SlimShadey User Manual

Avi Shah, Aneesha Kodati, and Sudhir Nayak
The College of New Jersey
2000 Pennington Rd.
Ewing, NJ, USA 08628-0718

Contents

Introduction

What is Slimshadey?

SlimShadey is a lightweight Java-based tool that allows for the targeted analysis of sequences and offers direct residue level control in the editing, trimming, shading, and export of multiple sequence alignments. The extensive functionality implemented in SlimShadey takes advantage of JavaFX and includes complete editing (inserting, deleting, degapping, subsequence export), real-time shading (identity, similarity, dissimilarity, counts, matrix), annotation (domains, motifs, logos), and exporting of publication quality images in PNG and RTF format. This manual and associated videos use the GLD-1 or CD44 protein sequences (reviewed in Lee and Schedl 2010 and Senbanjo and Chellaiah, 2017, respectively) to illustrate the major tools contained in the package. Additional sample sequences are provided to illustrate the range sequence alignments that can be manipulated. All of the calculation, shading, and editing of sample sequences is performed in real-time.

Requirements

SlimShadey will run without modification on Windows 7 (or higher) and will also run on OS X / macOS, most Linux distributions, and servers when Java 8 JRE is installed (download from Oracle). Internet access is not required for SlimShadey, except for ScanProsite- and Hmmscan-based shading. SlimShadey will also run on most server platforms that support Java and Chromebooks that support Linux. For information on how to run SlimShadey on Linux / Chrome or on a server please see the sections after the FAQ.

Before you begin

Using the command line (terminal in OS X, cmd in Windows, shell in Linux), run the command “java -version” to check your version of Java JRE. SlimShadey requires Java 1.8.0 or higher. If you have this version installed then SlimShadey should run without any additional modifications.

SlimShadey may be partially functional if other versions of Java are also installed; however, if you would like to use all functionality then download the Java Uninstall Tool from Oracle and uninstall all versions of Java that you have currently installed and reboot. Once you have fully uninstalled Java and rebooted, reinstall Java 8 JRE to restore the path variable. You can also fix this error by directly installing or editing the path variable.

Quick start video tutorials

To use this tutorial you will need Java 8 JRE installed, at least 2GB RAM, the “SeqVerter.jar” file downloaded, and 1MB of free disk space. Videos cover the following topics:

- Checking Java version and installing Java (on Mac)
- Performance on PC and on Mac
- Sample data for p53 and HOXA1
- Getting input to SlimShadey
- Coloring using Rasmol and changing the default color scheme
- Basic shading options
- Annotation
- Shading with a substitution matrix
- Shading with ScanProsite, Hmmscan, and regular expressions
- Sequence cropping
- Subsequences to a new tab, degapping, and realignment
- Removing sequences
- Removing gap-only or gap-containing columns
- Editing individual residues
- Sequence logos
- Presentation/publication quality output
- Output to and read in SlimShadey format alignments

Performance

SlimShadey has minimal RAM requirements (2GB), does not require a dedicated graphics card, and will run on legacy systems provided they support Java 8 JRE. However, for larger datasets and a smoother experience, the availability of 16GB RAM, dedicated graphics card (for example, NVIDIA GTX 1070), and quad-core processor or higher (for example, Intel Core i7-8700) will significantly improve performance. In addition, performance scales inversely with the number of characters displayed (i.e. the number of sequences x the number of characters per sequence); in larger alignments, selecting columns of interest and exporting them to a new tab, so that fewer characters are displayed, will dramatically improve performance. In addition, taking advantage of the fully reversible “collapse identical sequences" function (see below) can dramatically improve performance on legacy hardware when dealing with larger datasets with multiple identical sequences.

When it is not necessary to view the entire dataset on the screen or the rendering is not smooth during processing, additional performance can be gained by using lower the output resolution of your monitor. For example, using 720p (1280x720), 1080p (1920x1080), or 1440p (2650x1440) will result in smoother performance versus 4k (3840 x 2160) or higher. For most applications, we recommend setting the screen output resolution to 1080p, however, any common resolution may be used as Slimshadey will scale the default window to ¾ of the height and width of the user's screen output resolution.

Our recommendations for large datasets (>250 sequences x 500 characters) are as follows:

- CPU: Intel Core i7 8700K or AMD Ryzen 1700X
- GPU: NVIDIA GTX 1060 or AMD Radeon RX 580
- RAM: 16GB

For very large datasets, the memory allocation for the JVM may have to be altered (see FAQ). Sample data has been included to test the performance of the UI.

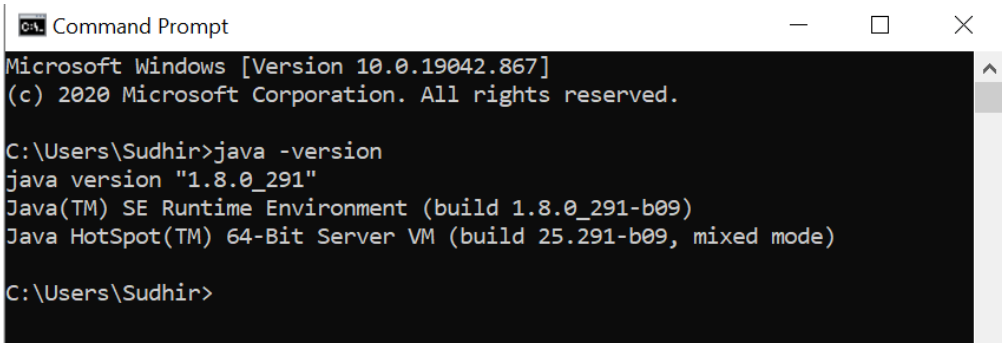
There is no output penalty for working at lower screen resolutions, other than the text may not be as clear on your screen. The resolution of the output alignment (as produced by the 'print preview' option) is dictated by user-defined font size, and NOT the screen resolution. The output font size can be changed in the file menu in the print preview window if a high-resolution printable or projectable image is desired. As such, SlimShadey's performance may be improved by reducing the screen resolution, without affecting the resolution of the final graphical output.

Checks to run before starting

On your PC or server, from the terminal check your Java using "java -version". This should return something similar to the following:

```
java version "1.8.0_291"
Java(TM) SE Runtime Environment (build 1.8.0_291-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.291-b09, mixed mode)
```

An example of using the Command Prompt from Windows 10 is below. Some of the version numbers (e.g. 291-b09) will change based on updates. The same process is used to check the version through the Terminal on Mac OS X or Linux distributions.



If your system meets or exceeds the requirements above then SlimShadey will operate on your system. If the "java -version" command does not work on your system but you are certain that you have installed Java and SlimShadey still runs then see the FAQ for an explanation.

Input file formats

The input file format for SlimShadey is sequential FASTA (aligned or unaligned). Alignments in other formats can easily be converted to FASTA using SeqRet (Madeira et al, 2019). For unaligned sequences, SlimShadey has been packaged with OpalFX (Wheeler and Kececioglu,

2007) to allow for the rapid alignment of protein and nucleic acid sequences. SlimShadey supports the direct cut/paste of FASTA files from NCBI, Ensembl, and other databases. Multiple files can also be imported using drag and drop into SlimShadey's UI for convenience.

