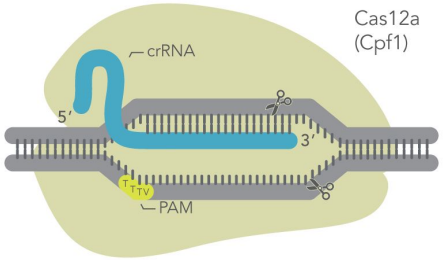
A Python Package to Automate the Generation of CRISPR Arrays: crispr_array_generator

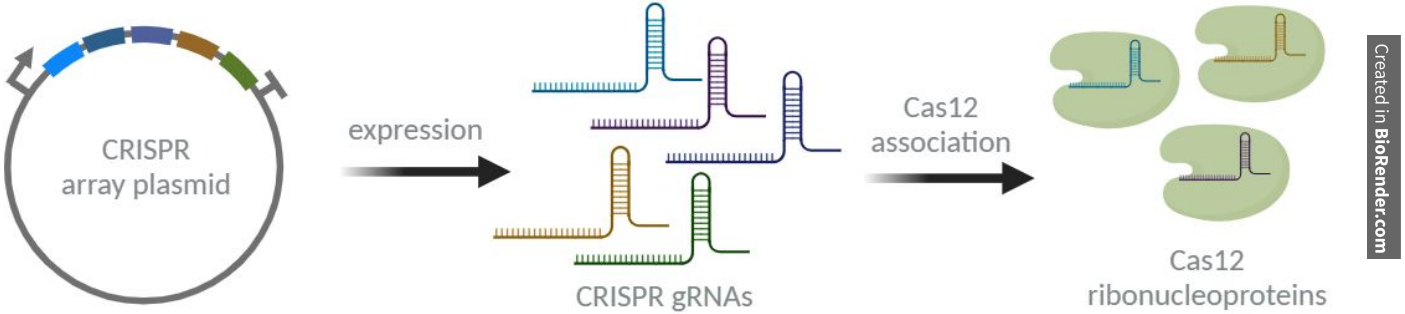
Willow Chernoske
BIOEN 537
December 6, 2023

Background: What are CRISPR arrays and why are they useful?

- CRISPR nucleases are widely used gene editing systems that are site-directed by RNA molecules
- CRISPR arrays encode for multiple guide (g)RNAs within a single transcript, allowing for multiplexed genome editing
  - This system utilizes the capacity of CRISPR Cas12 to auto-process its own gRNAs



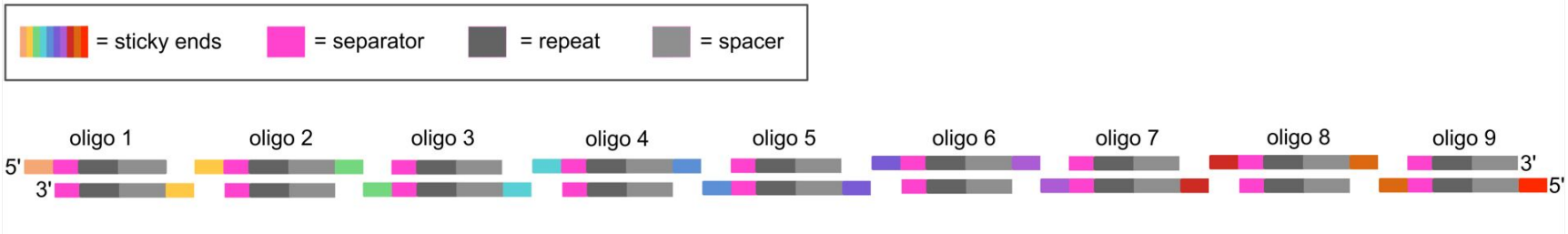
Cas12a nuclease [source: IDT]



CRISPR array structure and function

Problem statement

- Ordering a fully-assembled array is expensive and troublesome due to significant repeat sequences
- To combat this, CRISPR arrays can be ordered in oligonucleotide (single stranded DNA) parts, but this makes the design process more tedious and prone to error
- No tool currently exists to automate this design process



