRISPRcleanR is equipped with functions to test the depletion logFCs of sgRNAs targeting different reference gene sets (for example, prior known essential genes or CN-amplified genes) for statistically significant differences concerning the background pre- and post-CRISPRcleanR correction. This analysis aims to show that the CRISPRcleanR correction reduces false-positive essential gene calls while maintaining true-positive rates. We will visually render all these functions in future versions of CRISPRcleanR^{WebApp}.

Furthermore, CRISPRcleanR includes the ccr.impactOnPhenotype function, which computes the percentages of genes whose depletion signal is attenuated post-CRISPRcleanR correction or is potentially “distorted” (i.e., loss-of-fitness genes in the uncorrected screens becoming gain-of-fitness genes post-correction, and vice versa). In Iorio et al.,[1] we demonstrated that the amount of this distortion introduced by CRISPRcleanR is negligible; however, this analysis will also be possible in future versions of CRISPRcleanR^{WebApp}.

Finally, another feature we plan to integrate into future versions is the possibility to pipeline CRISPRcleanR^{WebApp} with existing tools that robustly estimate gene essentiality after depletion logFC correction supervisedly (like BAGEL).[45,46]

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Material availability
  - Data and code availability
- METHOD DETAILS
  - Web-application architecture
  - CRISPRcleanR^{WebApp} management of input files
  - Validation of the supported CRISPR-Cas9 libraries in the new version of CRISPRcleanR

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.crmeth.2022.100373.

ACKNOWLEDGMENTS

The graphical abstract was created with BioRender.com.

AUTHOR CONTRIBUTIONS

A.V. conceived the study, designed and performed the benchmark analyses, tested the web application, and wrote and revised the manuscript. R.R.D.L. designed, developed, and tested the web application, integrated all libraries and technologies with the IT infrastructure, and wrote and revised the manuscript. R.R.D.L. is also responsible for the maintenance and future releases of the web application. P.C. customized the core package, added original functionalities, tested the web application, and revised the manuscript. U.P. contributed to the design of the benchmark analyses, tested the web application, and revised the manuscript. E.K. assembled a CRISPRcleanR annotation for the GeCKOv2 library and revised the manuscript. L.M., C.F., K.H.K., and D.S.A. set up the IT infrastructure and technologies. F.I. conceived the study, designed the original algorithms and developed the R-based version of CRISPRcleanR, contributed to the design of the benchmark analyses, wrote and revised the manuscript, and supervised the study.

DECLARATION OF INTERESTS

F.I. receives funding from Open Targets, a public-private initiative involving academia and industry, and performs consultancy for the joint CRUK-AstraZeneca Functional Genomics Center.

Received: April 7, 2022
Revised: October 6, 2022
Accepted: December 7, 2022
Published: January 3, 2023