Example input format (partial GLD-1 FASTA OPAL aligned):

```
>GLD-1_Caenorhabditis_latens
FGVTSQLETQNAESPSRSSILTPTSLEDETSPRKRFPLIETNISASDRWPPAPRRDGWSS

>GLD-1_Caenorhabditis_briggsae
FGVSAQHENPSVDSPSRSSILTPTSLDDETSPRKSFQILESSVS-ADRWP-APRRDGWSS

>GLD-1_Caenorhabditis_nigoni
FGVSAQHENPSVDSPSRSSILTPTSLDDETSPRKGFPILESSVS-TDRWP-APRRDGWSS
```

SlimShadey also accepts the Shadey project file format (extension `.mm`), which stores all sequence, annotation, consensus, and shading/styling information. This allows users to share their alignment markup/workflow and preferred session settings, to work on an alignment collaboratively or over multiple SlimShadey sessions.

How to input files to SlimShadey

Files can be imported to SlimShadey in three ways.

- Aligned sequences can be dragged and dropped into SlimShadey's UI directly
- Aligned or unaligned sequences via a file explorer, accessible through the 'Start' menu
- Individual FASTA-format alignments can be copied and pasted into SlimShadey via the 'Start' menu

Note that multiple files can be input simultaneously via the file explorer and drag/drop; however, this does not apply when cut/paste of FASTA sequences is being used.

SlimShadey Format (.mm)

The .mm format has been designed to hold all of the information from a SlimShadey project. The text file holds all sequence information plus annotations and complete shading (including position, color, shade) information. You will not need to directly edit the contents of .mm files, as they can be edited by opening in SlimShadey and directly editing the alignment's contents, then saving as a new .mm file.

File structure:

- The first 5 characters of a `.mm` file are "`.slim`".
- Data fields begin with "`$`" followed by "`field_name`" (e.g. `fontsize`, `collapse`, `luminance`, etc.), and, within curly brackets ("`{`" and "`}`") the data for that field. For

example, to denote that the given alignment's UI font size is 24, a line in the file would be: `$fontsize{24}`

- Sequences and annotations are stored with one character per line, and separated by sequence names. For example, valid sequence data for the following FASTA alignment:

```
>seq1
AY
>seq2
AW
```

- With one empty (i.e. all space characters) annotation row, and shaded with white background and black text, the alignment would be saved as:

```
$annotation_data{
%EMPTY_ANNO_1
 .#FFFFFF.#000000
 .#FFFFFF.#000000
}
$sequence_data{
%seq1
A.#FFFFFF.#000000
Y.#FFFFFF.#000000
%seq2
A.#FFFFFF.#000000
W.#FFFFFF.#000000
}
```

Sample data provided

Multiple aligned and unaligned sample files are provided to illustrate functionality. Aligned sequences are ported directly to SlimShadey, whereas unaligned sequences are passed to Opal for alignment. After alignment, aligned sequences can be imported to SlimShadey.

Parameter file

SlimShadey always initializes with default shading rules for each method (i.e. ClustalX, ScanProsite, RASMOL, etc.), even if an alignment given in SlimShadey's .mm format file is edited by a different user who has changed their default settings. Note that the previous user's shading can be retained during input. For example, when clicking RASMOL shading, choose 'Retain' when prompted to clear shading, and their shading will still be accessible; however, your RASMOL remote will open with your default settings.

For users who require custom default shading, the package includes a parameter file that can be easily modified with a text editor to change default rules for each alphabet without the need for recompiling. For example, new amino acid groups can be defined, the matrices can be modified or new ones added, consensus rules changed, default coloring modified, and

non-standard characters can be added. While default coloring RGB values (in hex format) and consensus rules can be changed without any issues, if you add a new character, it must be represented in each substitution matrix.

Unzip the .jar file, and go to the sub-directory:
`[edu/tcnj/seqverter/sequence/]`
In the `[.../sequence/]` directory, there are `*.salpha` files, which contain the rules. For example, to change the rules for protein sequences, the user can edit the `Protein.salpha` file.

File menu options

Rename tab

Allows the user to rename the selected tab. Custom tab names allow for streamlining of workflow, by specifying the contents of each open alignment, and giving a default filename when the project is exported to FASTA, RTF, PNG, or the Shadey project format. Each tab is treated as a different project and must be saved separately. The number of tabs that can be opened during a session is limited by the total memory allocated to the JVM (see troubleshooting).

Realign sequences (align sequences)

SlimShadey is provided with OpalFX, an interface to the Opal multiple sequence alignment algorithm (Wheeler, 2007), as a fast, accurate, and convenient sequence alignment tool. OpalFX can be used to align unaligned sequences and to realign a multiple sequence alignment already loaded into a tab. OpalFX also supports the translation and alignment of nucleotide sequences with all 30+ known genetic codes, as well as codon-delimited alignment. Once alignments have been completed, the user can directly import the results to SlimShadey or save the alignment in FASTA format from the file menu.

Print preview

Print preview allows for WYSIWYG formatting and rendering of publication-quality images. The multiple sequence alignment is organized into rows, and the user is given control over numbering, truncation of names, font size, number of columns per row, and whether to include annotations or a consensus sequence. The output can be saved as a PNG file for downstream analysis for further annotation in image processing software.

- *NOTE:* PNG resolution is based on font size. Increasing the font size will increase the output resolution. As such, if additional resolution is necessary then font size can be adjusted to meet your requirements.

Output to RTF (rich text format)

Exporting the contents of a tab in RTF format allows the output of SlimShadey to be opened and edited manually in word processing software (e.g. MS Word, OpenOffice, Google Docs).

Exporting to RTF allows users to directly paste a styled/formatted alignment into other documents. This format also allows the user to change (add, insert, and delete) characters after the contents of a tab have been output without using the SlimShadey interface. As output options, the user is provided control over font size, name truncation (i.e. number of characters in name), and whether to include annotations or numbering.

- *NOTE:* The RTF is saved with all color and formatting specified exactly as in the SlimShadey tab. Some word processors may not display the RTF properly. For best results, we recommend Microsoft Word.

Save alignment as text

SlimShadey allows any tab to be exported to a flat text file for use in other applications. The pulldown menu under the file name: "Save as type" allows the selection of multiple supported formats for export. Additional formats can subsequently be generated using SeqRet (Madeira et al, 2019).

Save as project

SlimShadey outputs projects to the .mm format, allowing complete sessions to be shared. The "*.mm"* format saves the alignment along with all shading, annotations, and consensus sequence, session settings, including UI font size, the first index of the alignment, luminance threshold for text coloring, the alphabet of the alignment (RNA, DNA, or protein), and whether to collapse identical sequences.

Edit menu options

Session settings

- *First index:* Sets the index of the first column of the alignment to the specified value. By default, the first column is '1'. For example, if you have a portion of a larger sequence, you may want to change this value to better reflect the position of the sequence.
- *Font size:* Changes the UI font size to the specified value. The font size is 18px by default, to display as much of the alignment as clearly as possible on most machines. SlimShadey renders all characters in real-time so changes in font size will change the number of characters displayed on the screen and may alter the performance of the UI.
- *Luminance threshold:* The luminance of a color can be calculated from its RGB color space. Depending on whether the shading color's luminance is above or below the threshold value, the text color is set to either black or white. The user can change the text color to any RGB value manually, but built-in shading/coloring algorithms will only color text black or white when shading is performed, depending on the background shading assigned. After shading is performed, the user has residue-level control over the color of the letter and the background. The luminance threshold is by default 0.179, which generally assigns white or black text color appropriately based on the background shading color. Changing this value is not recommended.

