PNA Finder

A bioinformatics-based tool for rapid antisense PNA design

Overview

PNA Finder is a Python-based toolbox that combines the functions of several bioinformatics alignment programs with a series of custom scripts as a means to design antisense peptide nucleic acids (PNA). The *PNA Finder v1.0* consists of the `Get Sequences`, `Find Off-Targets`, and `Check Sequence Warnings` functions, which can both be used in tandem to identify, design, and screen antisense sequences in a high-throughput manner.

Table of Contents

Setup

Prerequisite programs

The *PNA Finder* toolbox is built in Python 3.7 to run on a Windows operating system. It requires the following programs:

- Python (≥ version 2.7)
- Alignment programs
  - Bowtie (version 1, not Bowtie2)
  - BEDTools (version 2.25)
  - SAMTools (any version)

In order to run the toolbox, it is also necessary to install a Bash shell that is able to run these alignment programs. *PNA Finder* has built-in compatibility with both Cygwin and the Windows 10 Bash shell (other shell options will require slight modification to the *PNA Finder* scripts). All three alignment programs must be compiled on the Bash shell prior to using *PNA Finder* (see the programs' respective online manuals for installation instructions).

For Cygwin, the following packages are necessary for installing these programs:

```
make
```

```
gcc-g++
zlib
zlib-devel
libncurses-devel
libbz2-devel
liblzma-devel
curl
libcurl-devel
gdb
```
`python2` or `python3`

Once these packages are installed, the programs may be installed according to their own manuals using a `make` command. For Bowtie, installation in Cygwin requires the `NO_TBB=1` option following the `make` command.

The Windows native bash shell does not require additional package downloads. It should be noted, however, that a single Bowtie distribution cannot be installed simultaneously on both the Windows bash shell and Cygwin.

Installing PNA Finder

A standard command line pip command may be used for package installation:

```
pip install pna_finder
```

or

```
python -m pip install pna_finder
```

Alternately, the zipped file of the package (complete with example files) may be downloaded from: https://github.com/taunins/pna_finder. If installing *PNA Finder* using the zip file, it will be necessary to install the Python packages `gffutils` and `biopython` (other packages may need manual installation as well, depending on what pre-installed packages your Python distribution came with).

PNA Finder start_info

Before running *PNA Finder* for the first time, it is necessary to load the `start_info` module and call the `run()` function. This can be done using a Python shell or by directly running the script `run_start_info.py` in the package's top-level directory.

Calling this function will open the following dialog box, which allows the user to select a home directory for *PNA Finder*'s file browsing, as well as the directory where the desired `bash.exe` program is located. The dialog box also has two radio buttons for the user to select which of the supported Bash shells they have chosen to use. The user may then select `OK` to save these settings, and they will be saved in the module folder as `start_info.txt`. The settings can be changed by calling `start_info.run()` again.

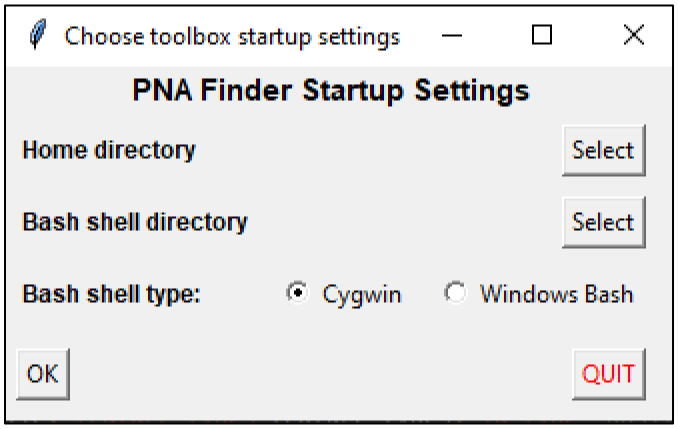
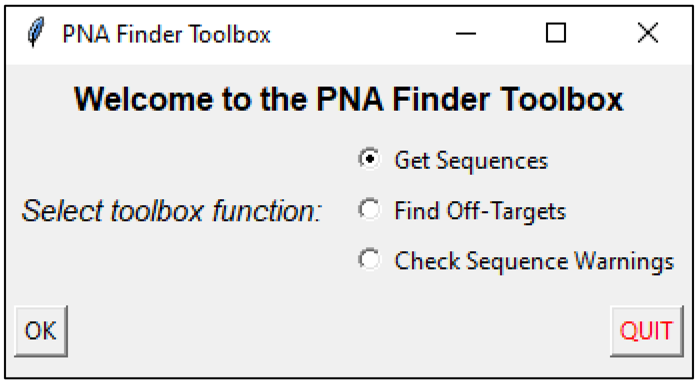


