PEGG: A computational pipeline for rapid design of prime editing guide RNAs and sensor libraries

Samuel I. Gould[1,2], Francisco J. Sánchez-Rivera[1,2,#]

[1]Department of Biology, Massachusetts Institute of Technology, Cambridge, 02142, Massachusetts, USA.

[2]David H. Koch Institute for Integrative Cancer Research, Massachusetts Institute of Technology, Cambridge, 02142, Massachusetts, USA.

[#]Correspondence: fsr@mit.edu

Abstract

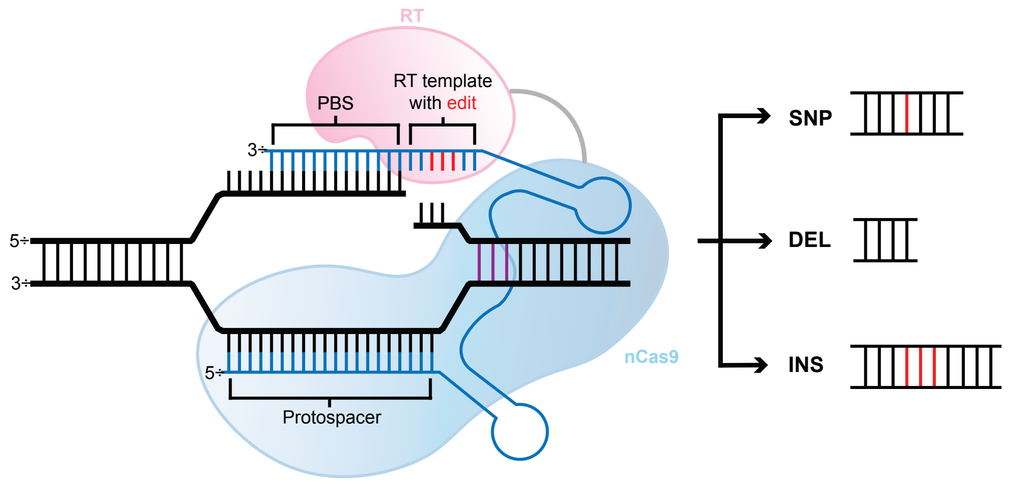
Prime editing leverages Cas proteins fused to reverse transcription enzymes and prime editing guide RNAs (pegRNAs) to programmably engineer diverse genetic alterations without DNA double strand breaks or exogenous donor templates. Prime editing is powerful, but the process of pegRNA design remains a challenge that limits its widespread adoption by the community. Although a number of computational pegRNA design tools have been developed, their capability for rapid and systematic design of pegRNAs for variant engineering and high-throughput genetic screens is limited. Here, we describe Prime Editing Guide Generator (PEGG): a computational pipeline for rapid design of pegRNAs and paired 'sensor' pegRNA libraries. PEGG is a user-friendly Python package that generates and visualizes pegRNA designs for installing desired mutations while integrating user-defined properties like variable length of primer binding site and reverse transcription template regions, and outputs ready-to-order pegRNA oligos and libraries containing optimized sequence adapters for pooled cloning. PEGG (including documentation) can be accessed at **https://pegg.readthedocs.io**.

Main text

Prime editing (PE) is a promising technology for engineering essentially any type of genetic alteration across various organisms and mammalian cell types[1] but designing, testing, and optimizing prime editing guide RNAs (pegRNAs) remains a major challenge. While the functional architecture of standard CRISPR nuclease[2,3], interference[4], activation[5,6], and base editing[7] single guide RNAs (sgRNAs) consists of a simple 20 nucleotide (nt) long sequence, pegRNAs are significantly more complex due to an additional 3' extension region[1,8,9]. This region contains a primer binding site (PBS), which is bound by an exposed DNA flap that contains a 3' OH group and serves as a primer for reverse transcription and extension of a reverse transcription template (RTT) that encodes the mutation of interest[10]. The PBS typically varies in length from 8-15 nt while the RTT typically varies from 10-74 nt **(Figure 1a, Extended Figure 1)**. Given these inherent structural complexities, pegRNA design is significantly more difficult relative to sgRNAs, particularly since many pegRNAs need to be tested to engineer a single mutation of interest with reasonable efficiency and precision[10]. The process becomes exponentially more complicated when attempting to design pegRNAs to engineer dozens, hundreds, or even thousands of mutations for high-throughput screening applications[11].

A number of computational tools for pegRNA design have been developed[12–19], but their use for rapid and systematic design of diverse pegRNAs for single, multiplexed, or high-throughput engineering of genetic variants remains limited. Most existing pegRNA design tools are also web-based applications with limited customization capabilities and throughput[12–17]. While these applications are very well-suited for designing a handful of pegRNAs, they are often not capable of handling tens of thousands of pegRNA designs at once, which is needed to construct pooled pegRNA libraries for high-throughput screening. The few locally installed software solutions for pegRNA design that can handle higher throughput have limited documentation, are difficult to use, and do not include many of the features required to take full advantage of the capabilities of PE[18,19] **(Table 1)**.

With this challenge in mind, we reasoned that the ideal pegRNA design tool would allow users to simply and rapidly generate pegRNA designs while also including extensive customization and filtration options, visualizations of pegRNA designs, automated oligonucleotide designs for rapid ordering, and the ability to design large pools of thousands of pegRNAs for high-throughput screening purposes. To this end, we a



b

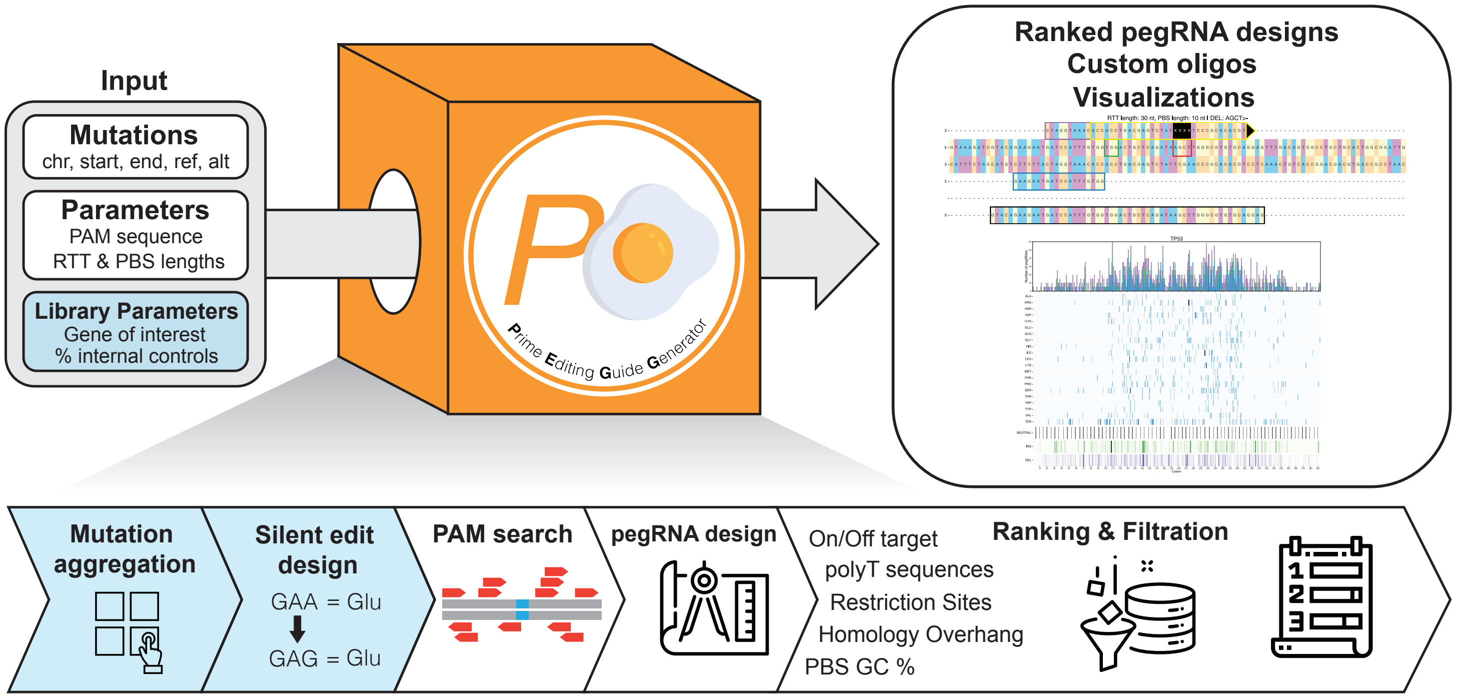


Figure 1. PEGG: An end-to-end solution for high throughput pegRNA design. a. Schematic of prime editing. A prime editor consists of a nicking Cas9 (nCas9), shown in blue, fused to a reverse transcriptase (RT), shown in pink. This editing machinery is brought to a particular locus by the pegRNA (blue), which consists of a protospacer, a primer binding sequence (PBS) and reverse transcriptase template (RTT) that contains the edit of interest. Prime editing is capable of producing any SNP as well as small deletions and insertions. PAM sequence shown in purple. Genomic DNA in black. **b.** PEGG takes in a list of input mutations and parameters and outputs a list of ranked pegRNA designs, as well as custom oligonucleotide designs and built-in visualizations. For a given input mutation, PEGG searches for PAM sequences, generates potential pegRNA designs, and ranks those designs according to their properties by creating a composite PEGG score. PEGG also includes automated library design functionality that aggregates mutations in a gene of interest and produces silent substitutions as internal controls at a desired frequency.

developed Prime Editing Guide Generator (PEGG), an open-source Python package available through the Python Package Index, that provides an end-to-end solution for pegRNA design **(Figure 1b, Extended Figure 2)**. PEGG allows users to easily generate pegRNA designs based on a list of input mutations and parameters. Moreover, PEGG only requires a small amount of information about input mutations: (1) their chromosomal location, and (2) the reference and mutant alleles. The simplicity of the mutation input format allows PEGG to directly query the evolving information available within the vast array of clinical datasets available on the cBioPortal without any modifications[20]. PEGG also includes the option to input a list of ClinVar variation IDs, greatly simplifying the process of modeling allelic variants[21].