CRISPR array design layout

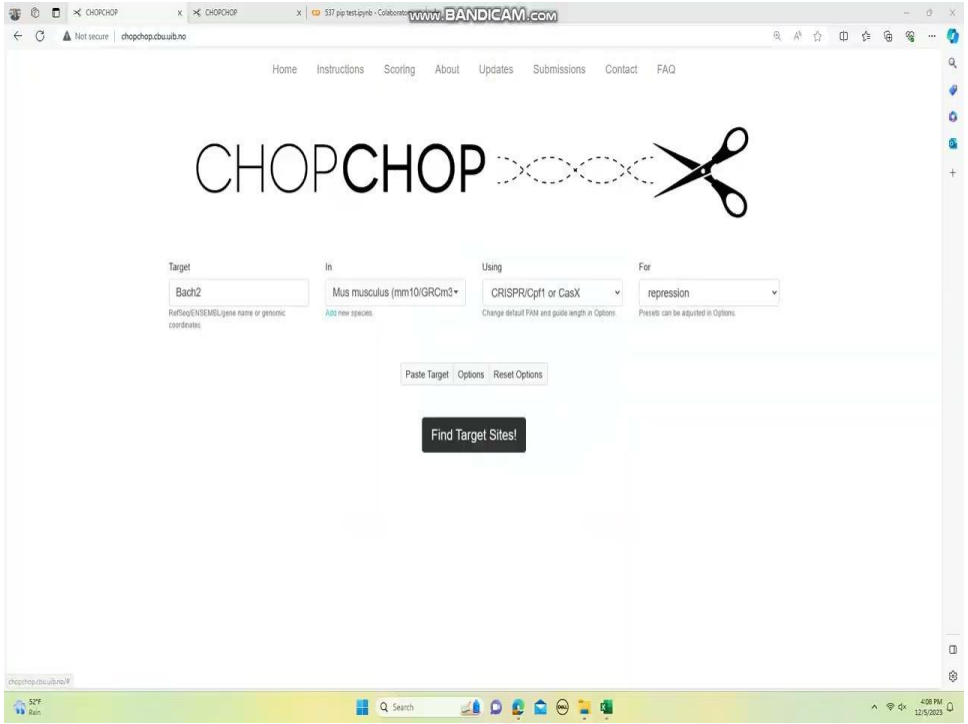
Use cases: crispr_array_generator

Goal: Develop a python program that automates the design of CRISPR Cas12a arrays for up to 9 gRNAs

Use Cases:

- Check existing CRISPR gRNAs for errors or use them to generate CRISPR arrays
  a. Open python, install crispr_array_generator using pip
  b. Upload an excel file containing all gRNAs or input them in an array
  c. Call the **check_grna** or **get_array** function and input the name of the excel file or array
  d. Function will output an excel file with the processed gRNAs and notices of any errors if present. If using get_array, output will include a second sheet with all ready-to-order forward and reverse oligos and the fully assembled array sequence.

Demo


www.BANDICAM.com
CHOPCHOP
537.pip test.ipynb - Colaboratory
Not secure
chopchop.cbu.uib.no
Home
Instructions
Scoring
About
Updates
Submissions
Contact
FAQ
CHOPCHOP
Target
Bach2
RefSeq/ENSEMBL/gene name or genomic coordinates.
In
Mus musculus (mm10/GRCm3
Add new species.
Using
CRISPR/Cpf1 or CasX
Change default PAM and guide length in Options.
For
repression
Presets can be adjusted in Options.
Paste Target
Options
Reset Options
Find Target Sites!
chopchop.cbu.uib.no/#
Search

Design: package functions

extract_excel_data(excel_file: str)

- Extracts all gRNA components from an excel file and puts them into a python array

check_grna(grnas: str {name of excel file} or array)

- Checks grnas for common errors

get_reverse_complement(dna: str)

- Produces the reverse complement of a DNA string

make_columns_best_fit(excel_file: str)

- Formats excel file columns

get_array(grnas: str {name of excel file} or array)

- Checks grnas for common errors and generates ready-to-order array components

get_array(grnas)

1. Checks if input is array or string
2. If string, calls extract_excel_data
3. Calls check_grnas
4. Calls get_reverse_complement
5. Assembles array
6. Calls make_columns_best_fit and outputs excel file

Project structure

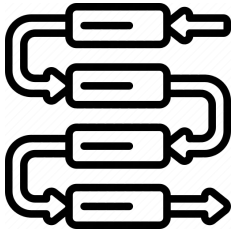
- crispr_array_generator
  - __init__.py
  - constants.py
  - crisprarraygenerator.py → Bulk of the code required for my package
- docs
  - Component_Specification.pdf → Documentation
  - Functional_Specification.pdf
- test
  - test_crisprarraygenerator → Test code
- .gitignore
- LICENSE
- README.md → Overview and user instructions
- setup.py

Github link

Lessons learned and future work

Software engineering lessons

- Test your code regularly
- Pay attention to dependencies



Future work

- Finish writing and testing test_crisprarraygenerater.py
- More thorough guide check (GC content, self complementarity)
- Make compatible for different CRISPR enzymes
- Graphical user interface (GUI) to simplify user interaction

Thank you