Figure 1 *GUI window for input of PNA Finder startup settings. Launch by running the script run_start_info.py, or by using the run() command from the start_info module.*

<u>Running *PNA Finder*</u>

Similar to the `start_info.run()` command, the main functions of *PNA Finder* can be run by loading the `pna_finder` module and calling the function `run()`. This can be done using a Python shell or by running the script `run_pna_finder.py` in the package's top-level directory.

Calling this function will open the following dialog box. The user must then select either the `Get Sequences`, `Find Off-Targets`, or `Check Sequence Warnings` function and select `OK` to continue.


PNA Finder Toolbox

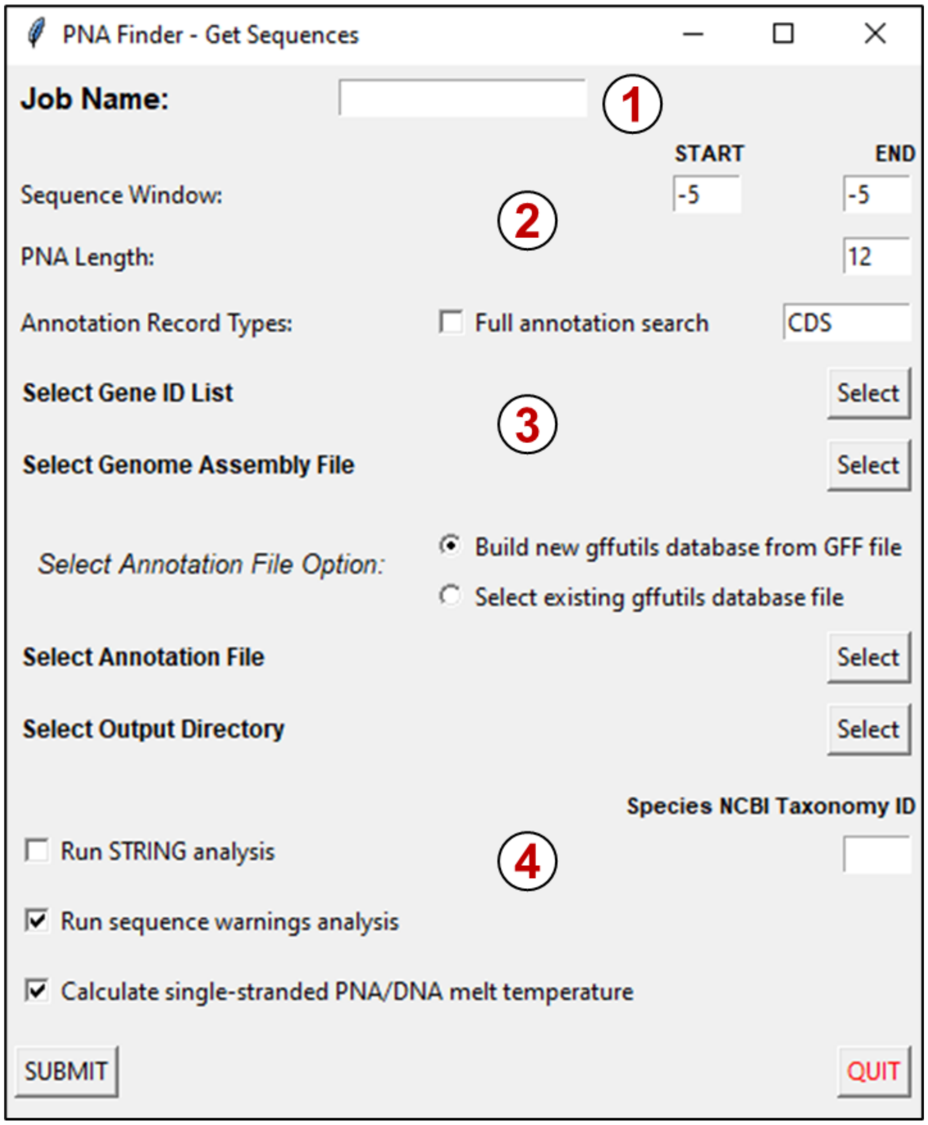
Welcome to the PNA Finder Toolbox

Select toolbox function: [✓ Get Sequences] Find Off-Targets Check Sequence Warnings

Figure 2 *GUI for function selection for PNA Finder. Launch by running the script run_pna_finder.py, or by using the run() command from the pna_finder module.*

Get Sequences

The *Get Sequences* function is used to build a set of antisense PNA candidates based on a user-provided list of target genes. The setup dialog box for *Get Sequences* is shown below, with each individual section numbered:


PNA Finder - Get Sequences

Job Name:

Sequence Window: START -5 END -5

PNA Length: 12

Annotation Record Types: ☐ Full annotation search CDS

Select Gene ID List

Select Genome Assembly File

Select Annotation File Option: [✓ Build new gffutils database from GFF file] Select existing gffutils database file

Select Annotation File

Select Output Directory

Species NCBI Taxonomy ID

☐ Run STRING analysis

☑ Run sequence warnings analysis

☑ Calculate single-stranded PNA/DNA melt temperature

Figure 3 *Get Sequences dialog box.*

1. Job name

The results of the *Get Sequences* job will be written to a directory with this title.

2. *Get Sequences* parameters