Once provided with a set of mutations, PEGG searches for eligible PAM sequences for each mutation and generates pegRNA designs based on either recommended[1,8–10,15,22,23] or user-defined RTT and PBS lengths **(Figure 1b, Extended Figure 2).** These pegRNA designs are then filtered and ranked according to their properties, which include: (1) on- and off-target scores of a given protospacer[24,25], (2) length of the homology overhang in the RTT, (3) polyT termination sequences, (4) presence of certain restriction sites for cloning (optional), and (5) PBS GC content. These properties are added together to form a composite 'PEGG score' (see **Methods**) that is used to rank the pegRNAs. The properties of each pegRNA are also individually returned, allowing users to filter and rank pegRNAs based on these properties, rather than exclusively relying on PEGG's ranking algorithm. PEGG also returns a 100 bp long DNA region containing the target edit for visual sequence alignment. Future versions of PEGG will include ranking algorithms based on experimental datasets currently being generated in our laboratory.

PEGG also automatically designs ready-to-order oligonucleotides with optimized 3' and 5' sequence adapters for cloning into pegRNA and PE sensor lentiviral backbones (see **Methods**). Users can choose default adapter and scaffold sequences, or provide their own. A unique feature of PEGG is the option to include a 'sensor' region—a 60 nt synthetic version of the target site—in these oligos[22,23,26] **(Extended Figure 3a-b)**. This design makes it possible to 1) measure editing outcomes on a single or multiplexed pegRNA basis using next-generation DNA sequencing (NGS), 2) use those measurements as a proxy of editing outcomes at target endogenous loci, and 3) read-out and correlate pegRNA enrichment or depletion over time in the context of high-throughput genetic screens[22,23,26] **(Extended Figure 3a-b)**.

In addition, PEGG includes the feature of automatically generating partial PE saturation mutagenesis libraries for a particular gene or genes of interest. Here, users can specify their gene(s) of interest and PEGG then aggregates mutations occurring in the gene and designs silent substitution mutations as internal controls at a desired frequency **(Figure 1b, Extended Figure 4).**

PEGG also allows users to visualize individual pegRNAs and libraries of pegRNAs **(Figure 2b, Figure 3b),** making the process of pegRNA design significantly more transparent.

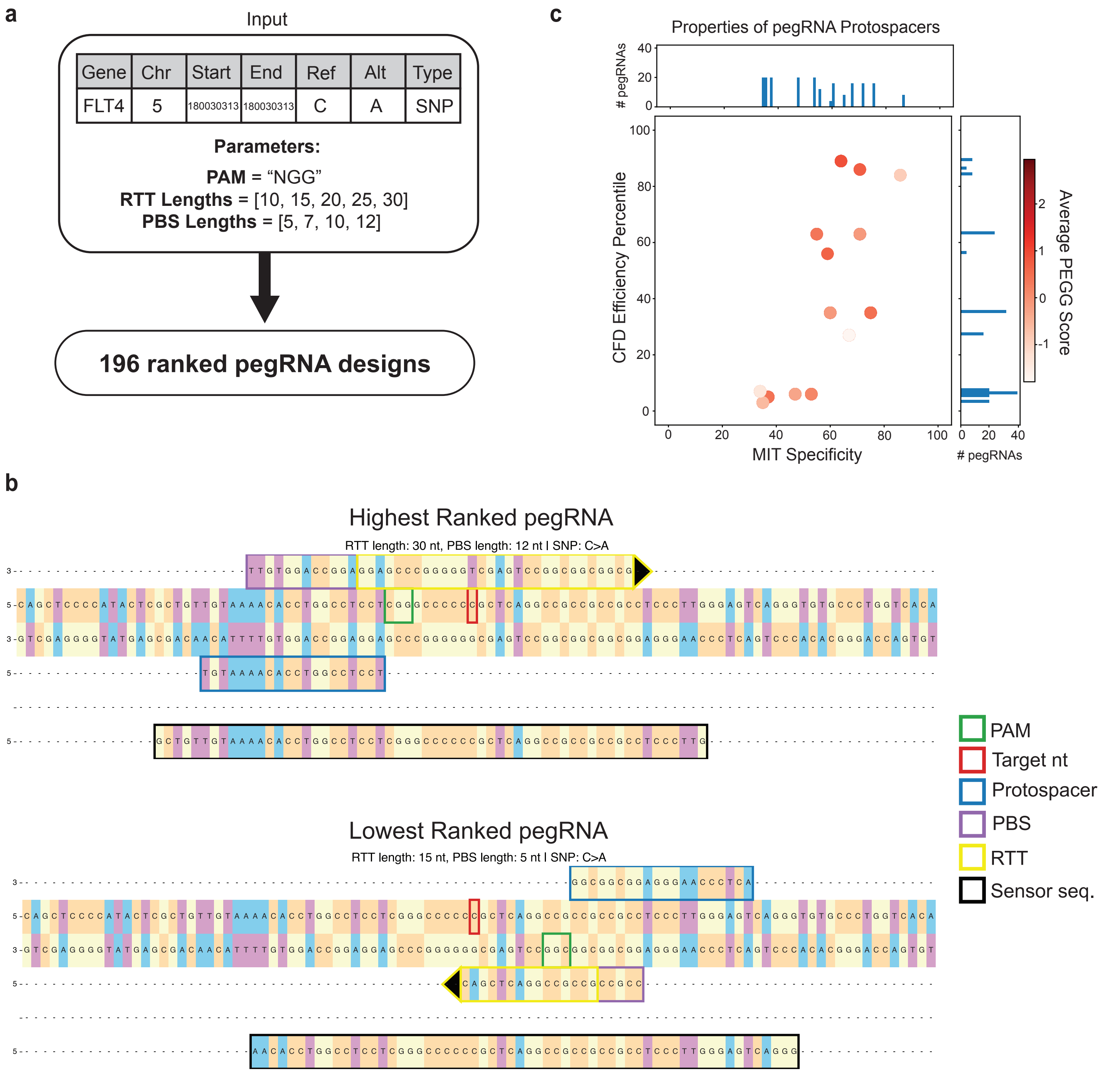
Importantly, PEGG is highly documented (**https://pegg.readthedocs.io**), with extensive tutorials and quickstart guides that allow users to start designing pegRNAs in a matter of minutes. Below are some representative case studies that serve as examples of the utility of PEGG for various types of pegRNA designs.

Case Study 1: Optimization of pegRNA design

Using prime editing to engineer a particular mutation often requires extensive optimization due to the inherent sequence and variable structural complexities of pegRNAs, as well as the wide range of editing efficiencies that have been reported to date[1,9,10,22]. This raises a key practical question—*How can we rapidly design and test various types of pegRNAs to engineer and study a handful of mutations?*

To demonstrate the utility of PEGG, we performed a case study on pegRNA design optimization, using a single nucleotide polymorphism (SNP) in the *FLT4* gene as an example. Generating many potential pegRNA designs to engineer this mutation only required three inputs: 1) genomic coordinates of the mutation, 2) reference and mutant alleles, and 3) structural pegRNA design parameters **(Figure 2a)**. In this example, we asked PEGG to design pegRNAs with a “NGG” PAM sequence, PBS lengths from 5 to 12 nt, and RTT lengths from 10 to 30 nt, resulting in the output of 196 ranked pegRNA designs **(Figure 2a)**.

We visualized the highest and lowest ranked pegRNAs using PEGG’s built-in visualization tools **(Figure 2b)**. It was clear from this visualization that the lowest ranked pegRNA is likely poorly efficient due to having a single nt of homology overhang in the 5’ region of the RTT and a high GC content in the PBS. These visualizations also show a 60 nt sensor region that can be included in oligo designs. Using a sensor-based oligo design here


a
Input
Parameters:
PAM = "NGG"
RTT Lengths = [10, 15, 20, 25, 30]
PBS Lengths = [5, 7, 10, 12]
196 ranked pegRNA designs
c
Properties of pegRNA Protospacers
# pegRNAs
CFD Efficiency Percentile
MIT Specificity
Average PEGG Score
b
Highest Ranked pegRNA
RTT length: 30 nt, PBS length: 12 nt I SNP: C>A
3 - TTGTGGACCGGAGGAGCCCGGGGGTCGAGTCCGGCGGCGGCG
5 - CAGCTCCCCATACTCGCTGTTGTAAAACACCTGGCCTCCTCGGGCCCCCCGCTCAGGCCGCCGCCGCCTCCCTTGGGAGTCAGGGTGTGCCCTGGTCACA
3 - GTCGAGGGGTATGAGCGACAACATTTTGTGGACCGGAGGAGCCCGGGGGGCGAGTCCGGCGGCGGCGGAGGGAACCCTCAGTCCCACACGGGACCAGTGT
5 - TGTAAAACACCTGGCCTCCT
5 - GCTGTTGTAAAACACCTGGCCTCCTCGGGCCCCCCGCTCAGGCCGCCGCCGCCTCCCTTG
Lowest Ranked pegRNA
RTT length: 15 nt, PBS length: 5 nt I SNP: C>A
3 - GGCGGCGGAGGGAACCCTCA
5 - CAGCTCCCCATACTCGCTGTTGTAAAACACCTGGCCTCCTCGGGCCCCCCGCTCAGGCCGCCGCCGCCTCCCTTGGGAGTCAGGGTGTGCCCTGGTCACA
3 - GTCGAGGGGTATGAGCGACAACATTTTGTGGACCGGAGGAGCCCGGGGGGCGAGTCCGGCGGCGGCGGAGGGAACCCTCAGTCCCACACGGGACCAGTGT
5 - CAGCTCAGGCCGCCGCCGCC
5 - AACACCTGGCCTCCTCGGGCCCCCCGCTCAGGCCGCCGCCGCCTCCCTTGGGAGTCAGGG
PAM
Target nt
Protospacer
PBS
RTT
Sensor seq.