REFERENCES

1. Iorio, F., Behan, F.M., Gonçalves, E., Bhosle, S.G., Chen, E., Shepherd, R., Beaver, C., Ansari, R., Pooley, R., Wilkinson, P., et al. (2018). Unsupervised correction of gene-independent cell responses to CRISPR-Cas9 targeting. BMC Genom. *19*, 604.
2. Jinek, M., Chylinski, K., Fonfara, I., Hauer, M., Doudna, J.A., and Charpentier, E. (2012). A programmable dual-RNA-guided DNA endonuclease in adaptive bacterial immunity. Science *337*, 816–821.
3. Mali, P., Yang, L., Esvelt, K.M., Aach, J., Guell, M., DiCarlo, J.E., Norville, J.E., and Church, G.M. (2013). RNA-guided human genome engineering via Cas9. Science *339*, 823–826.
4. Cong, L., Ran, F.A., Cox, D., Lin, S., Barretto, R., Habib, N., Hsu, P.D., Wu, X., Jiang, W., Marraffini, L.A., and Zhang, F. (2013). Multiplex genome engineering using CRISPR/Cas systems. Science *339*, 819–823.
5. Shalem, O., Sanjana, N.E., Hartenian, E., Shi, X., Scott, D.A., Mikkelson, T., Heckl, D., Ebert, B.L., Root, D.E., Doench, J.G., and Zhang, F. (2014). Genome-scale CRISPR-Cas9 knockout screening in human cells. Science *343*, 84–87. https://doi.org/10.1126/science.1247005.
6. Behan, F.M., Iorio, F., Picco, G., Gonçalves, E., Beaver, C.M., Migliardi, G., Santos, R., Rao, Y., Sassi, F., Pinnelli, M., et al. (2019). Prioritization of cancer therapeutic targets using CRISPR-Cas9 screens. Nature *568*, 511–516.
7. Tzelepis, K., Koike-Yusa, H., De Braekeleer, E., Li, Y., Metzakopian, E., Dovey, O.M., Mupo, A., Grinkevich, V., Li, M., Mazan, M., et al. (2016). A CRISPR dropout screen identifies genetic vulnerabilities and therapeutic targets in acute myeloid leukemia. Cell Rep. *17*, 1193–1205. https://doi.org/10.1016/j.celrep.2016.09.079.
8. Hart, T., Chandrashekhar, M., Aregger, M., Steinhart, Z., Brown, K.R., MacLeod, G., Mis, M., Zimmermann, M., Fradet-Turcotte, A., Sun, S., et al. (2015). High-resolution CRISPR screens reveal fitness genes and genotype-specific cancer liabilities. Cell *163*, 1515–1526. https://doi.org/10.1016/j.cell.2015.11.015.
9. Wang, T., Wei, J.J., Sabatini, D.M., and Lander, E.S. (2014). Genetic screens in human cells using the CRISPR-Cas9 system. Science *343*, 80–84. https://doi.org/10.1126/science.1246981.
10. Wang, T., Birsoy, K., Hughes, N.W., Krupczak, K.M., Post, Y., Wei, J.J., Lander, E.S., and Sabatini, D.M. (2015). Identification and characterization of essential genes in the human genome. Science *350*, 1096–1101.
11. Tsherniak, A., Vazquez, F., Montgomery, P.G., Weir, B.A., Kryukov, G., Cowley, G.S., Gill, S., Harrington, W.F., Pantel, S., Krill-Burger, J.M., et al. (2017). Defining a cancer dependency map. Cell *170*, 564–576.e16.
12. Dempster, J.M., Rossen, J., Kazachkova, M., Pan, J., Kugener, G., Root, D.E., and Tsherniak, A. (2019). Extracting biological insights from the project achilles genome-scale CRISPR screens in cancer cell lines. Preprint at bioRxiv. https://doi.org/10.1101/720243.
13. Kurata, M., Yamamoto, K., Moriarity, B.S., Kitagawa, M., and Largaespada, D.A. (2018). CRISPR/Cas9 library screening for drug target discovery. J. Hum. Genet. *63*, 179–186.