The parameters "Sequence Window Start" and "Sequence Window End" determine the number of PNA sequences that will be produced for each gene target submitted. The integers refer to coordinates relative to the feature (e.g. gene, exon, mRNA) start locus. As an example, take the nucleotide sequence surrounding the start codon for the *E. coli* gene *thrL*. Using the default entries of -5 for both window parameters, as well as the default entry of 12 for the "PNA Length" parameter, we are provided with a single 12-nt PNA target sequence. This sequence is highlighted in the nucleotide sequence below:

```
E. coli MG1655 thrL
-20           -5    0                 +20
TTACAGAGTACACAACATCCATGAAACGCATTAGCACCAC
```

However, if instead the “Sequence Window End” parameter is changed to 0, we obtain six 12-nt PNA target sequences, as shown below:

```
E. coli MG1655 thrL
-20           -5    0                 +20
TTACAGAGTACACAACATCCATGAAACGCATTAGCACCAC
                ATCCATGAAACG
                 TCCATGAAACGC
                  CCATGAAACGCA
                   CATGAAACGCAT
                    ATGAAACGCATT
```

PNA Finder will raise an error message if the “Sequence Window End” parameter is less than the “Sequence Window Start” parameter. Target sequence length can be adjusted by changing the “PNA Length” parameter.

The defaults of [-5,-5] for the sequence window and 12 for the PNA length are based on observations from prior research in bacterial gene inhibition. PNA that binds complementarily to the mRNA start codon has been found to be the most inhibitory of protein translation,[1–5] and PNA lengths between 8 and 12 bases have been shown to most effectively balance translation inhibition with intracellular delivery.[6] Additionally, 12 bases is long enough such that the expected number of off-targets in even the largest bacterial genomes will be less than one (under the simplifying assumption of nucleotide sequence randomness).

3. File upload and annotation search options

The main sequence search function of *Get Sequences* works in two parts: (1) searching a genome annotation file for records matching the user provided gene ID list and (2) using the corresponding genome assembly to extract the desired translation start site nucleotide sequences for these records.