- *Hover refresh rate:* This number (in milliseconds) ranges from 0 to 999, and is 250ms by default. Lower refresh rates require more robust hardware as calculations will be performed more frequently. We recommend 250ms or greater for systems using integrated graphics. If you notice a substantial lag in hovering over residues in the alignment, increasing this value may help reduce this lag, lower system load, and improve performance. It allows the user to fine-tune the interface to match their system capabilities.

Gap only columns

SlimShadey can identify gap-only columns so they can be removed by the user or avoided prior to exporting subsequences.

Calculate consensus

SlimShadey can calculate three types of the consensus sequence (Simple, Dots and Identity, and ClustalX). Simple consensus outputs a lowercase character for the majority in a column and an uppercase when characters are identical in a column. Dots and identity use the ClustalW rule set to calculate the consensus (Thompson, 1994). ClustalX consensus uses the rule set by Jeanmougin et al., 1998. These can be viewed in ClustalX's manual, under the 'Color' heading.

Add annotation row

This function allows the user to add an annotation row. For detailed instructions on editing, moving, deleting, or adding annotations, please refer to the section on **Directly Editing the Alignment**. Annotation rows are useful for highlighting features of the alignment, such as regions with conserved structures, outstanding residues, insertions, or deletions of interest. Annotations can be added by individual column or by region (block annotation over multiple columns). For an annotation over a single column, right-click above the region of interest and select "Annotate selection" from the pull-down menu. If you want to annotate a region, right-click-and-drag over the region desired and select "Annotate selection". Commonly used characters are available for copy/paste used in the annotation window. For regional annotation, the single character will be copied over all columns. For example, to define an alpha-helical region, the user can cut-paste the 'α', which will be copied over all columns selected. If multiple characters are used then the characters will repeat over the region, e.g if 4 columns are selected, and 'α' is entered, the annotation 'αααα' will be applied.

Collapse identical sequences

This function collapses identical sequences into groups, which can be unpacked when analysis has been completed without any loss of data. This function works by performing an all-against-all comparison for identity and only displays unique sequences. When a group of identical sequences is collapsed into a group, only the shading and text color of the first sequence is saved; when uncollapsing the sequences, the shading that was applied to the collapsed sequence is applied to all constituent sequences.

SlimShadey performs real-time calculations on the alignment as rendered. For example, if there are multiple identical sequences that result in the overrepresentation of characters when the collapse function is called and they are reduced to a single sequence, the same characters may now be underrepresented. More specifically, if you have 5 sequences, 3 of which are identical, then the 3 identical sequences will be a majority when uncollapsed, and will not be a majority after collapsing.

When working with large datasets that may have many identical sequences, using this function can improve the performance of SlimShadey. For example, rapidly identifying low-frequency polymorphisms in a population benefits from not rendering uninformative sequences.

Shade menu options

Clear shading

This function clears all shading and returns the input sequence to the default settings.

Shade by unique residues

To reveal outstanding residues, this function highlights unique residues in a column. For flexibility, the remote allows adjusting the maximum number of residues in the column. This allows restricting shading to those residues within a column whose count is below a threshold count. This is an easy and sensitive method for determining interesting polymorphisms in sequence alignments. The ability to store logs of differing residues by position, and visualize the locations and apparent densities of these differences, may be helpful in directing further analysis and generating testable hypotheses.

Shade by residue frequency

This function allows shading to a simple majority within columns. The threshold frequency for calculating the consensus can be adjusted using the slider, to between 0.5 and 1, to change shading in real time. Within a column, if the proportion of a single residue exceeds this threshold frequency, the residue is shaded black to denote identity, and similar residues gray to denote similarity; if the total proportion of different, but physically or chemically similar residues exceeds the threshold, those similar residues are shaded gray to denote similarity (e.g. if in a column neither leucine nor isoleucine alone exceed the threshold frequency, but together do, both are shaded ‘gray’). A threshold between 0.5 and 1 can be set by the user. The lower limit of 0.5 is set to avoid determining more than one consensus residue within a column.

This color scheme can be inverted (Inverse shading) to produce ‘inverse shading,’ in which residues to be shaded black are left white, residues to be left white are shaded black, and residues to be shaded grey are shaded grey; color inversion is particularly useful for demonstrating dissimilarity or lack of conservation.

Shade by sequence

Shade by sequence allows for shading matches, similarities, and differences relative to a master sequence selected by the user. The master sequence can be any of the sequences in the alignment. By default, the master sequence is not shaded; however, this can be changed to any color from the remote. This function is useful for revealing the identity to a specific lineage.

Shade with substitution matrix

Sequences can be shaded with a relative match to one of the BLOSUM matrix series (30-100), Gonnet250, PAM250, or identity. For each substitution matrix, a shading matrix is calculated based on two RGB values, one for a perfect match and one for the worst possible match. For each residue, this corresponds to the maximum (i.e. identity) and minimum score possible, respectively. For example, when using BLOSUM62, and comparing a residue to alanine (A), the maximum score is when aligned to another A, at +4. This represents a perfect match and is assigned the match color. Similarly, using BLOSUM62, alanine aligned with tryptophan (W) results in the minimum score of -3, and produces shading closest to the mismatched color. As the residue changes from the worst possible match and the score of the aligned residue increases, the RGB values scale linearly to the matching color. Comparisons to a gap character (-) always result in no shading.

Custom Prosite regex shading

SlimShadey allows the shading of sequences using matches to regular expressions using SwissProt's ScanProsite syntax. The SwissProt regular expression syntax can be downloaded here or viewed here. Each sequence in the alignment is scanned for a match to the pattern, regardless of position, and matches are previewed in a graphical window. Any matches can be applied to the alignment and viewed directly on the sequence alignment. SlimShadey's scanner disregards gap characters, e.g. the pattern '`MP`' will match a subsequence given by '`...ARM--PLC...`' by ignoring gaps. In addition, the scanner supports the use of multiple regular expressions simultaneously, which allows for the mapping of multiple motifs in a single scan. Using the extended ScanProsite syntax, simple strings can be rapidly searched. For example, '`MPSC`' is the same as the full expression in ScanProsite syntax, '`M-P-S-C`'.

Prosite-based database shading

Prosite-based shading retrieves matches from prosite and renders them on the alignment (Hulo, 2004; Sigrist, 2013). The default implementation uses ExPASy's standard settings and filters low complexity patterns and includes ProSite profiles in the output. The hits can be applied to the alignment, where all related domains will be rendered in the same color. Information about the pattern, including ScanProsite hit documentation, can be retrieved by clicking on the colored region of the diagram. Sequences are degapped prior to scanning so that hits will be rendered even if there are gaps or the hits are present at different positions.

hmmscan based shading

The hmmer sequence analysis package implements an algorithm for finding motifs and domains within sequences using profile HMMs (Potter et al., 2018). SlimShadey uses an online implementation of hmmscan, hosted at the EBI (https://www.ebi.ac.uk/Tools/hmmer/search/hmmscan) to identify motifs in each sequence in the alignment. Motifs are all from the Pfam database (Potter et al., 2018), and documentation can be retrieved for these within SlimShadey. By default, all hits to the same accession are assigned the same RGB value for shading. Each motif's default color can be changed independently, and individual motifs can be shown or hidden on the alignment. All positional information is carried from hmmscan to SlimShadey and retained on the alignment, even when gaps are present within a motif.