Gene	Chr	Start	End	Ref	Alt	Type
FLT4	5	180030313	180030313	C	A	SNP

Figure 2. Case Study 1: Rapid Optimization of pegRNA design. a. Input required by PEGG. **b.** Built-in visualization tool visualizes highest and lowest ranked pegRNAs for engineering the mutation. **c.** On- & off-target scores of protospacers selected by PEGG, colored by average PEGG score (used for ranking).

would allow for pooled pegRNA optimization experiments by linking pegRNA design to editing outcomes at a synthetic version of the endogenous locus[22,23,26].

Among the 196 ranked pegRNA designs, 14 distinct protospacers with a range of on/off target scores are represented **(Figure 2c)**[24,25]. While serving as a useful proxy for Cas9 sgRNA binding and cutting[10], these on/off target scores are not the only factors affecting the composite scores used for ranking pegRNAs **(Figure 2c)**, and are not perfect predictors of pegRNA efficiency[22]. Thus, iterative testing of different RTT and PBS length combinations at different protospacers is essential to identify high efficiency pegRNAs. PEGG returns pegRNA designs for each of the combinations of RTT and PBS length, though fewer designs can accommodate shorter RTT lengths due to the distance between potential PAM sites and the target mutation. pegRNAs with longer RTT lengths tend to have longer regions of homology overhang at their 5' ends, which is also the case here, but longer RTTs can be less efficient depending on the context[10,22].

Altogether, PEGG can rapidly design a large number of pegRNAs with diverse properties, allowing users to quickly optimize their pegRNAs to efficiently engineer their mutations of interest, especially when taking advantage of the sensor-based approaches provided by PEGG.

Case Study 2: Rapid design of pegRNA (sensor) libraries

We performed a second case study to illustrate the high throughput power of PEGG's library design features. If provided with a set of mutations, PEGG can automatically aggregate unique mutations on a per-gene basis, and design a desired number of silent substitution mutations to act as internal controls.

As an example, we analyzed mutation data from >40,000 cancer patients[27] to design a library of pegRNAs to engineer mutations in *TP53*, the most frequently mutated gene in human cancer[28]. PEGG automatically aggregated unique *TP53* mutations observed in this dataset, which we further filtered to only focus on insertions and deletions occurring in at least 3 patients. We also chose to include pegRNAs for silent substitution mutations in *TP53* (7.5% of total library) to use as internal controls, which were automatically designed by PEGG.

In total, our library contained more than 1,200 mutations **(Figure 3a, Extended Figure 4a)**. PEGG used these mutations to design over 28,000 pegRNAs **(Figure 3b)**, with RTT lengths ranging from 10 to 30 nt, and PBS

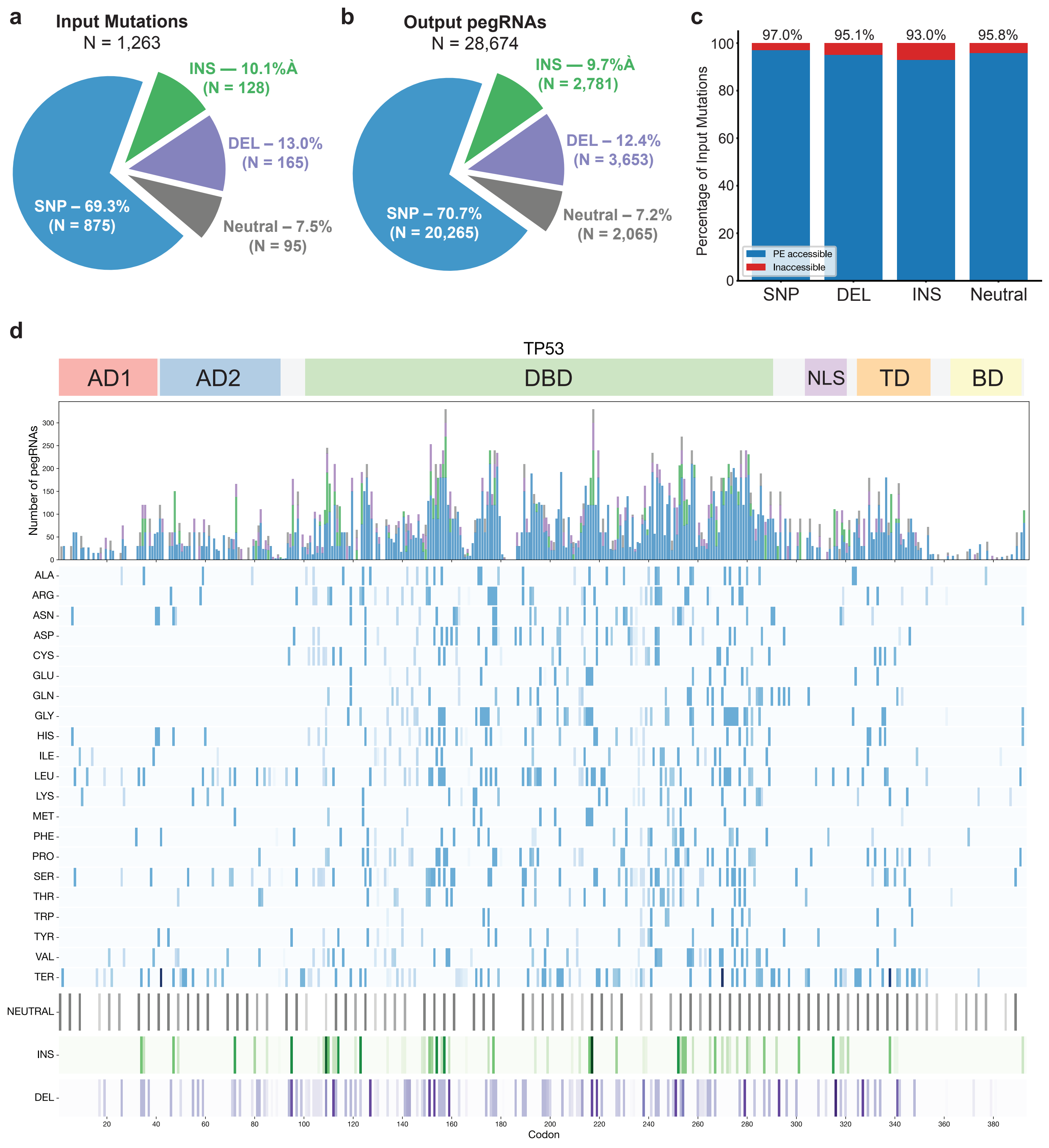


Figure 3. Case Study 2: Automated design of pegRNA (sensor) libraries. a. Input of unique mutations in TP53 observed in >40,000 patients[27]. Neutral, silent substitutions are automatically designed. **b.** Breakdown of pegRNAs designed to engineer input mutations. **c.** PE is able to model nearly all of the input mutations. **d.** Visualization (generated using PEGG's built-in visualization tool) of pegRNAs generated to engineer amino acid substitutions. Coloring refers to # of pegRNAs designed to install a particular mutation.

lengths ranging from 10 to 15 nt **(Extended Figure 5a-d)**. Our goal here was to include 30 distinct pegRNA designs for each mutation in order to maximize the representation of each alteration and produce experimental data that could serve as a training dataset to better predict pegRNA efficiency. We also performed additional filtration on the pegRNAs to minimize off-target editing by requiring a minimum MIT specificity score of 50 per protospacer **(Extended Figure 5e-f)**.

Using the above parameters, PEGG was able to design pegRNAs targeting almost all of the input mutations, regardless of variant type **(Figure 3c)**. The full spectrum of mutations can also be visualized using one of PEGG's built-in visualization tools, which shows specific amino acid changes represented, as well as the locations of insertions, deletions, and tiled neutral substitutions **(Figure 3d)**. Around 10% of all of the possible codon substitutions are represented in this partial saturation mutagenesis library **(Extended Figure 4b)**.

Ongoing projects in our laboratory are using these types of libraries to understand how genetic variation shapes disease outcomes in cancer. Given these are sensor libraries, functional editing outcomes can be coupled with enrichment and depletion patterns of specific pegRNAs in a massively parallel fashion to quantitatively assess variant fitness in a multiplexed setting[26].

Case Study 3: Rapid design of pegRNAs to model pathogenic variants from ClinVar

Many diseases with a strong genetic component, such as cancer, are often associated with specific DNA alterations in the form of SNPs. While some of these genetic variants are strongly associated with pathogenicity, many are often classified as variants of unknown significance (VUS)[29]. Determining whether these VUS are *bona fide* pathogenic variants is critical to understand and better treat cancer and other genetic diseases. An important step towards that goal is to experimentally model these genetic alterations in cellular or animal models with high efficiency and precision. With this goal in mind, we performed a third case study using *MSH6* — a gene that encodes for a DNA mismatch repair protein (MMR) that is mutated in certain individuals and associated with Lynch syndrome and cancer predisposition[30,31].