The “Annotation Record Types” entry will depend on the contents of the GFF/GTF annotation file that is being used. *Get Sequences* will search through the annotation file for only records that are labeled with one of the comma-separated list of feature types provided in this box. For example, the standard label for coding sequences in NCBI RefSeq annotation is “CDS,” so this is used as a default. If this entry is left blank, all feature types are included in the search. The “Full annotation search” option is unchecked by default. When this option is not selected, the gffutils database will only be searched for the features whose primary key (see https://pythonhosted.org/gffutils/database-ids.html#database-ids) matches the user-supplied gene ID. If the option is selected, other identifiers (locus_tag, gene_name, various database identifiers, etc.) will also be searched for the gene ID. This may take longer, and is particularly inefficient as a means to search large mammalian genome annotation files.

(Note: The label “gene” is often applicable and equally useful when using prokaryotic genomes. With eukaryotic genomes, the labels “mRNA” and “exon” are often more useful, depending on the PNA application.)

The first file upload is the gene ID list. This should be formatted as a plaintext single-column file containing gene IDs. Different types of gene ID can be included within the same file. The following GFF/GTF annotation identifiers are currently supported:

GFF: gene, Name, gene_synonym, protein_id, locus_tag, Dbxref

GTF: gene_id, gene_name, transcript_id, tss_id

The next upload is a genome assembly. This file should be in the FASTA file format.

Next, the user must select an option for handling the annotation file. *Get Sequences* uses the Python package gffutils to parse the annotation file, which requires creation of a gffutils database file. If the database has not yet been created, the user should select the first option and use the “Select Annotation” file below to upload a GFF/GTF file. *Get Sequences* will then automatically create the database file and use it for the gene ID search. If the required database has already been created, the user should select the second option and upload the database file.

The final selection is an output directory. This will be the location in which *Get Sequences* will create a new results directory (named for the “Job name” entry).

4. PNA candidate analysis options

Get Sequences provides three options for gene target and sequence analysis of PNA candidates. The first is to perform an analysis of each target gene’s protein interaction network using the STRING database.[7] The density of network connections for targeted genes/proteins has been shown to be positively correlated with PNA knockout growth inhibition of bacterial cultures. In order to use this option, the NCBI Taxon ID for the species of interest must be entered as well. The integer ID number can be looked up at the following web address: https://www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi (Note: This option will only work if the STRING database can recognize the PNA candidates as gene names)

The second option is to analyze the PNA candidate sequences for potential solubility issues based on purine content, as described by Gildea et al.[8] *Get Sequences* will produce a solubility warning if it finds, within a stretch of 10 bases, five purines in a row, four guanines in a row, or more than six total purines. Warnings are also produced for a PNA molecule longer than 30 bases or with more than 6 bases of a self-complementary subsequence.

The third option is to calculate the single-stranded PNA/DNA melt temperature, using a correlation described by Giesen et al.[9]