14. Martinez-Lage, M., Puig-Serra, P., Menendez, P., Torres-Ruiz, R., and Rodriguez-Perales, S. (2018). CRISPR/Cas9 for cancer therapy: hopes and challenges. Biomedicines *6*, 105. https://doi.org/10.3390/biomedicines6040105.
15. Hsu, P.D., Lander, E.S., and Zhang, F. (2014). Development and applications of CRISPR-Cas9 for genome engineering. Cell *157*, 1262–1278.
16. Rouet, P., Smih, F., and Jasin, M. (1994). Introduction of double-strand breaks into the genome of mouse cells by expression of a rare-cutting endonuclease. Mol. Cell Biol. *14*, 8096–8106.
17. Symington, L.S., and Gautier, J. (2011). Double-strand break end resection and repair pathway choice. Annu. Rev. Genet. *45*, 247–271.
18. Panier, S., and Durocher, D. (2013). Push back to respond better: regulatory inhibition of the DNA double-strand break response. Nat. Rev. Mol. Cell Biol. *14*, 661–672.
19. Davis, A.J., and Chen, D.J. (2013). DNA double strand break repair via non-homologous end-joining. Transl. Cancer Res. *2*, 130–143.
20. Gomez, V., and Hergovich, A. (2016). Chapter 14 - cell-cycle control and DNA-damage signaling in mammals. In Genome Stability, I. Kovalchuk and O. Kovalchuk, eds. (Academic Press), pp. 227–242.
21. Shen, M.W., Arbab, M., Hsu, J.Y., Worstell, D., Culbertson, S.J., Krabbe, O., Cassa, C.A., Liu, D.R., Gifford, D.K., and Sherwood, R.I. (2018). Predictable and precise template-free CRISPR editing of pathogenic variants. Nature *563*, 646–651.
22. Aguirre, A.J., Meyers, R.M., Weir, B.A., Vazquez, F., Zhang, C.-Z., Ben-David, U., Cook, A., Ha, G., Harrington, W.F., Doshi, M.B., et al. (2016). Genomic copy number dictates a gene-independent cell response to CRISPR/Cas9 targeting. Cancer Discov. *6*, 914–929.
23. Munoz, D.M., Cassiani, P.J., Li, L., Billy, E., Korn, J.M., Jones, M.D., Golji, J., Ruddy, D.A., Yu, K., McAllister, G., et al. (2016). CRISPR screens provide a comprehensive assessment of cancer vulnerabilities but generate false-positive hits for highly amplified genomic regions. Cancer Discov. *6*, 900–913.
24. Meyers, R.M., Bryan, J.G., McFarland, J.M., Weir, B.A., Sizemore, A.E., Xu, H., Dharia, N.V., Montgomery, P.G., Cowley, G.S., Pantel, S., et al. (2017). Computational correction of copy number effect improves specificity of CRISPR-Cas9 essentiality screens in cancer cells. Nat. Genet. *49*, 1779–1784.
25. Gonçalves, E., Behan, F.M., Louzada, S., Arnol, D., Stronach, E.A., Yang, F., Yusa, K., Stegle, O., Iorio, F., and Garnett, M.J. (2019). Structural rearrangements generate cell-specific, gene-independent CRISPR-Cas9 loss of fitness effects. Genome Biol. *20*, 27.
26. Dempster, J.M., Boyle, I., Vazquez, F., Root, D., Boehm, J.S., Hahn, W.C., Tsherniak, A., and McFarland, J.M. (2021). Chronos: a CRISPR cell population dynamics model. bioRxiv. https://doi.org/10.1101/2021.02.25.432728.
27. de Weck, A., Golji, J., Jones, M.D., Korn, J.M., Billy, E., McDonald, E.R., 3rd, Schmelzle, T., Bitter, H., and Kauffmann, A. (2018). Correction of copy number induced false positives in CRISPR screens. PLoS Comput. Biol. *14*. e1006279.