Using the ClinVar database[21], we selected >5,600 short genetic lesions (SNPs; insertions and deletions > 50bp) identified in *MSH6*. Most of these variants are SNPs, and the vast majority of them are found in the exonic regions of *MSH6* **(Figure 4a-b)**. In addition, though some of these MSH6 variants have been clinically

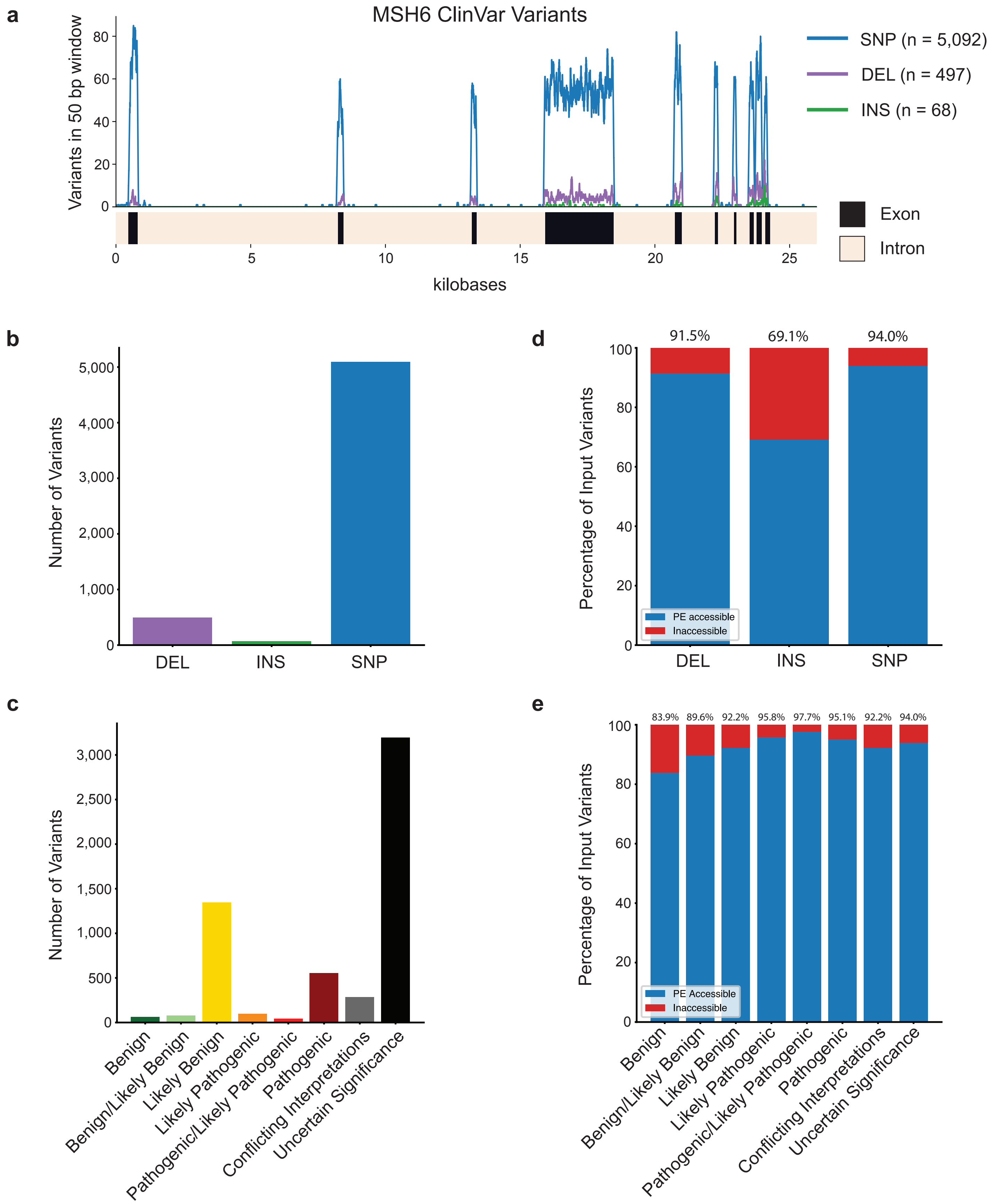


Figure 4. Case Study 3: Rapid design of pegRNAs to model pathogenic variants from ClinVar. a. Localization of >5,000 MSH6 ClinVar variants, segmented by variant type. **b.** Variant classification of MSH6 ClinVar variants. SNP, single nucleotide polymorphism; INS, insertion; DEL, deletion. **c.** Clinical classification of MSH6 ClinVar variants. **d.** Percentage of variants accessible to PE using a 30 nt RTT and 'NGG' PAM, stratified by variant type. **e.** Percentage of variants accessible to PE, using a 30 nt RTT and 'NGG' PAM, stratified by clinical classification.

classified, >50% of the variants are still considered VUS, highlighting the need for functional studies that test the activity of these unclassified variants **(Figure 4c)**.

PEGG includes the option to input a list of ClinVar Variation IDs, which can be easily downloaded from the ClinVar website. This feature allowed us to rapidly design pegRNA libraries targeting ClinVar annotated mutations by simply providing PEGG with a list of the corresponding Variation IDs. PEGG then automatically generated an input table in the correct format that also included all of the clinical information provided by ClinVar. Using this input table, we were able to generate pegRNAs for well over 90% of the mutations using an RTT length of 30 nt **(Figure 4d-e)**. While only ~70% of the insertions were amenable to prime editing due to their relatively large size, insertions in general made up a very small percentage of the input mutations **(Figure 4b,d)**. Remarkably, 94% of the VUS were amenable to prime editing **(Figure 4e)**. Overall, PEGG enables users to rapidly and easily design pegRNAs to engineer variants in the ClinVar database. Future versions of PEGG also plan to include corrective pegRNA designs that restore the wild type sequence of pathogenic and VUS identified in ClinVar.

Discussion

PE is a transformative genome editing technology that, in principle, allows for precise and efficient engineering of any type of genetic alteration. While powerful, PE is still a relatively new technology that remains technically challenging to deploy in the laboratory. One of the challenges relates to the complexities associated with design and testing of pegRNAs for specific mutations, particularly when scaling up these approaches for high-throughput precision genome editing screens. PEGG fulfills these and other challenges by allowing users to rapidly and easily design pegRNAs for specific mutations of interest, as well as PE sensor[22,23,26] constructs that exploit synthetic target sequences to measure editing efficiency and precision at scale.

Supplementary Note

PEGG is an evolving software package, with plans for future improvements and updates. We plan to use training data generated from ongoing pooled screening experiments to create a data-based prediction algorithm for pegRNA efficiency. Future versions of PEGG will offer the option to include silent, mismatch repair (MMR)-evading mutations in the RTT of pegRNAs[23]. We also plan to enable the ability to model human-to-mouse orthologous mutations, allowing for more facile generation of mouse models of human mutations.

Methods

Code Availability/Installation, Documentation & Tutorials

PEGG is an open-source python package. The code is available at the following github repository: https://github.com/samgould2/PEGG. The code is also available through the python package index (the preferred method for installation of PEGG): https://pypi.org/project/pegg/. Extensive documentation and tutorials are available at the documentation website: https://pegg.readthedocs.io. All reference files can be accessed at the following dropbox link: https://tinyurl.com/2p8pv2dc.

The methods below provide a stable set of documentation, but **it is recommended that users refer to the documentation website for more thorough and interactive directions for using PEGG**, as well as keep updated on future versions of PEGG.

Installation

PEGG is available through the python package index. To install, use pip, with the following action in the command line: **“pip install pegg”.**

Required Input Files

In order to use PEGG, a minimum of 2 input files are required: (1) a pandas DataFrame containing a list of input mutations, and (2) a reference genome. An optional, but highly recommended input file for use with the reference genome build GrCh37 is **“chrom_dict,”** which provides on/off target scores for each protospacer occuring in exonic regions of GrCh37 for ~600 cancer-associated genes. All of these reference files, as well as a sample mutation dataset, are available in the reference files dropbox.

Reference File Formatting

PEGG requires that the reference files are in a specific format (see below).

Input Mutations

Input mutations must be loaded into a pandas DataFrame containing at minimum seven column names and their associated information:

1. **Hugo_Symbol:** The gene name of a mutation.
2. **Chromosome:** The chromosome on which a mutation falls (integers 1-22, 'X', or 'Y').
3. **Start_Position:** The genomic start position of a mutation.
4. **End_Position:** The genomic end position of a mutation.
5. **Reference_Allele:** The reference allele (on the + strand).
6. **Tumor_Seq_Allele2:** The mutant allele (on the + strand).
7. **Variant_Type:** The variant type of the mutant (options: 'SNP', 'INS', 'DEL', 'ONP')

All of the datasets on the cBioPortal (http://www.cbioportal.org/datasets)[20] are compatible as they already contain these column headers. Thus, they can be directly loaded into a pandas DataFrame and read as input by PEGG. Alternatively, a list of ClinVar Variation IDs can be provided to the **pegg.clinvar_VCF_translator()** function, along with a clinvar.vcf.gz file, and this function will produce a pandas DataFrame in the correct format for interpretation by PEGG. ClinVar Variation IDs can be directly downloaded or viewed from the ClinVar website, and clinvar.vcf.gz files are accessible at the following link (https://ftp.ncbi.nlm.nih.gov/pub/clinvar/), or in the reference files dropbox.

Reference Genome

PEGG contains a built-in function for formatting reference genomes properly. To use this function, users can run: "records, index_list = **pegg.genome_loader(filepath)**", where filepath is the path to the reference genome .fna.gz file. Reference genome assemblies GrCh37 and GrCh38 are provided in the reference files dropbox link. For using alternative genomes, refer to the documentation website.

On/Off Target Reference File

The last reference file, **"chrom_dict,"** provides on/off target scores for protospacers in exonic regions of GrCh37 for ~600 cancer-associated genes. These scores are used in the ranking and filtration of pegRNAs. To load this file, users should use the pandas.read_pickle() function. If users do not want to include this reference file, simply set "chrom_dict = 'none'".

While this is currently only available for GrCh37, an equivalent file for GrCh38 will be provided shortly and made available in the reference files dropbox. Future versions of PEGG will expand this on/off target scoring system.

Generating pegRNAs with PEGG

With the reference files in the correct format, and PEGG imported as a module into the local environment, using PEGG is simple. Users simply need to specify: (1) which mutations within their mutation dataset to generate pegRNAs for, by providing a list of indices that correspond to the desired mutations, (2) the PAM sequence (e.g. 'NGG'), (3) the desired number of pegRNAs to return per input mutation, and (4) a list of RTT and PBS lengths to search over. With regard to (4), **it is recommended that users search for PBS lengths between 5-15 nt (max = 17 nt), and RTT lengths between 5-30 nt** (very long RTT lengths will likely have decreased efficiency). This information, along with the reference files in the proper format (as described above), are provided to the **pegg.run()** function, and PEGG returns a DataFrame containing a ranked table of pegRNAs and their associated properties. For full syntax, and other options, refer to the documentation website.