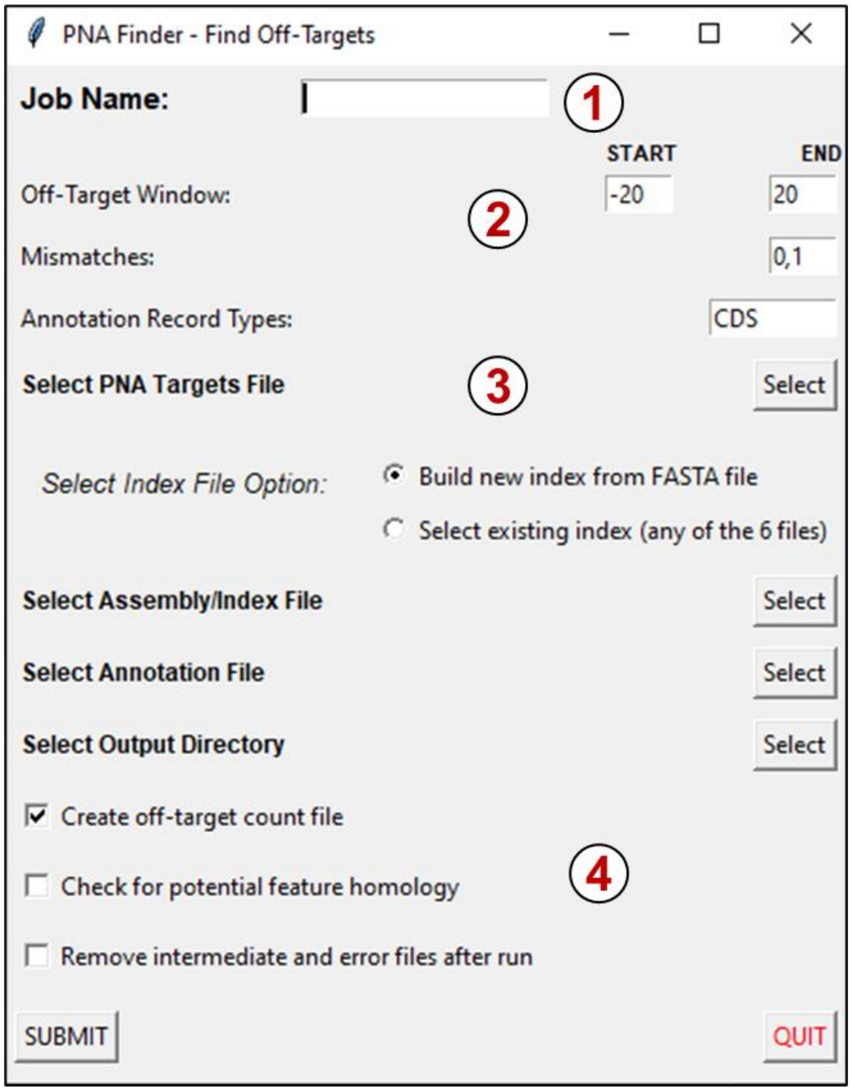
Once all information has been entered into the *Get Sequences* dialog box, the user must click “SUBMIT” to run the *Get Sequences* function. This will produce five files in total, including two BED files, one FASTA file, one tab-delimited “.out” file of PNA sequences, and one BEDTools error record file.

The first BED file will have the same name as the gene ID list file, except with the “.bed” extension. It contains a single record for each gene ID that was matched to an entry in the genome annotation file. The second BED file is an edited version of the first, with the records replicated for as many PNA that will be designed per gene, and indices changed to match the locations of the desired sequences. The BEDTools error record file will contain error information about the ‘bedtools getfasta’ command within *Get Sequences*. It will be empty if the job completes successfully.

The FASTA file will contain a FASTA-formatted list of all candidate PNA target sequences. This may be used as the input for the *Find Off-Targets* function. The tab-delimited “.out” file will contain a table of PNA sequences (the reverse complements of the target sequences), sequence warnings, and PNA/DNA melting temperatures, if the latter two options were selected in the main *Get Sequences* window. If the STRING analysis option is selected, a tab-delimited file named “string.out” will be produced in the output folder, containing each targeted gene’s protein interaction network nodes, edges, and interactors.

Find Off-Targets

The *Find Off-Targets* function is used to search genomes for PNA off-targets that are likely to inhibit gene expression and/or RNA translation. The setup dialog box for *Find Off-Targets* is shown below, with each individual section numbered:



Job Name:

Off-Target Window: START -20 END 20

Mismatches: 0,1

Annotation Record Types: CDS

Select PNA Targets File

Select Index File Option: [✓ Build new index from FASTA file] Select existing index (any of the 6 files)

Select Assembly/Index File

Select Annotation File

Select Output Directory

[✓] Create off-target count file

[ ] Check for potential feature homology

[ ] Remove intermediate and error files after run

Figure 3 *Find Off-Targets dialog box.*

1. **Job name**

   As with *Get Sequences*, the results for the *Find Off-Targets* job will be written to a directory with this title.