28. Venkatraman, E.S., and Olshen, A.B. (2007). A faster circular binary segmentation algorithm for the analysis of array CGH data. Bioinformatics *23*, 657–663.

29. Olshen, A.B., Venkatraman, E.S., Lucito, R., and Wigler, M. (2004). Circular binary segmentation for the analysis of array-based DNA copy number data. Biostatistics *5*, 557–572.

30. Pacini, C., Dempster, J.M., Boyle, I., Gonçalves, E., Najgebauer, H., Karakoc, E., van der Meer, D., Barthorpe, A., Lightfoot, H., Jaaks, P., et al. (2021). Integrated cross-study datasets of genetic dependencies in cancer. Nat. Commun. *12*, 1661.

31. Dempster, J.M., Pacini, C., Pantel, S., Behan, F.M., Green, T., Krill-Burger, J., Beaver, C.M., Younger, S.T., Zhivich, V., Najgebauer, H., et al. (2019). Agreement between two large pan-cancer genome-scale CRISPR knock-out datasets. Nat. Commun. *10*, 5817.

32. Li, W., Xu, H., Xiao, T., Cong, L., Love, M.I., Zhang, F., Irizarry, R.A., Liu, J.S., Brown, M., and Liu, X.S. (2014). MAGeCK enables robust identification of essential genes from genome-scale CRISPR/Cas9 knockout screens. Genome Biol. *15*, 554.

33. Maza, E., Frasse, P., Senin, P., Bouzayen, M., and Zouine, M. (2013). Comparison of normalization methods for differential gene expression analysis in RNA-Seq experiments: a matter of relative size of studied transcriptomes. Commun. Integr. Biol. *6*. e25849.

34. Hart, T., Tong, A.H.Y., Chan, K., Van Leeuwen, J., Seetharaman, A., Aregger, M., Chandrashekhar, M., Hustedt, N., Seth, S., Noonan, A., et al. (2017). Evaluation and design of genome-wide CRISPR/SpCas9 knockout screens. G3 *7*, 2719–2727.

35. Hart, T., Brown, K.R., Sircoulomb, F., Rottapel, R., and Moffat, J. (2014). Measuring error rates in genomic perturbation screens: gold standards for human functional genomics. Mol. Syst. Biol. *10*, 733.

36. Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., and Mesirov, J.P. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl. Acad. Sci. USA *102*, 15545–15550.

37. Doench, J.G., Fusi, N., Sullender, M., Hegde, M., Vaimberg, E.W., Donovan, K.F., Smith, I., Tothova, Z., Wilen, C., Orchard, R., et al. (2016). Optimized sgRNA design to maximize activity and minimize off-target effects of CRISPR-Cas9. Nat. Biotechnol. *34*, 184–191. https://doi.org/10.1038/nbt.3437.

38. Sanson, K.R., Hanna, R.E., Hegde, M., Donovan, K.F., Strand, C., Sullender, M.E., Vaimberg, E.W., Goodale, A., Root, D.E., Piccioni, F., and Doench, J.G. (2018). Optimized libraries for CRISPR-Cas9 genetic screens with multiple modalities. Nat. Commun. *9*, 5416. https://doi.org/10.1038/s41467-018-07901-8.

39. Sanjana, N.E., Shalem, O., and Zhang, F. (2014). Improved vectors and genome-wide libraries for CRISPR screening. Nat. Methods *11*, 783–784. https://doi.org/10.1038/nmeth.3047.

40. Park, R.J., Wang, T., Koundakjian, D., Hultquist, J.F., Lamothe-Molina, P., Monel, B., Schumann, K., Yu, H., Krupzcak, K.M., Garcia-Beltran, W., et al. (2017). A genome-wide CRISPR screen identifies a restricted set of HIV host dependency factors. Nat. Genet. *49*, 193–203.

41. Gonçalves, E., Thomas, M., Behan, F.M., Picco, G., Pacini, C., Allen, F., Vinceti, A., Sharma, M., Jackson, D.A., Price, S., et al. (2021). Minimal genome-wide human CRISPR-Cas9 library. Genome Biol. *22*, 40.

42. Iorio, F., Knijnenburg, T.A., Vis, D.J., Bignell, G.R., Menden, M.P., Schubert, M., Aben, N., Gonçalves, E., Barthorpe, S., Lightfoot, H., et al. (2016). A landscape of pharmacogenomic interactions in cancer. Cell *166*, 740–754.

43. Mermel, C.H., Schumacher, S.E., Hill, B., Meyerson, M.L., Beroukhim, R., and Getz, G. (2011). GISTIC2.0 facilitates sensitive and confident localization of the targets of focal somatic copy-number alteration in human cancers. Genome Biol. *12*, R41.