Oligonucleotide Generation

To automatically generate oligonucleotides that contain the pegRNAs designed using PEGG, the **pegg.oligo_generator()** function provides multiple options, and produces both a pegRNA oligo and an epegRNA oligo (with a 3' structural motif, tevopreQ1)[9]. A unique feature of PEGG is the option to include a sensor region in the oligo, as described above **(Extended Figure 3b)**. To include a sensor region, set "sensor = True" within pegg.oligo_generator(). To exclude a sensor region, set "sensor = False". Additionally, users need to specify whether they want to append a 'G' nucleotide to the beginning of the protospacer. This is recommended in the original Anzalone et al., 2019 prime editing paper[1]. To include an appended 'G', set "append_proto_g = True/False" within the pegg.oligo_generator() function. It is also widely established that sgRNAs need to contain a 5' G for efficient transcription from U6 promoters[10,11].

Users can also specify which 3' and 5' adapter sequences they want to use, or simply leave these options blank and use the built-in adapters provided by the authors. The built-in adapters contain an Esp3I site in the 3' adapter, and an EcoRI site in the 5' adapter for restriction enzyme-based cloning into pegRNA and sensor backbones (plasmids will be made available through Addgene)[26]. In addition, users can specify to use a different sgRNA scaffold, or use the default option of the sgRNA scaffold provided by the authors[32].

Running pegg.oligo_generator() returns a dataframe that has the oligos appended as columns ('pegRNA_oligo' and 'epegRNA_tevopreQ1_oligo' are the column names).

Library Generation

PEGG also includes an automated library generation function, **pegg.libary_input_generator()**. This function automatically selects all of the mutations associated with a particular gene, generates neutral silent substitutions as internal controls at a desired frequency, and returns a DataFrame that can be fed to **pegg.run()** to generate pegRNA libraries. To use this function, users provide a mutation dataset, specify the gene whose mutations they want to aggregate, and input the desired fraction of silent substitution control mutations. Users also need to provide a list of the coding sequence locations of the relevant transcript for the gene selected to allow PEGG to generate neutral mutations. While this can be done manually, the in-depth tutorial on the documentation website shows how this can be automated using genome annotation files. Once this library input is generated, it can simply be fed into the pegg.run() function as described previously.

Users can also generate neutral, silent substitutions in a given gene separately using the **pegg.neutral_substitutions()** function. Again, for full syntax, refer to the documentation website.

Visualization

PEGG has four built-in visualization functions, two for visualizing individual pegRNAs, and two for visualizing libraries of pegRNAs. To visualize individual pegRNAs, users can make use of **pegg.pegRNA_display(),** which was used to generate the diagrams in Figure 2b, or **pegg.align_display()**, which provides an alignment between the 3' extension of a pegRNA and the genomic DNA. To visualize libraries of pegRNAs, users can use **pegg.lollipop_library(),** which does not require any additional information, or **pegg.matrix_rep_library()**, which requires that users add in HGVSp information about mutations. The latter function was used to generate Figure 3d. For precise syntax for using these visualization tools, please refer to the documentation website.

Acknowledgments

We thank Nicolas Mathey-Andrews, Zackery A. Ely, Tyler Jacks, Edward Kastenhuber, Lukas Dow, Varun Narendra, and Luke Koblan for discussions, as well as everyone in the Sánchez-Rivera laboratory for support. Work in the Sánchez-Rivera laboratory is supported by HHMI (Hanna Gray Fellowship), the V Foundation for Cancer Research (V2022-028), a NCI Cancer Center Support Grant (P30-CA1405), the Ludwig Center at MIT, Koch Institute Frontier Awards, and the MIT Research Support Committee. S.I.G. was also supported by T32GM136540. We also thank the Koch Institute Swanson Biotechnology Center for technical support.

Author contributions

S.I.G. and F.J.S.R. conceived of the study. S.I.G. designed and implemented PEGG. S.I.G. and F.J.S.R. wrote the manuscript.

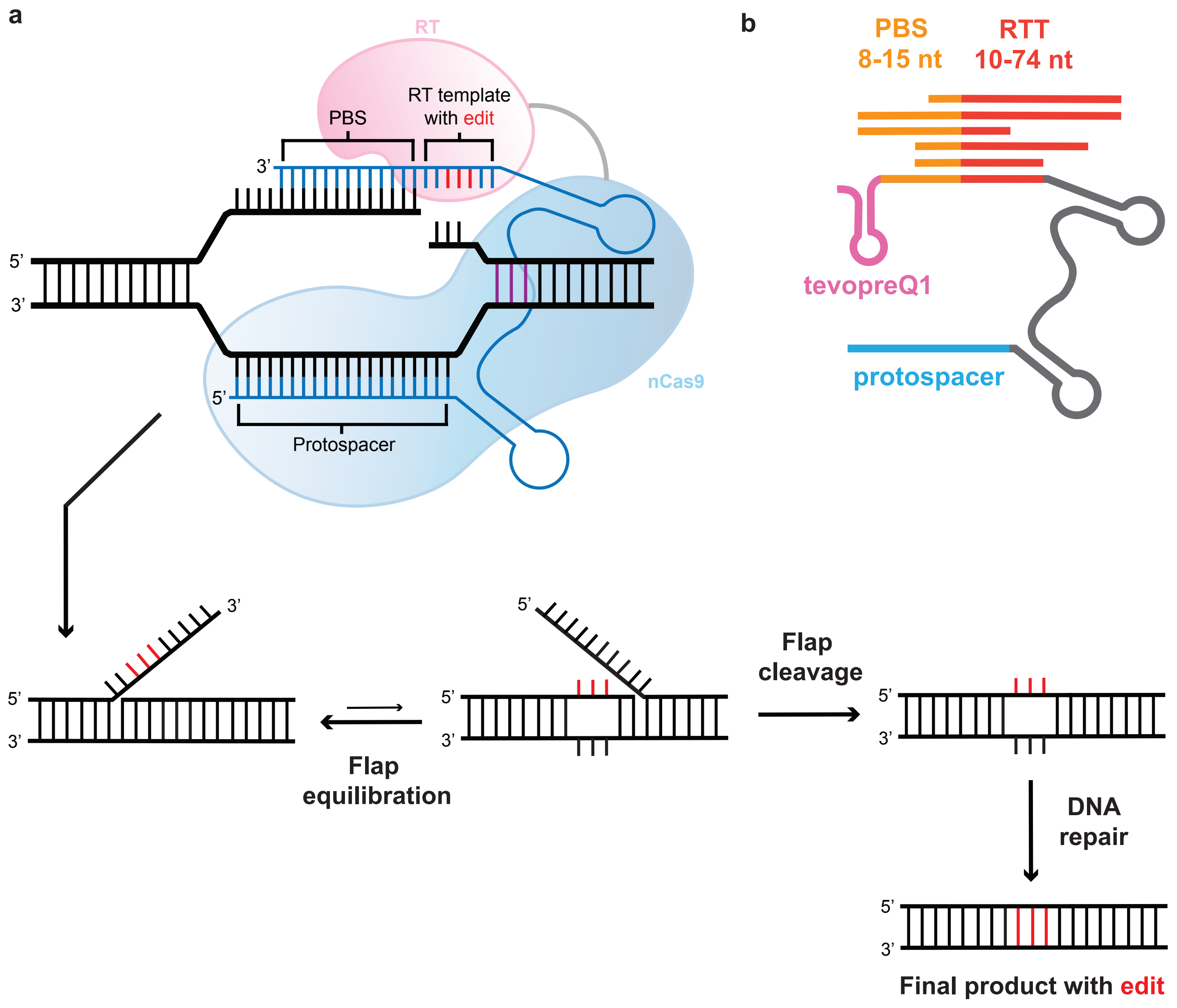
References

1. Anzalone, A. V. *et al.* Search-and-replace genome editing without double-strand breaks or donor DNA. *Nature* **576**, 149–157 (2019).
2. Jinek, M. *et al.* A programmable dual-RNA-guided DNA endonuclease in adaptive bacterial immunity. *Science* **337**, 816–821 (2012).
3. Cong, L. *et al.* Multiplex genome engineering using CRISPR/Cas systems. *Science* **339**, 819–823 (2013).
4. Qi, L. S. *et al.* Repurposing CRISPR as an RNA-guided platform for sequence-specific control of gene expression. *Cell* **152**, 1173–1183 (2013).
5. Gilbert, L. A. *et al.* CRISPR-mediated modular RNA-guided regulation of transcription in eukaryotes. *Cell* **154**, 442–451 (2013).
6. Konermann, S. *et al.* Genome-scale transcriptional activation by an engineered CRISPR-Cas9 complex. *Nature* **517**, 583–588 (2015).
7. Gaudelli, N. M. *et al.* Programmable base editing of A•T to G•C in genomic DNA without DNA cleavage.