Default RASMOL coloring

Default coloring of alignments uses the RASMOL color scheme (Sayle et al, 1995), which uses contrasting colors to distinguish residues with different physicochemical properties.

Default ClustalX coloring

This function calculates a consensus and applies shading using the rules from ClustalX for protein sequences (Larkin, 2007). Each amino acid is shaded according to its identity and the consensus character for its column. The rules for this are described in the '`colorprot.par`' parameter file distributed with ClustalX.

NOTE: If you plan to shade in multiple layers, the dynamic coloring and shading of characters implemented in JavaFX requires a specific order of operations.

1) Use either: shade by unique residues, shade by residue frequency, shade by sequence, default RASMOL coloring, or default ClustalX coloring. Updating the properties of these shading modalities will overwrite all shading in the alignment.
2) Once one of the aforementioned shading modes have been used as a 'base' for the alignment, domains or motifs may be highlighted by using: shade with Prosite Motifs.

Directly Editing the Alignment

In SlimShadey, an alignment is split into two panes. The left pane holds the names of the sequences and the names of the annotation rows. The right pane holds the alignment and annotations. Clicking on a sequence or annotation row's name, and left- or right-clicking somewhere on the alignment, will open a menu with options to manipulate properties of the alignment.

Sequence-level control (left pane, clicking on sequence name or annotation name)

In an alignment in SlimShadey, the left pane has the names of the annotations and sequences. Clicking on the name of a sequence or annotation allows the user to rename, reorder, or delete the selection.

- *Annotations:* Clicking on an annotation allows the user to edit the annotation row's name, add a new annotation row, change the relative position of an annotation row (up or down), and delete an annotation row.
- *Sequence names:* Similar to annotation rows, sequences can be renamed, reordered, or deleted by clicking on the name. Sequences can also be reordered relative to a chosen sequence, based on pairwise scoring between every other sequence and the chosen sequence with a substitution matrix - the chosen sequence is moved to the top, and sequences are reordered below in order of highest- to lowest-scoring.
- *Consensus:* If changes are made to the sequence alignment without realignment, the consensus can be recalculated by clicking on "`Consensus`" in the left pane at the bottom of the sequence alignment. If the sequences are realigned, the consensus sequence is automatically recalculated based on a simple majority.

Active functions (right pane, left-click or right-click on alignment)

SlimShadey allows for residue-level control of editing, shading, and recoloring of sequence alignments. Custom editing and shading can also be used with annotations. These options can be accessed by right- or left-clicking directly on a character in the alignment. This includes the following functions:

- *Residue-level control:* editing (changing, deleting, inserting) individual residues and their properties (color, shade)
- *Column-level control:* deleting individual columns or groups of columns, editing properties of groups of columns, moving subsequences to new tabs for downstream analysis, shading subsequences to indicate new motifs, domains, or regions

The active functions are implemented as follows:

- **Left-click** on sequence alignment activates the residue level edit function. This allows for the following processes:
    - *Edit:* From the edit strip at the bottom of the window the properties of the character can be changed. These properties include the character identity, shading color, and text color can be modified.
- **Right-click** on the sequence alignment activates the column editing function. This allows for the following processes:
    - *Insert left/right*: Allows for the insertion of a new column to the left or right of the position identified by the right-click (green shade column indicator).
    - *Delete selection:* This allows the deletion of individual columns or groups of columns defined with the right-click-and-drag functionality.
    - *Make sequence logo:* Sequences that are defined by the generation of sequence logos from subsequences right-click-and-drag can be converted to sequence logo in PNG format that can be printed or inserted in presentations.
    - *Annotate selection:* Allows block annotation of selected subsequences (see above, Add annotation row**)**

        - If 5 columns are selected and the user enters the annotation '`A`', then the annotation produced will extend the pattern to '`AAAAA`'. Similarly, if the user enters '`AB`', the annotation will be '`ABABA`'. This allows quickly adding an annotation for multiple columns with a fixed sequence, e.g. '⇒⇒ . . . ⇒⇒' by selecting those columns,
    - *Edit sequences*: This function allows for the column-level editing of individual residues
        - Like the option above, a smaller subsequence can be entered and extended (e.g. the user can select 3 columns, enter '`A`' for a sequence, and that sequence's characters will be '`AAA`' in those columns). Alternatively, the user can choose to infer
    - *Shade manually:* Manual shading provides control of shading individual sequences
    - *Block shading:* Fully manual control of shading over motifs, domains, novel regions, and sections of annotations
    - *Export selection:* The user can select right-click-and-drag to identify the subsequences to export. The selected sequence, with the names retained, will be exported to a new tab that is independent of the original. New shading rules and additional daughter tabs can be generated as desired and saved independently.

NOTE: An editing warning is included to warn the user of unintended changes that may alter the information content in the alignment. The user can turn off the warning and free-edit the sequence alignment to make multiple changes in sequence.

Troubleshooting and FAQ

What if SlimShadey fails to launch?

Use the terminal and use the “java -version” command to check your version of Java JRE. SlimShadey requires Java 1.8.0 or higher.

What if you have checked your Java version and SlimShady still does not launch?

In rare cases, Java JRE may need to be uninstalled and reinstalled. Follow the instructions for uninstalling all Java components here, reboot your system, reinstall Java (Oracle), and reboot your system again. Mac uninstall instructions can be found here.

The “java -version” command fails but SlimShadey still runs. Why is that?

The java -version command simply allows you to get information about the version of Java you have installed. When the command fails to return the expected version information it simply means the path has been corrupted in some way. SlimShadey will run even if this command fails if you have a recent version of Java installed. As such, the java -version command failing is not an issue if SlimShadey runs; however, if you would like to fix this error then download the Java Uninstall Tool from Oracle and uninstall all versions of Java that you have currently installed and reboot. Once you have fully uninstalled Java and rebooted, reinstall Java to restore the path variable. You can also fix this error by directly installing or editing the path variable.

What if SlimShadey is running slowly?

Slimshadey has been designed to run on legacy systems; however, it will run more slowly on systems that only support 32-bit versions of Java. Significant performance increases can be obtained by moving to hardware that can support 64-bit versions. The performance of SlimShadey can also be improved by using a system with at least 8GB of memory and a dedicated GPU. If larger datasets are being processed then SlimShadey will require additional memory (>16GB) and a faster CPU (e.g. Intel Core i7 or higher).

What is SlimShadey is running slowly on my new M1 Mac (Macbook Air, Macbook Pro)?

SlimShadey will run on any platform that supports JDK 8, which includes M1-based Macs; however, performance will vary based on the underlying support and optimization for the underlying hardware. The newer M1 processor-based Mac may gain performance improvements by installing an optimized version of Java such as Azul Zulu OpenJDK. Performance on Intel-based Macs is similar to Linux and PC (illustrated in the performance videos).

What if SlimShadey is running slowly on a dual monitor system?

This occurs when one monitor is using integrated graphics (e.g. Intel i5 with integrated HD graphics) and a second monitor is driven by a dedicated graphics card. In this case, SlimShadey may run significantly lower if it is rendered on the monitor supported by the

integrated graphics. The solution is to run both monitors off the dedicated graphics card if possible or move the SlimShadey window to the monitor supported by the dedicated graphics card. For Windows 10 laptops with multiple displays, you can configure these properties in Advanced display settings and specify the display for each monitor.