44. Barretina, J., Caponigro, G., Stransky, N., Venkatesan, K., Margolin, A.A., Kim, S., Wilson, C.J., Lehár, J., Kryukov, G.V., Sonkin, D., et al. (2012). The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity. Nature *483*, 603–607.

45. Hart, T., and Moffat, J. (2016). BAGEL: a computational framework for identifying essential genes from pooled library screens. BMC Bioinf. *17*, 164. https://doi.org/10.1186/s12859-016-1015-8.

46. Kim, E., and Hart, T. (2021). Improved analysis of CRISPR fitness screens and reduced off-target effects with the BAGEL2 gene essentiality classifier. Genome Med. *13*, 2.

47. Liao, Y., Smyth, G.K., and Shi, W. (2019). The R package Rsubread is easier, faster, cheaper and better for alignment and quantification of RNA sequencing reads. Nucleic Acids Res. *47*, e47.

48. Lawrence, M., Huber, W., Pagès, H., Aboyoun, P., Carlson, M., Gentleman, R., Morgan, M.T., and Carey, V.J. (2013). Software for computing and annotating genomic ranges. PLoS Comput. Biol. *9*. e1003118.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
HT-29 raw read counts from AVANA library	(Doench et al. 2016)[37]	https://github.com/francescojm/CRISPRcleanR
HT-29 raw read counts from Brunello library	(Sanson et al. 2018)[38]	https://github.com/francescojm/CRISPRcleanR
HT-29 raw read counts from GeCKOv2 library	(Shalem et al. 2014; Sanjana et al. 2014)[5,39]	https://github.com/francescojm/CRISPRcleanR
HT-29 raw read counts from KY library	(Tzelepis et al. 2016; Behan et al. 2019)[6,7]	https://github.com/francescojm/CRISPRcleanR
HT-29 raw read counts from MinLibCas9 library	(Gonçalves et al. 2021)[41]	https://github.com/francescojm/CRISPRcleanR
HT-29 raw read counts from Whitehead library	(Wang et al. 2014; Wang et al. 2015; Park et al. 2017)[9,10,40]	https://github.com/francescojm/CRISPRcleanR
Software and algorithms		
CRISPRcleanR^{WebApp}	This manuscript	https://crispcleanr-webapp.fht.org/(code https://doi.org/10.5281/zenodo.7347952)
CRISPRcleanR (v3.0.0)	This manuscript	https://github.com/francescojm/CRISPRcleanR https://doi.org/10.5281/zenodo.7347915
R (v4.2.1)	The R foundation	https://www.r-project.org/
Other		
Hart2014 reference nonessential gene-set	(Hart et al., 2014)[35]	https://github.com/hart-lab/bagel
Hart2017 reference essential gene-set	(Hart et al., 2017)[34]	https://github.com/hart-lab/bagel
MSigDB independent essential	(Subramanian et al. 2005; Iorio et al. 2018)[1,36]	https://github.com/francescojm/CRISPRcleanR
GDSC amplified	(Iorio et al. 2016)[42]	https://github.com/francescojm/CRISPRcleanR
CCLE amplified	(Mermel et al. 2011)[43]	https://github.com/francescojm/CRISPRcleanR
Amplified non-expressed	(Iorio et al. 2016)[42]	https://github.com/francescojm/CRISPRcleanR

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and analyses should be directed to and will be fulfilled by the lead contact, Francesco Iorio (francesco.iorio@fht.org).

Material availability

This study did not generate new unique reagents.