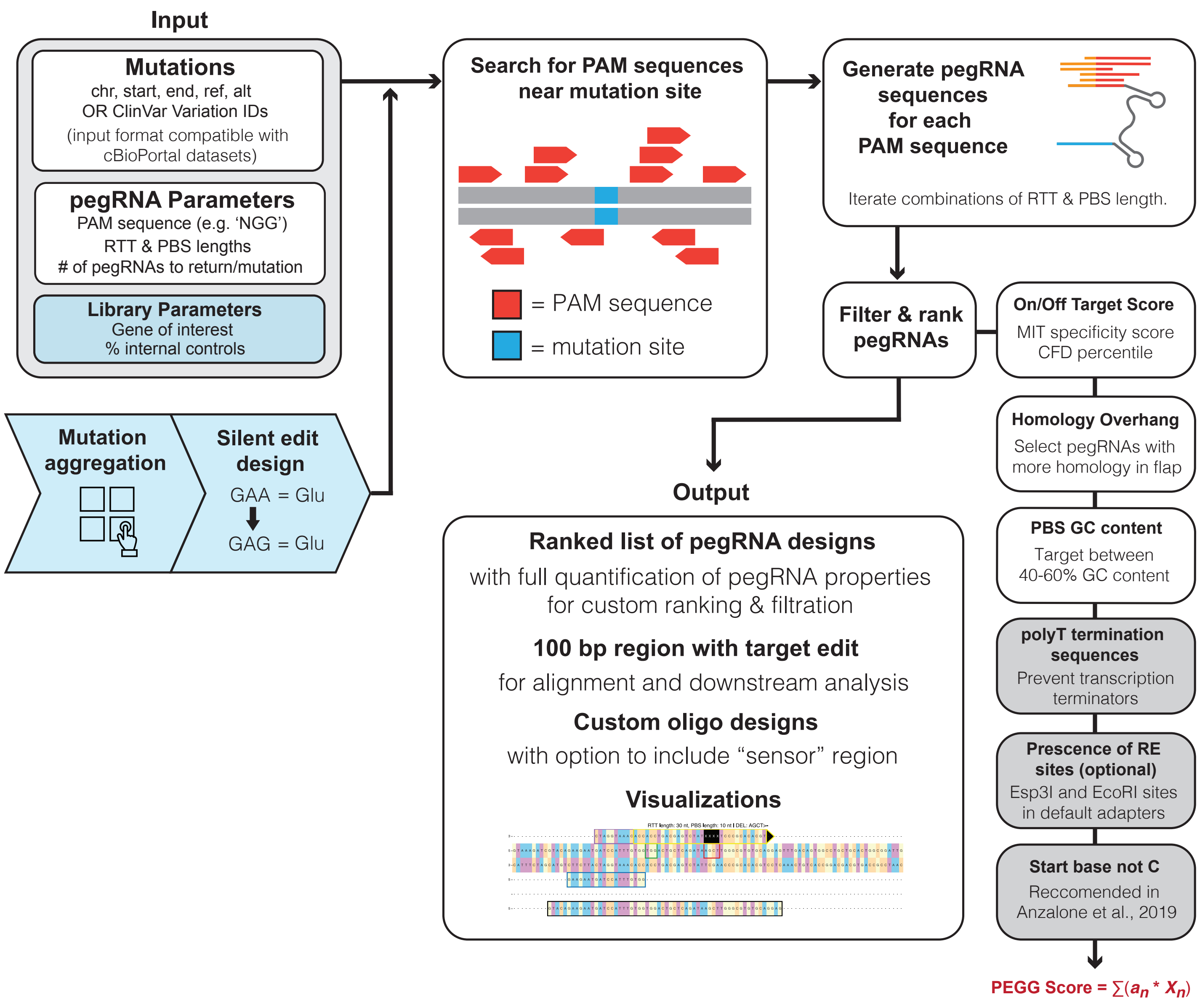
Nature **551**, 464–471 (2017).

8. Anzalone, A. V., Koblan, L. W. & Liu, D. R. Genome editing with CRISPR-Cas nucleases, base editors, transposases and prime editors. *Nat. Biotechnol.* **38**, 824–844 (2020).
9. Nelson, J. W. *et al.* Engineered pegRNAs improve prime editing efficiency. *Nat. Biotechnol.* **40**, 402–410 (2022).
10. Doman, J. L., Sousa, A. A., Randolph, P. B., Chen, P. J. & Liu, D. R. Designing and executing prime editing experiments in mammalian cells. *Nat. Protoc.* (2022) doi:10.1038/s41596-022-00724-4.
11. Bock, C. *et al.* High-content CRISPR screening. *Nature Reviews Methods Primers* **2**, 1–23 (2022).
12. Morris, J. A., Rahman, J. A., Guo, X. & Sanjana, N. E. Automated design of CRISPR prime editors for 56,000 human pathogenic variants. *iScience* **24**, 103380 (2021).
13. Li, Y., Chen, J., Tsai, S. Q. & Cheng, Y. Easy-Prime: a machine learning-based prime editor design tool. *Genome Biol.* **22**, 235 (2021).
14. Chow, R. D., Chen, J. S., Shen, J. & Chen, S. A web tool for the design of prime-editing guide RNAs. *Nat Biomed Eng* **5**, 190–194 (2021).
15. Hsu, J. Y. *et al.* PrimeDesign software for rapid and simplified design of prime editing guide RNAs. *Nat. Commun.* **12**, 1034 (2021).
16. Anderson, M. V., Haldrup, J., Thomsen, E. A., Wolff, J. H. & Mikkelsen, J. G. pegIT - a web-based design tool for prime editing. *Nucleic Acids Res.* **49**, W505–W509 (2021).
17. Hwang, G.-H. *et al.* PE-Designer and PE-Analyzer: web-based design and analysis tools for CRISPR prime editing. *Nucleic Acids Res.* **49**, W499–W504 (2021).
18. Standage-Beier, K., Tekel, S. J., Brafman, D. A. & Wang, X. Prime Editing Guide RNA Design Automation Using PINE-CONE. *ACS Synth. Biol.* **10**, 422–427 (2021).
19. Bhagwat, A. M. *et al.* multicrispr: gRNA design for prime editing and parallel targeting of thousands of targets. *Life Sci Alliance* **3**, (2020).
20. Cerami, E. *et al.* The cBio cancer genomics portal: an open platform for exploring multidimensional cancer genomics data. *Cancer Discov.* **2**, 401–404 (2012).
21. Landrum, M. J. *et al.* ClinVar: improving access to variant interpretations and supporting evidence. *Nucleic Acids Res.* **46**, D1062–D1067 (2018).

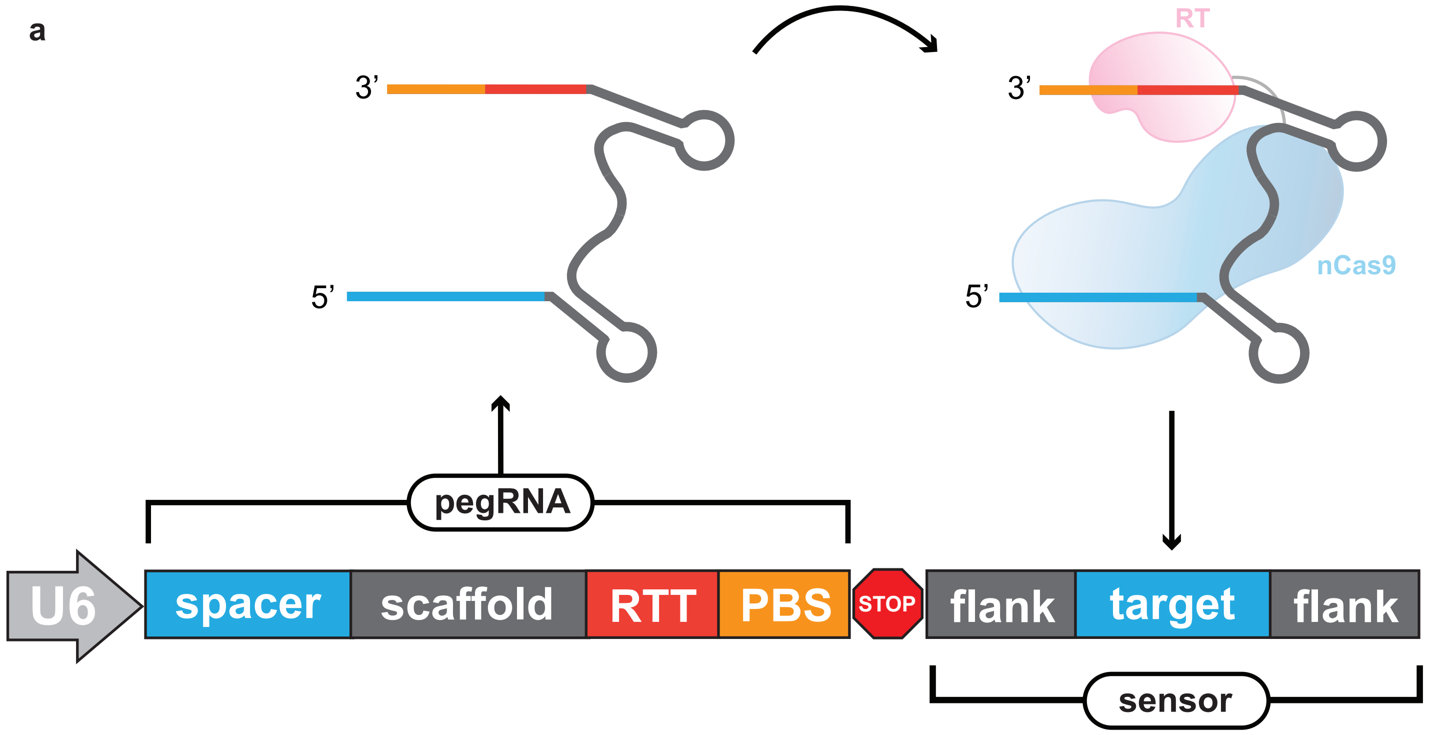
22. Kim, H. K. *et al.* Predicting the efficiency of prime editing guide RNAs in human cells. *Nat. Biotechnol.* **39**, 198–206 (2021).
23. Chen, P. J. *et al.* Enhanced prime editing systems by manipulating cellular determinants of editing outcomes. *Cell* **184**, 5635–5652.e29 (2021).
24. Doench, J. G. *et al.* Optimized sgRNA design to maximize activity and minimize off-target effects of CRISPR-Cas9. *Nat. Biotechnol.* **34**, 184–191 (2016).
25. Hsu, P. D. *et al.* DNA targeting specificity of RNA-guided Cas9 nucleases. *Nat. Biotechnol.* **31**, 827–832 (2013).
26. Sánchez-Rivera, F. J. *et al.* Base editing sensor libraries for high-throughput engineering and functional analysis of cancer-associated single nucleotide variants. *Nat. Biotechnol.* **40**, 862–873 (2022).
27. Zehir, A. *et al.* Mutational landscape of metastatic cancer revealed from prospective clinical sequencing of 10,000 patients. *Nat. Med.* **23**, 703–713 (2017).
28. Kastenhuber, E. R. & Lowe, S. W. Putting p53 in Context. *Cell* **170**, 1062–1078 (2017).
29. Li, M. M. *et al.* Standards and Guidelines for the Interpretation and Reporting of Sequence Variants in Cancer: A Joint Consensus Recommendation of the Association for Molecular Pathology, American Society of Clinical Oncology, and College of American Pathologists. *J. Mol. Diagn.* **19**, 4–23 (2017).
30. Houlleberghs, H. *et al.* Suspected Lynch syndrome associated MSH6 variants: A functional assay to determine their pathogenicity. *PLoS Genet.* **13**, e1006765 (2017).
31. Berends, M. J. W. *et al.* Molecular and clinical characteristics of MSH6 variants: an analysis of 25 index carriers of a germline variant. *Am. J. Hum. Genet.* **70**, 26–37 (2002).
32. Chen, B. *et al.* Dynamic imaging of genomic loci in living human cells by an optimized CRISPR/Cas system. *Cell* **155**, 1479–1491 (2013).