What if SlimShadey is running slowly on a system with updated hardware?
Depending on your system configuration you may have to increase the memory available to SlimShadey for better performance. Using the terminal from the SlimShadey directory launch the software using the following command:

```
java -Xmx16g -Xms16g -jar SlimShadey.jar
```

16g is the memory allocated to the JVM gigabytes. This value can be increased to the limit of the memory available on your machine (e.g. 32g, 64g, etc.). If you have less than 8GB of memory then use the following:

```
java -Xmx2048m -Xms2048m -jar SlimShadey.jar
```

2048m is the memory allocated to the JVM in megabytes (or 2 gigabytes). This value can be increased to the limit of the memory available on your machine in multiples of 1024.

How many tabs can I have open?
We have not restricted the number of tabs that can be opened at one time. Each tab is treated as a separate project but only the active tab is rendered in real time. Depending on your system resources and memory allocation, SlimShadey may run more slowly if a large number of tabs are open. Since only the active tab is rendered in real-time, having multiple tabs open does not impact the performance of SlimShadey unless system resources are limited.

Can I open multiple instances of SlimShadey?
Yes. It is possible to open multiple instances of SlimShadey, however, the additional rendering may compromise the performance of real-time functions. If you are working with multiple sequences, we recommend using different tabs in a single instance for the best performance.

What if I have a large dataset?
If you are working with a large dataset then you may need to increase the memory allocated to the JVM (Java Virtual Machine). See troubleshooting on how to get better performance on updated hardware (above). If you have access to a server with sufficient allocation of resources then you may be able to run SlimShadey without any input lag with extremely large datasets.

Is there a limit to the number of sequences SlimShadey can handle?

Slimshadey was designed for the laboratory scale analysis of sequence data; however, we have not limited the number of sequences that can be processed. The upper limit of sequence that can be analyzed will be determined by your system hardware, memory available to the JVM, and the type of processing (visualization vs alignment). If you have a genome-scale dataset then you will have to use a different software package. Based on the sample system used for this work (Intel Core i7 8750H, 16GB RAM, NVIDIA GTX 1060), 200 sequences with approximately 1000 characters each can be processed without significant lagging. Larger datasets or less robust hardware may result in some lag in the real-time processing functions.

Do I have to use OpalFX to align my sequences?
No. SlimShadey will accept any aligned or unaligned text file in FASTA format. We have provided OpalFX as an option if you want to align your sequence inside a SlimShadey session. This option is particularly useful if you export a subsequence to a new tab or delete a sequence from the original alignment and need to realign. Overall, the Opal algorithm has similar performance to MAFFT and Muscle (Wheeler, 2007).

What if the shading looks strange or unexpected?
SlimShadey is optimized for shading, highlighting, and editing of ***aligned*** nucleic acids and protein sequences. You can align the sequences using the algorithm of your choice and import the aligned sequences into SlimShadey as a FASTA file or you can use the included Opal FX to align the sequences in the tab you are working in.

What if the shading in my RTF file does not match the SlimShadey window?
SlimShadey adheres strictly to the rules for the RTF file format. Software such as Microsoft Word may not render all coloring, shading, and characters correctly. To guarantee an accurate representation in Microsoft Word RTF, use grayscale and standard UTF-8 characters. Most colors are supported, though some may be omitted or converted to grayscale by Microsoft Word. We recommend saving to a PNG file for WYSIWYG output.

When working with sequencing reads the collapse function does not work properly?
FASTA files of sequencing reads can contain variable numbers of unknown (N or X) characters at the 5' and 3' ends. SlimShadey treats sequences with any differences as unique and will not collapse them into a single group. Option 1: The 5' and 3' ends can be trimmed using the active right-click drag function to delete the columns with unknown residues. Option 2: The active right-click drag function can also be used to select a range of residues and transfer them to a new tab for subsequent analysis. Either approach will allow for the maximum reduction (grouping) of the dataset with reads with unknown residues are eliminated.

Can I modify SlimShadey and compile it on my own machine?

Yes, however, you will need a functional understanding of JavaFX. All source code for SlimShadey is freely available under GNU General Public License (GPLv3). SlimShadey was developed using NetBeans and the general steps are indicated below:

- File → New Project → Java with Ant → Java Application
- Name project SeqVerter and uncheck create main class
- Go to Documents → NetBeansProjects → SeqVerter folder
- Replace the src with the latest version (Desktop)

Can I write my own shading rules?

Yes, though this will require some knowledge of basic Java (no knowledge of JavaFX is required), and a standard procedure for mapping some characteristics of the alignment to RGB colorspace. For the identification of novel motifs and domains, SlimShadey supports shading using ScanProsite regular expressions. If you would like to write your own rules for shading, the rulesets are in the `edu.tcnj.biology.seqverter.sequence` package. Parameter files end in the extension .salpha, and the DNA.salpha file is shown here to demonstrate how to make a custom set of shading parameters for SlimShadey.

```
slim_alpha
$Name={DNA}
$Alphabet={(A,Adenosine,#A0A0FF),...,(C,Cytosine,#FF8C4B),(N,ANY,#FFFFFF),(-,gap,#FFFFFF)}
$Categories={(Purine,AG),(Pyrimidine,TC),(ANY,N)}
```

The first line must be “slim_alpha” for the parameter file to be recognized, and each subsequent line must begin with the “$” character followed by the name of the parameter being defined. This is followed by the “=” character and immediately followed by the value(s) for this parameter inside “{ }” brackets.

The “Alphabet” used is defined by each character, its name, and the default shading color associated with it. The default colors for DNA here are based on Rasmol’s color scheme, and the “gap” and “any nucleotide” characters “-” and “N” default to no shading (#FFFFFF is the hex representation of white).

The “Categories” are used by certain shading modalities when defining a weak match or consensus, for example, if a column in the alignment has a threshold for the majority of “Purine” (“A” or “G”), but not a majority of “A” or “G” independently, the basic shade option may shade purines gray.

There are further parameters for defining consensus rules. The consensus-building follows ClustalX’s rulesets. These have been applied for the protein parameter files, and are as follows:

```
$Consensus_alphabet={%#-+gnqptACDEFGHIKLMNPQRSTVWY}
```

```
$Consensus_rules={(%,0.6,WLVIMAFCYHP),(#,0.8,WLVIMAFCYHP),(-,0.5,ED),(+,0.6,KR),(g,0.5,G),(n,0
.5,N),(q,0.5,QE),(p,0.5,P),(t,0.5,TS),(A,0.85,A),(C,0.85,C),(D,0.85,D),(E,0.85,E),(F,0.85,F),(
G,0.85,G),(H,0.85,H),(I,0.85,I),(K,0.85,K),(L,0.85,L),(M,0.85,M),(N,0.85,N),(P,0.85,P),(Q,0.85
,Q),(R,0.85,R),(S,0.85,S),(T,0.85,T),(V,0.85,V),(W,0.85,W),(Y,0.85,Y)}
$Color_define={(RED,#f01505),(BLUE,#80a0f0),(GREEN,#15c015),(CYAN,#15a4a4),(PINK,#f08080),(MAG
ENTA,#c048c0),(YELLOW,#c0c000),(ORANGE,#f09048)}
$Color_consensus_rules={(G,ORANGE/%#-+gnqptACDEFGHIKLMNPQRSTVWY),(P,YELLOW/%#-+gnqptACDEFGHIKL
MNPQRSTVWY),(T,GREEN/tST%#0),(S,GREEN/tST#),(N,GREEN/nND),(Q,GREEN/qQE+KR),(W,BLUE/%#ACFHILMVW
YPp),(L,BLUE/%#ACFHILMVWYPp),(V,BLUE/%#ACFHILMVWYPp),(I,BLUE/%#ACFHILMVWYPp),(M,BLUE/%#ACFHILM
VWYPp),(A,BLUE/%#ACFHILMVWYPpTSsG),(F,BLUE/%#ACFHILMVWYPp),(C,BLUE/%#ACFHILMVWYPp,PINK/C),(H,C
YAN/%#ACFHILMVWYPp),(Y,CYAN/%#ACFHILMVWYPp),(E,MAGENTA/-DEqQ),(D,MAGENTA/-DEnN),(K,RED/+KRQ),(
R,RED/+KRQ)}
```

The “Consensus_alphabet” is the set of valid characters used when building a consensus sequence. These will be accessed later by the consensus builder when assessing a column’s sequence contents.