Data and code availability

- This paper analyzes existing, publicly available data, whose accession numbers are listed in the key resources table.
- CRISPRcleanR^{WebApp} is available at: https://crispcleanr-webapp.fht.org (front-end code https://doi.org/10.5281/zenodo.7347952). The latest version of the original CRISPRcleanR package (v3.0.0) can be found at the following GitHub repository: https://github.com/francescojm/CRISPRcleanR (https://doi.org/10.5281/zenodo.7347915).
- The raw read count files obtained from six screens performed on the HT-29 cell line using different genome-wide CRISPR-Cas9 libraries are available as external data in the CRISPRcleanR package: https://github.com/francescojm/CRISPRcleanR/tree/master/inst/extdata.
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

METHOD DETAILS

Web-application architecture

Front-end

The front-end web application (also known as client) is implemented as a Single Page Application (SPA). Dynamic data is retrieved from a backend API server, returning data in JSON format, which is then used from the SPA to render HTML/CSS accordingly. Example files are made downloadable from the API server, while all job related files are managed through a dedicated file server.

The client is based on Vue.js JavaScript framework (https://vuejs.org/). Interactive charts are implemented as scalable vector graphics documents managed through Vue components leveraging the D3.js library (https://d3js.org/). Design and styling were done entirely from scratch with Sass stylesheets, compiled as CSS during the application bundling process. Screen layout and responsiveness is achieved with Flexbox, CSS Grid layouts and Javascript. The bundled web application is served through a Dockerized (https://www.docker.com/) version of Nginx (https://www.nginx.com/), hosted on a virtual machine within HT IT infrastructure.

Back-end

The back-end consists of a docker multi-container app built upon the following containers and underlying technologies (see also Methods S1):

- Application Programming Interface (API) server: FastAPI python framework (https://fastapi.tiangolo.com/)
- File server: NodeJS with Express.js framework
- Background queue: Celery Python (https://docs.celeryproject.org/en/stable/getting-started/introduction.html)
- Message Broker: Redis (https://redis.io/)
- Database: MongoDB (https://www.mongodb.com/)

An on premises S3 bucket compliant object storage is used to store all job related files (both input and output).

Application Programming Interface

The API server is implemented with FastAPI, a modern and popular Python API framework. It is provided as a Docker container based on a python bullseye official docker image.

File server

A NodeJS server is used to manage the files' upload and download. This allows to directly and reliably stream data files to and from the S3 bucket, since it is not possible with FastAPI. A redis-based pub-sub messaging system exists to ensure the file server informs back the API main server about the upload outcome.

Task queue and message broker

To maintain the backend API server responsive while processing a job, we implemented an offline job processing mechanism. More specifically, job processing is delegated to another container that receives jobs and processes them asynchronously with respect to the actual job submission, following a producer-consumer pattern. When the backend receives a new submitted job, this is forwarded to the background task queue through the message broker to inform consumers about new tasks to be executed.

Background processing is implemented through Celery, a common python task queue manager. Celery is executed on a dedicated Docker container based on a python bullseye official docker image, further customised to perform R processing. This customised image installs a Debian-compatible R distribution, along with all required CRISPRcleanR dependencies. Communication with CRISPRcleanR package is performed through Rpy2 (https://rpy2.github.io/), a python package that enables us to register R functions and environments into python wrapping objects, such that R interactions can be directly executed from python code.

During job processing, the background queue container instantiates a single celery consumer, called worker, that consumes job computation requests from the message queue. In order to obtain parallel processing over multiple jobs, distinct independent workers can be spawned on container replicas by properly configuring the underlying container orchestrator. The message broker is then able to automatically send new job computations according to each worker's current workload, while still avoiding the same job execution to be performed on different workers. The communication between FastAPI backend and task queue is accomplished through a Redis-based message broker, the latter running on a dedicated container.

Database container

Our database container runs a MongoDB instance, a cross-platform document-oriented NoSQL database. In addition, the backend container communicates with MongoDB using a Motor asyncio driver (https://motor.readthedocs.io/en/stable/), whereas the Celery-based background queue uses a Pymongo driver (https://pymongo.readthedocs.io/en/stable/), not being based on asyncio patterns.

