

BracketMaker User Manual (v1_beta)



BracketMaker is a custom graphics program for protein chemists that quickly generates “bracket” style figures for chemical protein synthesis (CPS). These diagrams show the starting protein segments (typically produced by solid-phase peptide synthesis) and the order of all reaction steps, including sequential ligations (typically native chemical ligation, or NCL) joining segments two at a time into a full protein. CPS brackets are blueprints used to plan a protein synthesis, track progress throughout a project, and share results with others.

In BracketMaker, a bracket (i.e., a full CPS plan) is encoded in text format: a FASTA protein sequence with annotations describing all ligation reactions and side-chain modifications. BracketMaker converts this CPS plan to bracket image, which may be modified via user settings and saved in standard vector graphics (SVG) format. Additionally, BracketMaker is equipped with an “Auto-Bracket” tool to help users evaluate and compare different synthetic routes towards a given protein.

This user manual describes the procedure for creating a bracket (with illustrated examples) and details all of BracketMaker’s settings and features.

Version Note: This version of the manual is for the beta version of our program, published on GitHub prior to manuscript publication. All references to our “BracketMaker publication” refer to the manuscript in preparation.

TABLE OF CONTENTS:

<u>Section</u>	<u>Page #</u>
1: Installing and Running BracketMaker	2
2: Creating a Bracket	3
3: Saving/Exporting a Bracket	11
4: Comparing Multiple Brackets (Navigation Pane)	12
5: Modifying the Image (“Figure Settings” Menu)	13
6: Automatic Bracket Assembly (“Auto-Bracket” Menu)	16

1 INSTALLING AND RUNNING BRACKETMAKER

The BracketMaker executables for Windows and Mac, as well as the Python 3 source code, can be downloaded for free from our GitHub page:

<http://www.github.com/Kay-Lab/BracketMaker>

1.1 INSTALLATION ON WINDOWS

To use BracketMaker on Windows, download the *BracketMaker_Windows.zip* archive from the Releases tab on our GitHub. Extract the *BracketMaker.exe* executable anywhere on your computer and double-click to run.

Note: BracketMaker may take several seconds to launch. Upon launch, the program will create a 'bm_cache' folder in the executable's location; this folder is used to store temporary data while the program is running.

1.2 INSTALLATION ON MACOS

BracketMaker relies on the “Cairo” SVG graphics library for its image preview function. We are unable to bundle Cairo with the BracketMaker executable on MacOS, and thus Cairo must be installed separately via the Homebrew package manager by a user with admin permissions. Homebrew and Cairo are not affiliated with the developers of BracketMaker.

a. Install Homebrew on MacOS

- These instructions come from the Homebrew website, <https://brew.sh/>
- If logged into an admin account, open a Terminal window and copy/paste the following command. Enter your admin password when prompted, and allow several minutes for installation to proceed.

```
/bin/bash -c "$(curl -fsSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

- If not logged into an admin account, include “su” before the command to prompt for admin login information:

```
su /bin/bash -c "$(curl -fsSL  
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

b. Install Cairo on MacOS

- In a Terminal window, enter the following command. Cairo will be automatically installed by the Homebrew package manager.

```
brew install cairo
```

- (Optional) If you receive a “command not found” error, you must also install the Xcode Command Line Tools (usually included with Homebrew installation). Enter the following command, then follow system dialogs to install Xcode Command Line Tools:

```
xcode-select --install
```

c. Install BracketMaker on MacOS

- a. From the BracketMaker GitHub page, download the *BracketMaker_MacOS.zip* archive anywhere on your computer, and double-click to extract the contents (a single executable file named *BracketMaker_MAC*). Double-click the executable to run.

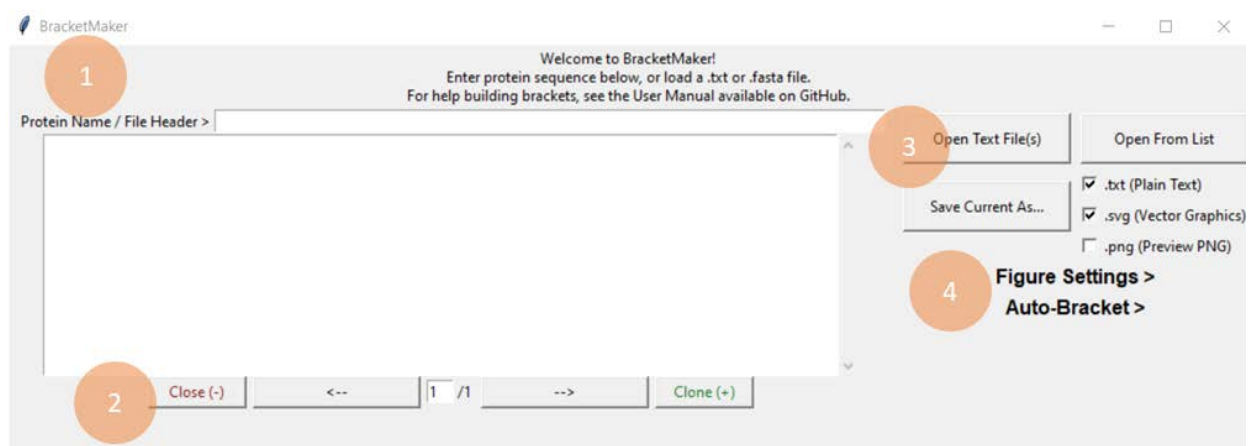
d. Security Settings (newer versions of MacOS)

- a. As we are a small academic team, we have not paid to enter the database of Apple-certified developers, and thus our software is unsigned; if BracketMaker sees more use, we will look into this signing ability. For now, modern versions of MacOS will flag any unsigned software and prevent it from running unless security settings are bypassed by the user. If you receive a security warning when running BracketMaker, please do the following:
 - i. In a folder containing the executable, ctrl-click *BracketMaker_Mac* and select “Open”.
 - ii. A prompt will appear saying “BracketMaker is from an unidentified developer. Are you sure you want to open it?” Press **OK** on this prompt.