The “Consensus_rules” define in triplets three values; the consensus character from the alphabet, the proportion of residues in a column that must satisfy the rule to apply that consensus character to the column, and the characters in the column that satisfy the rule. For example, the value in this parameter “`(+,0.6,KR)`” signals the consensus builder to place a “+” character at this column of the alignment as long as 60% of the residues in the column are either “K” or “R”. The order of values in this rules parameter matters. Later values will overwrite the consensus character if an earlier value’s rule was also satisfied. For example, the parameter values “`(t,0.5,TS)`” and “`(T,0.85,T)`” can both be satisfied by a column composed entirely of “T” residues, but because the second parameter occurs later in the list, “T” will be the consensus character instead of “t” (as should be expected for a column consisting entirely of “T”s).

The “Color_define” defines colors by name and their corresponding hex representations. These colors must be defined by name to be used by the consensus coloring scheme.

The “Color_consensus_rules” allows SlimShadey to take advantage of the consensus built according to the first two parameters, to shade the entire alignment. For a given value, a color/consensus pairing is formed and used to shade a given residue. For example, the value “`(S,GREEN/tST#)`” dictates that a “G” residue in the alignment should be shaded if the consensus character for its column is one of “t”, “S”, “T”, or “#”; further, it should be shaded “GREEN” (which was defined as a value in previous parameter by its value “`(GREEN,#15c015)`”). Note that in this parameter’s values, the 2nd and 3rd values are separated by a “/” character and not a “,”.

Finally, default substitution matrices can be defined for a given set of parameters.

```
$MATRIX={(BLOSUM62,ARNDCQEGHILKMFPSTWYVBZX*,10,1,10,1,[(4,-1,-2,-2,0,-1,-1,0,-2,-1,-1,-1,-1,-2
,-1,1,0,-3,-2,0,-2,-1,0,-4),(-1,5,0,-2,-3,1,0,-2,0,-3,-2,2,-1,-3,-2,-1,-1,-3,-2,-3,-1,0,-1,-4)
```

```
,(-2,0,6,1,-3,0,0,0,1,-3,-3,0,-2,-3,-2,1,0,-4,-2,-3,3,0,-1,-4),(-2,-2,1,6,-3,0,2,-1,-1,-3,-4,-
1,-3,-3,-1,0,-1,-4,-3,-3,4,1,-1,-4),(0,-3,-3,-3,9,-3,-4,-3,-3,-1,-1,-3,-1,-2,-3,-1,-1,-2,-2,-1
,-3,-3,-2,-4),(-1,1,0,0,-3,5,2,-2,0,-3,-2,1,0,-3,-1,0,-1,-2,-1,-2,0,3,-1,-4),(-1,0,0,2,-4,2,5,
-2,0,-3,-3,1,-2,-3,-1,0,-1,-3,-2,-2,1,4,-1,-4),(0,-2,0,-1,-3,-2,-2,6,-2,-4,-4,-2,-3,-3,-2,0,-2
,-2,-3,-3,-1,-2,-1,-4),(-2,0,1,-1,-3,0,0,-2,8,-3,-3,-1,-2,-1,-2,-1,-2,-2,2,-3,0,0,-1,-4),(-1,-
3,-3,-3,-1,-3,-3,-4,-3,4,2,-3,1,0,-3,-2,-1,-3,-1,3,-3,-3,-1,-4),(-1,-2,-3,-4,-1,-2,-3,-4,-3,2,
4,-2,2,0,-3,-2,-1,-2,-1,1,-4,-3,-1,-4),(-1,2,0,-1,-3,1,1,-2,-1,-3,-2,5,-1,-3,-1,0,-1,-3,-2,-2,
0,1,-1,-4),(-1,-1,-2,-3,-1,0,-2,-3,-2,1,2,-1,5,0,-2,-1,-1,-1,-1,1,-3,-1,-1,-4),(-2,-3,-3,-3,-2
,-3,-3,-3,-1,0,0,-3,0,6,-4,-2,-2,1,3,-1,-3,-3,-1,-4),(-1,-2,-2,-1,-3,-1,-1,-2,-2,-3,-3,-1,-2,-
4,7,-1,-1,-4,-3,-2,-2,-1,-2,-4),(1,-1,1,0,-1,0,0,0,-1,-2,-2,0,-1,-2,-1,4,1,-3,-2,-2,0,0,0,-4),
(0,-1,0,-1,-1,-1,-1,-2,-2,-1,-1,-1,-1,-2,-1,1,5,-2,-2,0,-1,-1,0,-4),(-3,-3,-4,-4,-2,-2,-3,-2,-
2,-3,-2,-3,-1,1,-4,-3,-2,11,2,-3,-4,-3,-2,-4),(-2,-2,-2,-3,-2,-1,-2,-3,2,-1,-1,-2,-1,3,-3,-2,-
2,2,7,-1,-3,-2,-1,-4),(0,-3,-3,-3,-1,-2,-2,-3,-3,3,1,-2,1,-1,-2,-2,0,-3,-1,4,-3,-2,-1,-4),(-2,
-1,3,4,-3,0,1,-1,0,-3,-4,0,-3,-3,-2,0,-1,-4,-3,-3,4,1,-1,-4),(-1,0,0,1,-3,3,4,-2,0,-3,-3,1,-1,
-3,-1,0,-1,-3,-2,-2,1,4,-1,-4),(0,-1,-1,-1,-2,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-2,0,0,-2,-1,-1,-1,-1
,-1,-4),(-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,-4,1)])}
```

The “MATRIX” parameter is the only parameter that may appear multiple times in the salpha file; matrices are differentiated by their name, which is the first value. The second value is the order of characters in the rows and columns of the matrix; the characters used here must be a superset of the valid characters defined in the “Alphabet” parameter. The four numbers following the alphabet-order value are, respectively, gap-open, gap-extend, terminal-gap-open, and terminal-gap-extend values used by Opal for aligning sequences using this matrix. This is followed by the matrix itself, where the entire matrix (and not just the upper triangular form) is given, row by row. The matrix is enclosed by “[ ]” brackets, and each row is enclosed in “( )” brackets and separated by “,” characters. Each row should have a number of elements equal to the number of characters defined in the row/column order.

Note that each parameter begins with a “$” and is only a single line. Introducing line breaks will cause the file to not be parsed correctly. The format and order of parameters should exactly match the description here.