Security

CRISPRcleanR*WebApp* implements an authorization schema based on OAuth2 protocol (https://datatracker.ietf.org/doc/html/rfc6749), following the Authorization Code Flow with a public client. A further layer of authentication is provided through OpenID Connect (https://openid.net/connect/), which enables us to transmit user information, such as username and e-mail, to the client application. This highly secure schema allows for great flexibility in configuration and it might easily be extended to implement federated/third party access through external acknowledged identity providers.

The security schema is implemented through Keycloak (https://www.keycloak.org/), an integrated authorization/authentication system which provides a comprehensive solution for managing users and providing an authorization server for managing authentication and applications' tokens. Our hosting infrastructure at Fondazione Human Technopole is equipped with a dedicated Keycloak system made accessible both from the backend (for token validation) and frontend (for issuing tokens).

CRISPRcleanR^{WebApp} management of input files

Computation of read count from FASTQ or BAM files

CRISPRcleanR^{WebApp} accepts trimmed FASTQ as well as BAM file formats as input to derive single-guide RNA (sgRNA) counts. For the FASTQ format, a preliminary quality control is performed on the sequencing data based on the quality scores of the corresponding nucleotide sequences. The sgRNA sequences are then mapped to the library index, which it's generated from the sequences retrieved in the library annotation file using the Rsubread R package.[47] In order to provide the most reliable counts estimation the alignment doesn't allow N bases, mismatches or gaps. All alignment summary statistics are provided in a text file available in the downloadable results. BAM files generated by the alignment step or supplied as input are processed using the GenomicAlignments R package.[48] The occurrence of the "seqnames" of the aligned reads are counted and matched with the sgRNA IDs in the library annotation to provide the counts for each sample. All sample count data are then merged to create a count matrix that is downloadable from the results page and used in the following steps of the pipeline.

Upload of custom genome-wide CRISPR-Cas9 library

In case of upload of a custom genome-wide CRISPR library (i.e. not part of the six built-in libraries of CRISPRcleanR^{WebApp}), the annotation file must include a "seq" field including the sgRNA sequences used in the screening. The application will then convert those sequences in a library index suitable for the alignment and then proceed to evaluate the read counts as described for the standard libraries. The matching has to be exact (i.e. no mismatches allowed), and at least 80% of the guides must be recapitulated to proceed with the workflow of the application and correct the depletion fold-changes for biases.

Validation of the supported CRISPR-Cas9 libraries in the new version of CRISPRcleanR

Data acquisition

We validated CRISPRcleanR (v2.2.1) on six screens obtained from transducing popular CRISPR-Cas9 libraries into the HT-29 cell line. We tested the following libraries: AVANA,[37] Brunello,[38] GeCKOv2,[5,39] KY,[6,7] MiniLibCas9[41] and Whitehead.[9,10,40] All raw read count files are available in the following GitHub repository: https://github.com/francescojm/CRISPRcleanR/tree/master/inst/extdata.

GeCKOv2 library mapping onto the human genome

We mapped the protospacer sequence of each sgRNA in the GeCKOv2 library onto the human genome (GRCh38 - hg38) using the short read mapping method bwa-mem. Most of the reads were mapped uniquely to the reference genome sequence and their positions were found within their targeted genes. On the other hand, the remaining guides were mapped ambiguously and with some mismatches. We remapped them to the reference human genome using first BLATt and then BLASTn. For the guides mapped onto multiple locations, we only considered those included in the targeted genes declared in the original library annotation file. The annotations of the genes were extracted from Gencode v38. Finally, the remaining sequences were mapped to the reference genome with mismatches, and the positions with the minimum mismatches within the targeted genes were selected. All the guides were mapped to a position on the reference genome within their targeted genes.

Comparison of CRISPRcleanR pre- and post-correction logFCs across screens

To compare the differences in log fold-changes (logFCs) pre- and post-correction, we first normalised the sgRNAs' raw read counts by scaling for the total number of reads, after filtering out those guides having a plasmid read count less than 30. This is implemented in the *ccr.NormfoldChanges* function and resulted in uncorrected depletion logFCs. Following the pipeline implemented in https://github.com/francescojm/CRISPRcleanR/blob/master/Quick_start.pdf, we applied the *ccr.GWclean* function with default settings to correct the logFCs of each screen.