2. ***Find Off-Targets* parameters**

The integer entries for “Off-Target Window Start” and “Off-Target Window End” refer to coordinates relative to the feature (e.g. gene, exon, mRNA) start locus. These coordinates are primarily used to search for off-target binding sequences that are most likely to inhibit mRNA translation. The default region for expected inhibition is set to [-20, 20] relative to the RNA start codon, based on prior work.[1]

The parameter “Mismatches” allows the user to request the Bowtie aligner allow for zero to three mismatches in the search for off-target alignments. PNA have shown very high mismatch discrimination in short sequences,[1,10–13] and as such the default mismatch tolerance is set to zero. The user can request multiple alignment searches for various mismatch tolerances by entering a comma-separated list of integers (e.g. “0,1” as in example window above), which will be given as separate files in the output folder. (Note: based on the aligner construction, the one-mismatch run will necessarily include all zero-mismatch alignments as well)

“PNA Length” must be specified by the user to run the appropriate Bowtie 2 alignment command. PNA of differing lengths cannot be combined into the same job.

3. File upload and annotation search options

As with *Get Sequences*, the “Annotation Record Types” entry will depend on the contents of the GFF/GTF annotation file. *Find Off-Targets* will search the uploaded genome for PNA alignments that are found within the off-target window for the types of features specified in this box. The default window, [-20, +20], is most applicable to prokaryotic coding sequence features (named “CDS” in NCBI RefSeq annotations), but can also knock down expression and/or translation in eukaryotic organisms as well.

The first file upload is the PNA Targets FASTA file. This can be manually constructed or taken from the output of *Get Sequences*. The FASTA file should contain a list of PNA target sequences, not the PNA sequences themselves.

Next, the user must select an option for the Bowtie 2 index. To use a genome assembly in Bowtie 2, a set of index files must first be created. To automatically create this index and use it for the *Find Off-Targets* function, the user should select the first option and use the

“Select Assembly/Index” button to upload the genome FASTA file. If the index has already been created, the user should select the second option and use the “Select Assembly/Index” button to upload any of the six Bowtie index files.

The next upload is the annotation file, which should be in GFF or GTF format.

As with *Get Sequences*, the final selection is an output directory. This will be the location in which *Find Off-Targets* will create a new results directory (named for the “Job name” entry).

4. Count, homology, file removal options

The first option allows the user choose whether to create a count file, which, for a single PNA target sequence, tallies the total number of unique alignments that fall within the given window (default [-20,+20], see above) of any gene in the off-target genome.

The second option allows the user to check for potential homology within this set of off-target alignments. This is done with a simple script that searches for the PNA gene name (as determined by the name of the PNA before the first underscore, e.g. PNA “thrL_3” would search for “thrL”) in the given alignment’s feature annotation. If the user requests multiple mismatch tolerances (see above), the homology check will only be run for the first to avoid redundancy (e.g. only for the zero-mismatch alignments in the example window shown). (Note: this function will be improved in future versions of the toolbox, as it is currently crude but still useful)

The third option allows the user to remove intermediate and error files after the *Find Off-Targets* run. This is useful when searching large eukaryotic genomes, as the intermediate files are often large (on the order of gigabytes) and unnecessary for final analysis.

Once all information has been entered into the *Find Off-Targets* dialog box, the user must click “SUBMIT” to run the *Find Off-Targets* function. This will produce 11 or 12 files in total (depending on whether the count file option was selected).

The initial Bowtie 2 alignment will produce a SAM file of all alignments that were found for the given mismatch setting, which will then be converted into a sorted BAM file through a series of SAMTools commands. Each of these commands produces at least one intermediate file, though these will likely not be useful output for the user. The Bowtie 2 and SAMTools commands will also produce individual error record files in the “error_files” subdirectory within the main job folder.

There will be a BED file produced based on these alignments, which contains all alignments within a certain distance (as defined by “Off-Target Window Start”) from any feature in the annotation file. Based on this BED file, a tab-delimited “.out” file will also be produced, containing all PNA alignments to the user-designated off-target window with the feature types specified in the *Find Off-Targets* dialog. If the user has selected the count file option, a “.count” file that tabulates total inhibitory off-target alignments for each PNA candidate will also be produced in the job directory. If the user has selected the homology check option, a tab-delimited file named “homology.out” will be produced in the output folder, containing the PNA candidate name, the line number within the “.out” file where it finds a matching annotation, and the annotation itself.