Once a custom alphabet parameterization has been made, it must be named (e.g. “`Custom.salpha`”) and loaded to the “`edu.tcnj.biology.seqverter.sequence`” package in Netbeans. The following lines may be added to the `DefaultAlphabets.java` class’s constructor:

```
this.add(load(this.getClass().getResourceAsStream("Custom.salpha")));
```

and the newly compiled SlimShadey will now be able to read, align, shade, and produce consensus sequences for any alignment satisfying your new alphabet/parameterization!

Can I modify classes and data files?

Yes. SlimShadey is provided with all source code. Classes can be modified and recompiled to customize options or add functionality to SlimShadey. Slimshadey was developed using JDK 8

and the NetBeans IDE. If you would like to edit it in the same environment as it was developed then follow the do the following:

- Delete NetBeans if it has been previously installed
- Delete all Java versions, JDKs, and updates.
    - NOTE: If you delete these first then Netbeans uninstaller for Windows 10 will not work. If you deleted them then you need to reinstall them and then delete Netbeans - then delete all Java versions
- Restart
- Install JDK 8u111 with NetBeans 8.2 from Oracle downloadable at this link or any standalone installation of NetBeans 8 or above.
    - JDK 8 and NetBeans 8.2 can be obtained separately; however, this installs them together so you don't have to set the paths later and the versions are known to be compatible

Why does SlimShadey run on Java 8, when Java's site shows the most recent version is on 16+?

Various versions of Java have been released over the past few years, but only some will continue to be supported. With Java 9 onwards, software developed using Java needed to be deployed with a custom Java runtime environment (JRE) and operating system (OS) dependent installers. Java 8, and all previous versions, required users to download, at a minimum, the JRE; users who downloaded the Java development kit (JDK) were automatically provided with JRE. This is why downloading Java 9 or higher will not allow you to run SlimShadey; these versions of Java are solely the JDK, and have no JRE. A secondary advantage to using Java 8 is that JRE 8 installations for various operating systems consistently run the SlimShadey.jar file correctly, and allow us to distribute a platform-independent executable version of SlimShadey. SlimShadey will run on newer versions of Java provided the three criteria are met:

- JRE has been installed (newer versions do not include this)
- JavaFX has been installed (newer versions do not include this)
- Dependencies and path variables appropriately defined

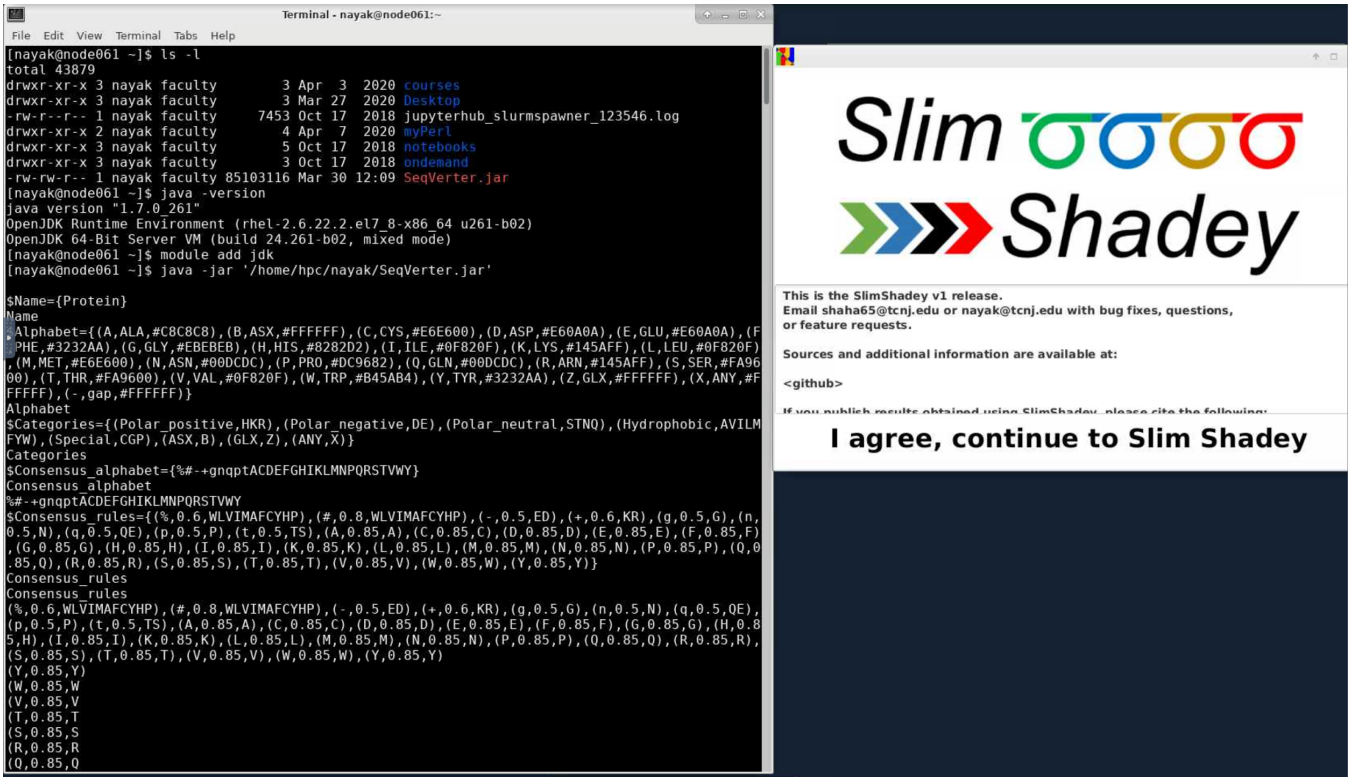
SlimShadey on a server

SlimShadey will run on most server platforms that support Java. From the terminal check your Java using "java -version" and output should be in the same general format provided as indicated below. Note that The College of New Jersey ELSA High-Performance Computing Cluster (TCNJ HPC) initial check results in Java version "1.7.0.261", which does not meet the minimum requirements for SlimShadey. The "module add jdk" updates the version to

“1.8.0.102”, which meets the minimum requirements.

```
Last login: Fri Jul  3 09:27:48 2020 from hpc-login-lb.hpc.tcnj.edu
[nayak@login001 ~]$ java -version
java version "1.7.0_261"
OpenJDK Runtime Environment (rhel-2.6.22.2.el7_8-x86_64 u261-b02)
OpenJDK 64-Bit Server VM (build 24.261-b02, mixed mode)
[nayak@login001 ~]$ module add jdk
[nayak@login001 ~]$ java -version
java version "1.8.0_102"
Java(TM) SE Runtime Environment (build 1.8.0_102-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.102-b14, mixed mode)
[nayak@login001 ~]$
```

Once the Java version minimum has been met, the SeqVerter.jar file is launched using “java -jar <path>”, in this case, the path is ‘/home/hpc/nayak/SeqVerter.jar’. Leaving the terminal console open provides error messages and information on data files used (left). The SlimShadey window running on the TCNJ HPC server is shown on the right.



Depending on the server resources and allocations, it may be possible to work with large datasets without collapsing any sequences or experiencing any input lag. The example below shows the sequence alignment of the HOXA1 protein family included in the same data as

“HOXA1 (large) shaded” with the default RASMOL shading.