We obtained gene-level logFCs by averaging the score of sgRNAs targeting the same gene, and considering only the genes in common across the six screens, leading to 8473 genes. In order to reduce technical variability due to the usage of different libraries, we applied for each screen a min-max normalisation on the concatenated vector of pre- and post-corrected gene-level logFCs. We then split back the vector in pre- and post-corrected logFCs, and computed gene-wise logFC differences across screens.

Gene-wise correction agreement across screens

In information theory, Shannon entropy is used to quantify the uncertainty of a random variable. It is formally defined as the average auto information associated with each outcome of the random variable. Efficiency (or normalised Shannon Entropy) is defined as the Shannon entropy divided by the maximum value that the entropy can assume (which depends on the size of the spectrum of the random variable under consideration). In our case: the lower the efficiency the higher the homogeneity in the CRISPRcleanR outcomes across screens. Thus, a low efficiency score means a high agreement of correction outcomes across the screens. We computed gene-wise normalised Shannon entropy across screens when considering the three possible correction outcomes outputted by the *ccr.GWclean* function.

We considered only the genes in common across the screens, leading to 8473 genes, and removed those found at the conjunction of two segments, totalling to 7923 genes. Then, we computed the gene-wise normalised Shannon entropy H_n, following the Equation 1:

$$H_n = \frac{H_g}{H_{\max}} = \frac{-\sum_{i=1}^{c} p(g_i) \cdot \log_2(p(g_i))}{\log_2(C)} \qquad \text{(Equation 1)}$$

where H_g is the Shannon entropy of a gene g, computed as the sum over the variable's probability values, $g^i \in \{-1, 0, 1\}$ and C is the total number of possible correction outcomes (i.e. 3 in this case), divided by the maximum expected entropy (log_2 of C).

Recall of genes in six predefined gene-sets following CRISPRcleanR correction

We assessed the effect of CRISPRcleanR correction on six predefined sets of genes, namely Hart2017 essential, Hart2014 nonessential, independent sets of essential genes derived from the MSigDB, copy number amplified genes from GDSC dataset, copy number amplified genes from CCLE dataset, and copy number amplified genes from GDSC that were not expressed (FPKM <0.1 in HT-29).

For each screen, we computed the essentiality threshold at 5% false discovery rate (FDR) for the pre- and post-corrected screens, using the Hart2017 essential and Hart2014 nonessential as source of reference essential (EG) and nonessential genes (NEG), respectively. In particular, we ranked the gene-level depletion logFCs of the EG and NEG in increasing order. For each rank position i, we calculated a set of predicted fitness genes (PFG) as follows:

$$PFG(i) = \{g \in EG \cup NEG | rank(FC_g) \geq i\}, \qquad \text{(Equation 2)}$$

where $rank(FC_g)$ is the corresponding rank position of gene g in the reference gene set based on its depletion logFC. The ranked list is then used to calculate positive predictive values (PPV) for the i^{th} rank position as follows:

$$PPV(i) = |PFG(i) \cap EG| / |PFG(i)|. \qquad \text{(Equation 3)}$$

We determined the highest threshold of depletion logFC (logFC*) in rank position i^* such that $PPV(i^*) \geq 0.95$, which is equivalent to an FDR of 5%. We considered all genes with a logFC < logFC* as essential for the viability of HT-29. For each gene-set S, we computed the recall by dividing the size of genes g_s below the 5% FDR threshold (logFC*) by the total size of g_s included in the screen, according to (4):

$$\text{Recall} = \frac{|g_s < \log_2 FC^*|}{|g_s|} \qquad \text{(Equation 4)}$$