Check Sequence Warnings

The *Check Sequence Warnings* function is used to check PNA sequences for the solubility and self-complement warnings mentioned in the *Get Sequences* section (subsection 4). The setup dialog box for *Check Sequence Warnings* is shown below:

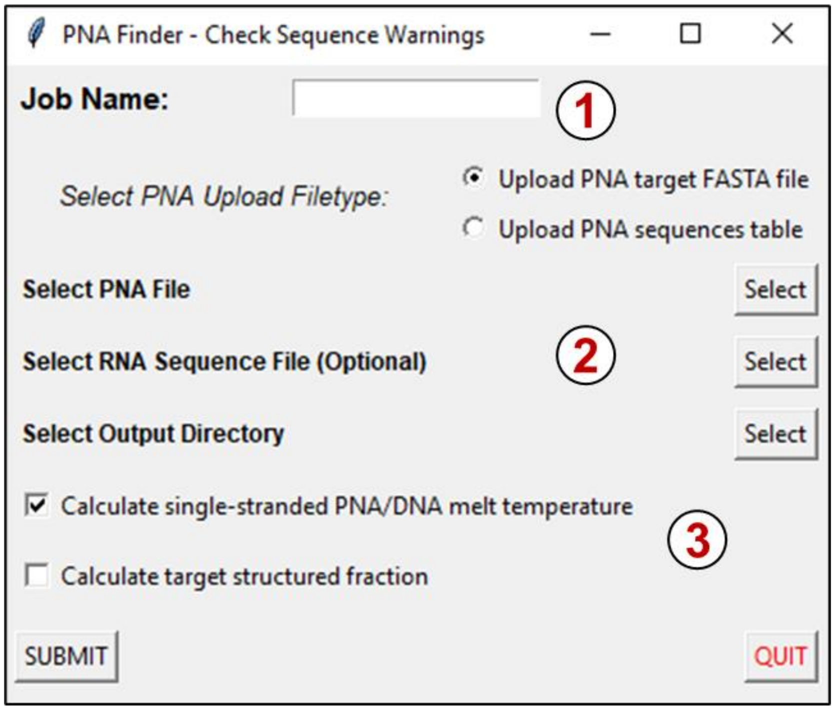

PNA Finder - Check Sequence Warnings
Job Name:
1
Select PNA Upload Filetype:
Upload PNA target FASTA file
Upload PNA sequences table
Select PNA File
Select
Select RNA Sequence File (Optional)
2
Select
Select Output Directory
Select
Calculate single-stranded PNA/DNA melt temperature
3
Calculate target structured fraction
SUBMIT
QUIT

Figure 4 *Check Sequence Warnings dialog box.*

1. **Job name**

   As with the other functions, the results for the *Check Sequence Warnings* job will be written to a directory with this title.

2. **File Upload**

   The only required file upload for *Check Sequence Warnings* is a file containing either a FASTA format list of target sequences or a table of PNA sequences. If the latter option is chosen (via the lower radio button), it must be formatted with the PNA name in the first column, followed by the PNA sequence (the reverse complement of the target sequence) in the second column. Columns should be tab-delimited.

   A list of RNA sequences may also be uploaded for analysis of the secondary structure of the PNA target sequences. Folding calculation is performed using the ViennaRNA RNAfold program, using default settings. Folding calculation details are placed in a folder named "folding_files" in the main output directory. Additionally, a column is appended to the output file with a calculation of the fraction of the RNA target sequence that is involved in folding.

   As with the other two functions, the final selection is an output directory. This will be the location in which *Check Sequence Warnings* will create a new results directory (named for the "Job name" entry).

3. **Analysis options**

As with *Get Seqeunces*, the user is provided the option to calculate the single-stranded PNA/DNA melt temperature, using a correlation described by Giesen et al.[9] Additionally, the "Calculate target structured fraction" box must be checked to performing folding analysis using the uploaded RNA sequences file.