All aspects of the alignment can be manipulated with no perceptible input lag on the TCNJ HPC, with all rendering performed on CPUs. At the time of this writing the TCNJ HPC has 113 servers, and 2,080 CPUs cores, with newer nodes composed of Intel Xeon Gold 6130 (16 core, 32 thread) CPUs with 192GB of RAM/node.

If the Java version on your server does not meet the minimum requirements for SlimShadey or you do not have the permissions to update Java, please contact your system administrator.

SlimShadey on Chrome / Linux

NOTE: Not all versions of Chrome OS support the Linux development environment and Linux distributions can vary so this may have to be modified for your specific implementation. SlimShadey is functional on Linux but has not been extensively tested so some functionality may not be fully supported. See the accompanying video for system information, checks, and the duration of the process without edits.

Check your Chromebook (if you know your system information then this can be skipped)

- **`chrome://system/`** (from the browser window)
  - This was implemented on Chrome version 91.0.4472.147
  - `Linux 5.4.109-26094-g381754fbb430 x86_64`
- Scroll down or search for “uname” to get your hardware specifications if you do not know them
  - The video tutorial uses a 2021 Acer 713 with an Intel i5 11th generation processor, 8GB RAM, and a 256GB SSD (7.5GB to Linux).
- SlimShadey also runs on Chromebooks with previous generation i-series and mobile processors provided the Linux development environment is supported; however, performance can be compromised.
  - This process has not been tested on Chromebooks with AMD (Ryzen) processors or non-standard installations of Linux.

Enable and check the Linux development environment

- From “Settings” search for Linux
- “Turn on” the development environment
  - Once the environment has been established the terminal should open by default
  - If the terminal does not open then use the search bar for “terminal”
- Check your Linux kernel version using: **`uname -srm`**
  - Should return something similar to the following:

```
Linux 5.4.109-26094-g381754fbb430 x86_64
```

- Check your Linux distribution using: **`cat /etc/os-release`**
  - Should return something similar to the following:

```
PRETTY_NAME="Debian GNU/Linux 10 (buster)"
NAME="Debian GNU/Linux"
VERSION_ID="10"
VERSION="10 (buster)"
VERSION_CODENAME=buster
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
```

Check current Java version

- **`java -version`**
- If it already says something similar to the output below then you can skip to "Make SlimShadey executable". *The numbers and letters in bold can vary based on the release date of the build.*

```
java version "1.8.0_291"
Java(TM) SE Runtime Environment (build 1.8.0_291-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.291-b10, mixed mode)
```

Get Oracle JDK8 from here.

- You will have to create an account (free for non-commercial use).
- This tutorial uses the "Linux x64 Compressed Archive" (jdk-**8u291**-linux-x64.tar.gz).
    - *The build number in bold will have to be changed in the instructions below so they can be executed.* SlimShadey may work with later versions of Java and OpenJDK if all the dependencies (JRE, JavaFX, etc.) are met, however, it has not been extensively tested. See FAQ for the advantages of using JDK 8 over other distributions.

Create /usr/lib/jvm

- Create a directory for the JVM using: **`sudo mkdir -p /usr/lib/jvm`**

Extract jdk-8u291-linux-x64.tar.gz (JDK download) and install in /usr/lib/jvm

- Move the jdk-8u291-linux-x64.tar.gz downloaded from Oracle to your main Linux directory
    - Check to see if it moved using: **`ls`**
- Extract the JDK and install using: **`sudo tar zxvf jdk-8u291-linux-x64.tar.gz -C /usr/lib/jvm`**
    - Be sure to change the file name to reflect the build you downloaded

Update and set

- Update Java alternative and set it as the new default using: **`sudo update-alternatives --install "/usr/bin/java" "java" "/usr/lib/jvm/jdk1.8.0_291/bin/java" 1`**
    - Be sure to change the file name to reflect the build you downloaded

Set the new JDK

- Set the JDK as the new default using: **`sudo update-alternatives --set java /usr/lib/jvm/jdk1.8.0_291/bin/java`**

Check Java version

- Check the Java version using: **`java -version`**
- Should output the following:

```
java version "1.8.0_291"
Java(TM) SE Runtime Environment (build 1.8.0_291-b10)
Java HotSpot(TM) 64-Bit Server VM (build 25.291-b10, mixed mode)
```

Get SlimShadey and make the file executable

- Download the SlimShadey.jar file from our GitHub and move it to your Linux directory.
    - Check to see if SlimShadey.jar moved to the correct directory using: **`ls`**
    - The file name may vary based on release date (e.g. early versions may be called SeqVerter.jar) but the process is the same.
    - At this point SlimShadey.jar is not an executable file. This can be checked using: **`ls -l`**
- Make SlimShadey.jar executable using: **`chmod +x SlimShadey.jar`**
    - Check to see if the it is executable using: **`ls -l`**

Run SlimShadey with the following

- To run SlimShadey.jar use: **`java -jar SlimShadey.jar`**
    - SlimShadey should launch with a few seconds

References

Hulo, N., Sigrist, C.J., Le Saux, V., Langendijk-Genevaux, P.S., Bordoli, L., Gattiker, A., De Castro, E., Bucher, P. and Bairoch, A., (2004) Recent improvements to the PROSITE database. Nucleic Acids Research, 32(suppl_1), pp.D134-D137

Jeanmougin F, Thompson JD, Gouy M, Higgins DG, Gibson TJ. Multiple sequence alignment with Clustal X. Trends Biochem Sci. 1998;23(10):403-405. doi:10.1016/s0968-0004(98)01285-7

Larkin MA, Blackshields G, Brown NP, Chenna R, McGettigan PA, McWilliam H, Valentin F, Wallace IM, Wilm A, Lopez R, Thompson JD, Gibson TJ, Higgins DG.
(2007). Clustal W and Clustal X version 2.0. Bioinformatics, 23, 2947-2948.

Lee MH, Schedl T. C. elegans star proteins, GLD-1 and ASD-2, regulate specific RNA targets to control development. Adv Exp Med Biol. 2010;693:106-122. doi:10.1007/978-1-4419-7005-3_8.

Madeira F, Park YM, Lee J, et al. The EMBL-EBI search and sequence analysis tools APIs in 2019. Nucleic Acids Research. 2019 Jul;47(W1):W636-W641. DOI: 10.1093/nar/gkz268.

Potter SC, Luciani A, Eddy SR, Park Y, Lopez R, Finn RD. HMMER web server: 2018 update. Nucleic Acids Res. 2018;46(W1):W200-W204. doi:10.1093/nar/gky448

Sayle RA, Milner-White EJ. RASMOL: biomolecular graphics for all. Trends Biochem Sci. 1995;20(9):374. doi:10.1016/s0968-0004(00)89080-5

Senbanjo LT, Chellaiah MA. CD44: A Multifunctional Cell Surface Adhesion Receptor Is a Regulator of Progression and Metastasis of Cancer Cells. Front Cell Dev Biol. 2017;5:18. Published 2017 Mar 7. doi:10.3389/fcell.2017.00018

Sigrist CJ, de Castro E, Cerutti L, et al. New and continuing developments at PROSITE. Nucleic Acids Res. 2013;41(Database issue):D344-D347. doi:10.1093/nar/gks1067

Thompson JD, Higgins DG, Gibson TJ. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res. 1994;22(22):4673-4680. doi:10.1093/nar/22.22.4673

Wheeler TJ, Kececioglu JD. Multiple alignment by aligning alignments. Bioinformatics. 2007;23(13):i559-i568. doi:10.1093/bioinformatics/btm226