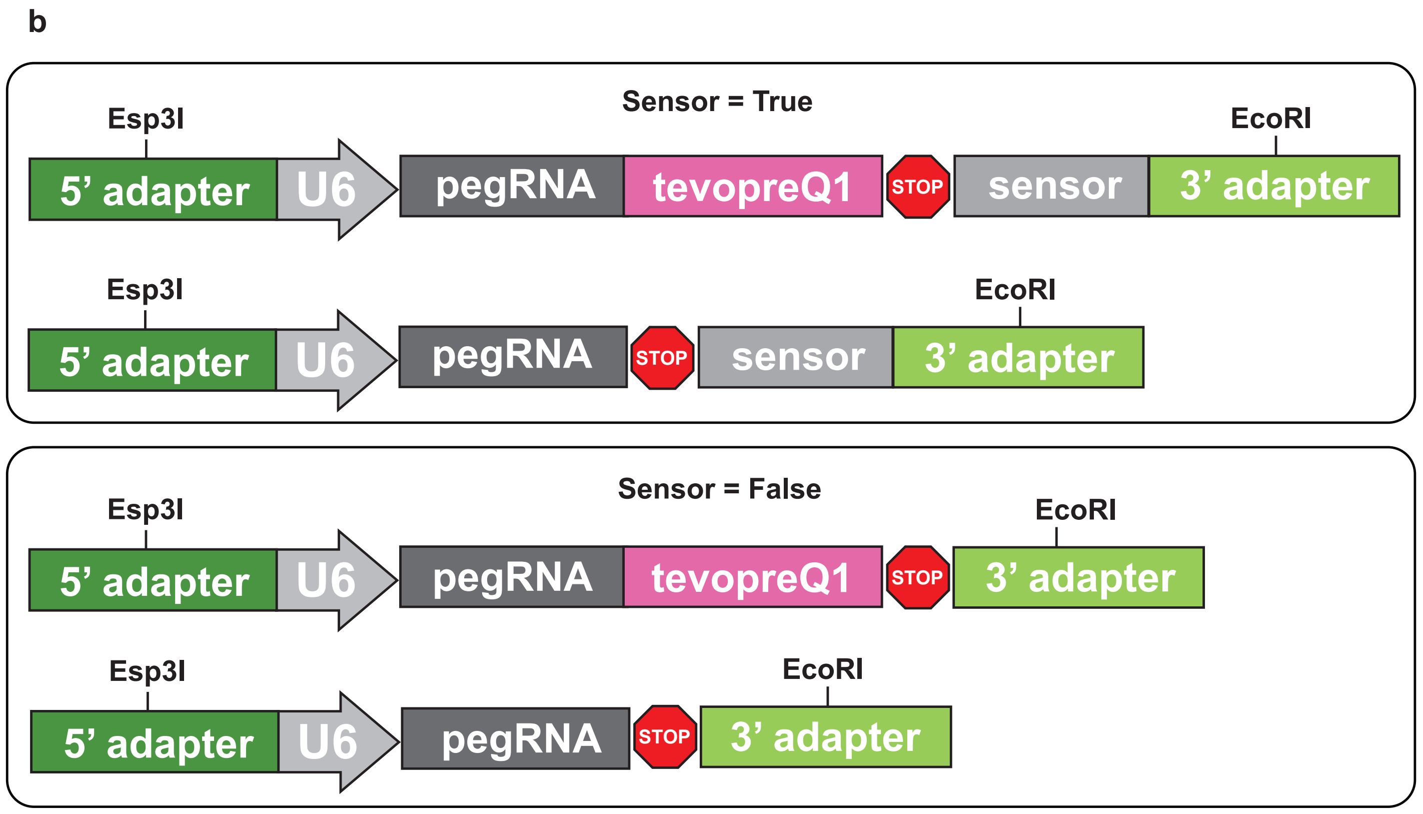


Extended Figure 1. Prime Editing Mechanism. a. Prime editing produces a 3' flap containing the edit of interest (shown in red). This 3' flap hybridizes with the genomic DNA, producing a 5' flap that is cleaved. Finally, the nick is ligated and DNA repair either produces the final product containing the edit, or reverts the sequence back to its original form. **b.** All pegRNAs contain a protospacer (blue), a scaffold region (gray), and a 3' extension, which consists of a reverse transcriptase template (RTT) (red), and a primer binding sequence (PBS) (orange). The RTT and PBS can vary in length, meaning many possible pegRNA designs are possible for a given mutation. In addition, a structured RNA psuedoknot, like tevopreQ[9], shown in pink, can be added to the pegRNA to prevent its degradation and thus improve its efficiency.


Input
Mutations
chr, start, end, ref, alt
OR ClinVar Variation IDs
(input format compatible with cBioPortal datasets)
pegRNA Parameters
PAM sequence (e.g. 'NGG')
RTT & PBS lengths
# of pegRNAs to return/mutation
Library Parameters
Gene of interest
% internal controls
Mutation aggregation
Silent edit design
GAA = Glu
GAG = Glu
Search for PAM sequences near mutation site
= PAM sequence
= mutation site
Generate pegRNA sequences for each PAM sequence
Iterate combinations of RTT & PBS length.
Filter & rank pegRNAs
On/Off Target Score
MIT specificity score
CFD percentile
Homology Overhang
Select pegRNAs with more homology in flap
PBS GC content
Target between 40-60% GC content
polyT termination sequences
Prevent transcription terminators
Prescence of RE sites (optional)
Esp3I and EcoRI sites in default adapters
Start base not C
Reccomended in Anzalone et al., 2019
PEGG Score = ∑(a_n * X_n)
Output
Ranked list of pegRNA designs
with full quantification of pegRNA properties for custom ranking & filtration
100 bp region with target edit
for alignment and downstream analysis
Custom oligo designs
with option to include "sensor" region
Visualizations

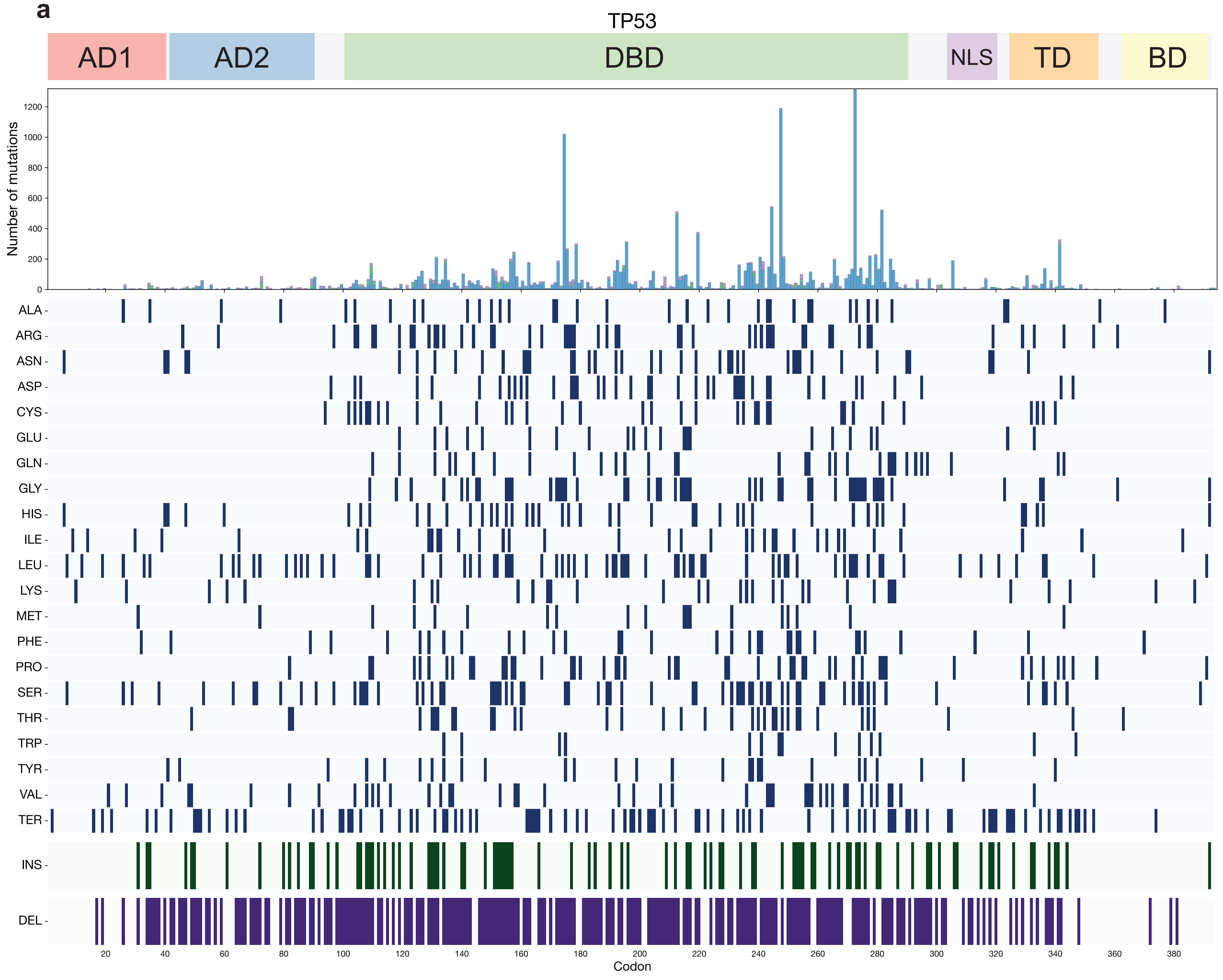
Extended Figure 2. Extended PEGG pipeline schematic. PEGG takes in a list of input mutations in a format compatible with datasets in the cBioPortal[20], or a list of ClinVar variation IDs, and a set of user-defined pegRNA parameters. For each mutation, PEGG searches for PAM sequences and generates pegRNA designs at each PAM sequence iterating over the possible combinations of PBS and RTT lengths. Lastly, PEGG ranks the pegRNA designs by their PEGG score, which is a linear weighted sum of the pegRNA features shown in the figure (binary features highlighted in gray) to return the output, a ranked list of pegRNA designs and additional functionalities.