As with the *Get Sequences* function, there will be a tab-delimited ".out" file in the output directory, which will contain a table of PNA sequences (the reverse complements of the target sequences) and sequence warnings.

PNA Finder Examples

Example files for the inputs and outputs of PNA Finder can be found in the "examples" subdirectory of the package folder.

The *Get Sequences* example uses the text file "ecoli_gene_id.txt". This file lists different annotation identifiers for the first six coding sequences of the reference *E. coli* MG1655 annotation file, located in the directory "genome_files/ecoli_MG1655". This text file, along with the annotation file and genome assembly FASTA file, can be uploaded to PNA Finder's *Get Sequences* function to produce the output found in the "get_sequences_example" directory.

The *Find Off-Targets* example uses the FASTA file output found in that same "get_sequences_example" directory, together with the *E. coli* MG1655 annotation file and Bowtie 2 index (located in "genome_files/ecoli_MG1655"), to produce the output found in the "find_off-targets_example" directory.

References

1. Courtney, C. M. & Chatterjee, A. Sequence-Specific Peptide Nucleic Acid-Based Antisense Inhibitors of TEM-1 β-Lactamase and Mechanism of Adaptive Resistance. *ACS Infect. Dis.* **1**, 253–263 (2015).
2. Dryselius, R., Aswasti, S. K., Rajarao, G. K., Nielsen, P. E. & Good, L. The translation start codon region is sensitive to antisense PNA inhibition in Escherichia coli. *Oligonucleotides* **13**, 427–433 (2003).
3. Mondhe, M., Chessher, A., Goh, S., Good, L. & Stach, J. E. M. Species-Selective Killing of Bacteria by Antimicrobial Peptide-PNAs. *PLoS One* **9**, e89082 (2014).
4. Otsuka, T. *et al.* Antimicrobial activity of antisense peptide–peptide nucleic acid conjugates against non-typeable Haemophilus influenzae in planktonic and biofilm forms. *J. Antimicrob. Chemother.* **72**, 137–144 (2017).
5. Bai, H. *et al.* Antisense inhibition of gene expression and growth in gram-negative bacteria by cell-penetrating peptide conjugates of peptide nucleic acids targeted to rpoD gene. *Biomaterials* **33**, 659–667 (2012).
6. Good, L., Awasthi, S. K., Dryselius, R., Larsson, O. & Nielsen, P. E. Bactericidal antisense effects of peptide-PNA conjugates. *Nat. Biotechnol.* **19**, 360–364 (2001).
7. Szklarczyk, D. *et al.* STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Res.* **47**, D607–D613 (2019).
8. Gildea, B. D. & Coull, J. M. Methods for Modulating the Solubility of Synthetic Polymers. (2004).
9. Giesen, U. *et al.* A formula for thermal stability (T(m)) prediction of PNA/DNA duplexes. *Nucleic Acids Res.* **26**, 5004–5006 (1998).
10. Doyle, D. F., Braasch, D. A., Janowski, B. A. & Corey, D. R. Inhibition of Gene Expression Inside Cells by Peptide Nucleic Acids: Effect of mRNA Target Sequence, Mismatched Bases, and PNA Length. *Biochemistry* **40**, 53–64 (2001).
11. Good, L. & Nielsen, P. E. Inhibition of translation and bacterial growth by peptide nucleic acid targeted to ribosomal RNA. *Biochemistry* **95**, 2073–2076 (1998).
12. Weiler, J., Gausepohl, H., Hauser, N., Jensen, O. N. & Hoheisel, J. D. Hybridisation based DNA screening on peptide nucleic acid (PNA) oligomer arrays. *Nucleic Acids Res.* **25**, 2792–2799 (1997).
13. Choi, J., Jang, M., Kim, J. & Park, H. Highly sensitive PNA Array Platform Technology for Single Nucleotide Mismatch Discrimination. *J. Microbiol. Biotechnol.* **20**, 287–293 (2010).