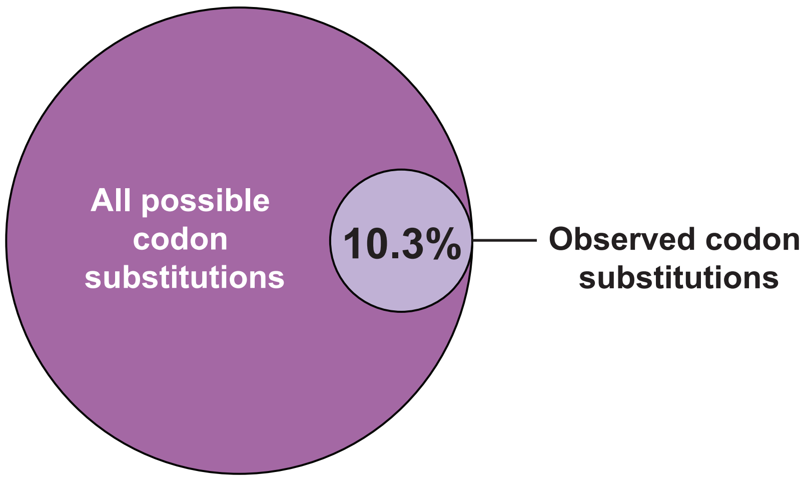


Extended Figure 3. Prime editing sensors provide a proxy of editing outcomes at the endogenous locus. a. In a prime editing sensor, the pegRNA is directly adjacent to a synthetic version of the target site–the sensor. Prime editing acts on this sensor site, allowing for determination of both enrichment and depletion of a given pegRNA, as well as a proxy measurement of editing outcomes at the endogenous locus with a single amplicon. Spacer = protospacer; RTT = reverse transcriptase template; PBS = primer binding sequence; Stop = U6 terminator sequence. **b.** Options for custom oligonucleotide output with PEGG. The sensor region can be included (top) or excluded (bottom). For each oligo, a version with or without the epeg motif tevopreQ1 is provided[9]. Shown are the default 3' and 5' adapters which contain Esp3I and EcoRI restriction sites for cloning. These, as well as the scaffold region, can be easily replaced. Stop = U6 terminator sequence.

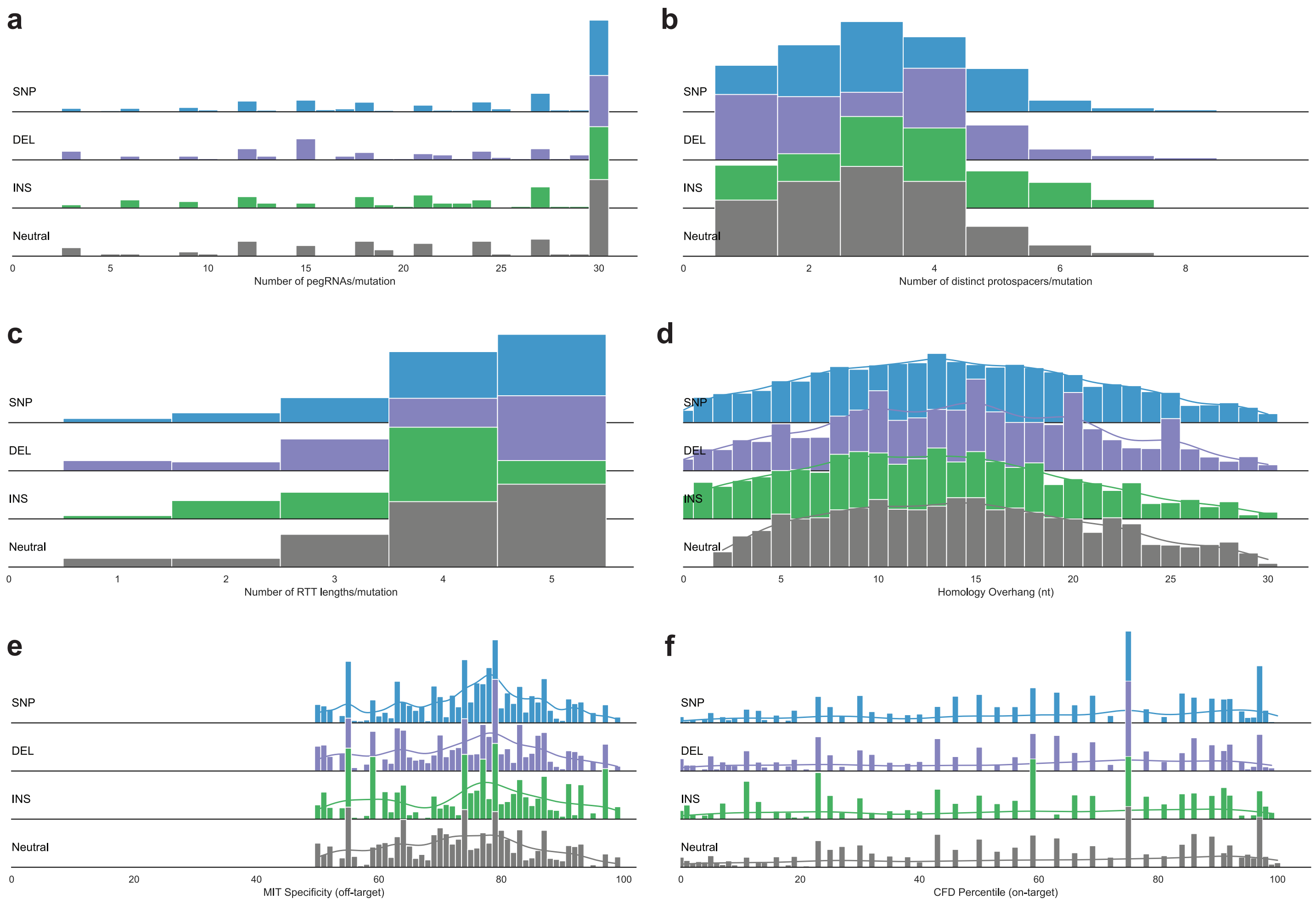
a



b


All possible codon substitutions
10.3%
Observed codon substitutions

Extended Figure 4. Input mutations for TP53 library design. a. The mutations selected as input into the PE sensor library: all observed SNPs, as well as insertions and deletions occurring in at least 3 patients. The identity of codon substitutions for SNPs is represented, as well as the start location of insertions and deletions. **b.** The observed codon substitutions in the MSK-IMPACT cohort represent ~10% of all possible codon substitutions[27].



Extended Figure 5. Properties of pegRNAs in TP53 library. a. Histogram of the number of pegRNAs designed per input mutation, stratified by variant type. We sought to design 30 pegRNAs for each mutation. However, this is not possible for all input mutations, though the majority have 30 pegRNA designs. **b.** Histogram of the number of distinct protospacers represented among the pegRNA designs for each input mutation. **c.** Histogram of the number of RTT lengths represented among the pegRNA designs for each input mutation. Mutations with only 1 RTT length represented have only the longest RTT length (30 nt) among their pegRNA designs, while mutations with 5 RTT lengths represented contain all of the RTT lengths (10, 15, 20, 25, 30 nt). **d.** Histogram of the homology overhang in the 3' flap of each pegRNA design in the library. **e.** Off-target scoring (MIT specificity)[23] of each pegRNA design in the library. We set a threshold of ≥50 to minimize off-target editing. **f.** On-target scoring (CFD) of each pegRNA design in the library[22].

Comparison of pegRNA design tools

Design Tool	Single pegRNA design?	Library design?	Oligo Design?	Automated sensor design?	Flexible data input?	Fully customiz-able output?	Visualiz-ations available?	Web-based?	Document-ation & tutorials available?	ML-based efficiency prediction?	REF
pegFinder	Yes	No	Yes	No	No	No	No	Yes	No	No	link
PrimeDesign	Yes	No	No	No	No	No	No	Yes	No	No	link
Easy-Prime	Yes	No	No	No	Yes	No	Yes	Yes	No	Yes	link
pegIT	Yes	No	Yes	No	Yes	No	Yes	Yes	No	No	link
Prime Editing Design Tool	Yes	No	Yes	No	No	No	No	Yes	No	No	link
PE-Designer	Yes	No	No	No	No	No	Yes	Yes	No	No	link
PINE-CONE	Yes	Yes	Yes	No	No	No	No	No	No	No	link
multicrispr	Yes	Yes	Yes	No	No	Yes	No	No	No	No	link
PEGG	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No*	link

Table 1. Comparison of pegRNA design tools.