Note: BracketMaker may take several seconds to launch. Upon launch, the program will create a 'bm_cache' folder in the executable's location; this folder is used to store temporary data while the program is running.

1.3 RUNNING BRACKETMAKER

Launching the program will open a terminal window and the BracketMaker graphic user interface (GUI) shown below (Windows version is used for all screenshots):



(1) Protein name and sequence text fields, (2) file navigation buttons, (3) buttons for opening and saving brackets, and (4) expandable menus with additional settings and functionality.

2 CREATING A BRACKET

To create a CPS bracket, one begins with a protein sequence (*Section 2.1*) and inserts special characters denoting ligation junctions (*Section 2.2*) and any additional side-chain reactions (*Section 2.3*).

As an illustration of the process, the following examples replicate the bracket for DapA, a 312-aa, 7-segment protein synthesized in the Kay lab (Weinstock, Jacobsen & Kay 2014, PNAS 111: 11679-84).

2.1 INPUT A PROTEIN SEQUENCE

To begin, either type a protein sequence into the main text entry box (using single-letter aa codes), or use the “Open Text File(s)” button to open one or more saved sequences.

When opening a sequence from a file, the file(s) must be in .txt or .fasta format, and the first line must begin with a “>” character; this header line is loaded into the “Protein Name / File Header” field, and the rest of the file is loaded into the sequence text box. Note that each protein must be contained in its own file; multi-protein FASTA files are not accepted.

Protein Name / File Header > DapA (Weinstock et al. 2014, PNAS 111: 11679-84)

MGSSHHHHSSGLVPRGSHMFTGSIVAIVTPMDEKGNVCRASLKKLIDYHVASGTSIAIVSVGTTGESATLNHDEHCDVV
MMTLELADGRIPVIAGTGANATAEAI SLTQRFND SGIVGCLTVTPYYNRPSQEGLYQHFKAI AEHTDLPQILYNVPSRTG
CDLLPETVGR LAKVKNI IGIKEATGNL TRVNQIKELVSDDFVLLSGDDASALDFMQLGGHGVISVTANVAARDMAQMCKL
AAEGHFAEARVINQRLMPLHNKLFVEPNPIPVKWACKELGLVAITD LRLPMT PITDSGRET VRAALKHAGLL

Close (-) <-- 1 / 1 --> Clone (+)

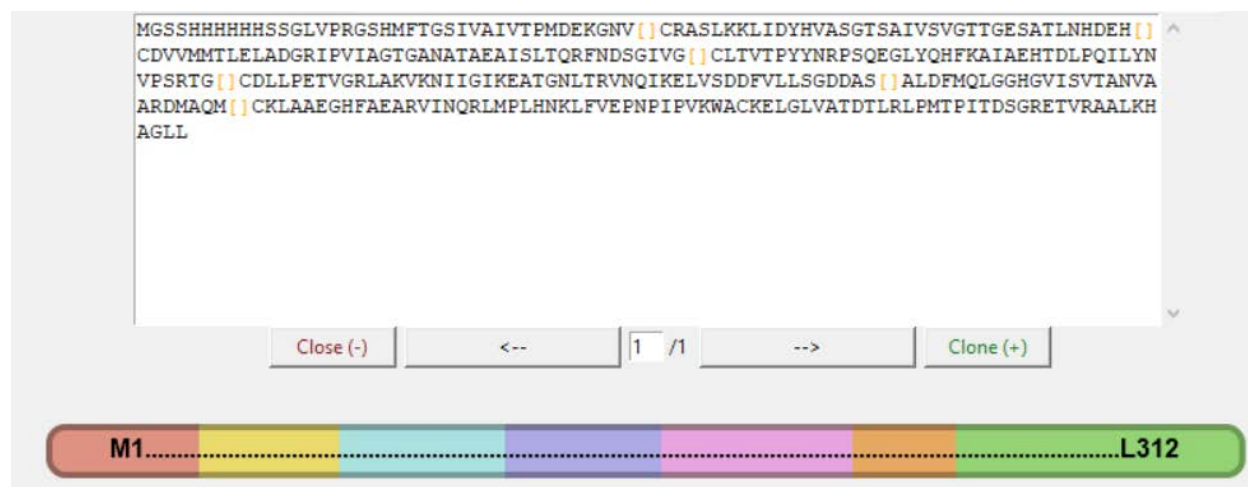
M1.....L312

When text is loaded or typed into the sequence text box, an image preview appears below. This sequence has no annotations yet, so it is interpreted as a single segment.

For any aa that can't be represented with a single-letter code, see Section 2.3 for syntax.

2.2 ADD LIGATION JUNCTIONS

To denote a ligation junction where two segments will undergo a ligation reaction (e.g., native chemical ligation), insert a pair of square brackets “[]” between the segments.

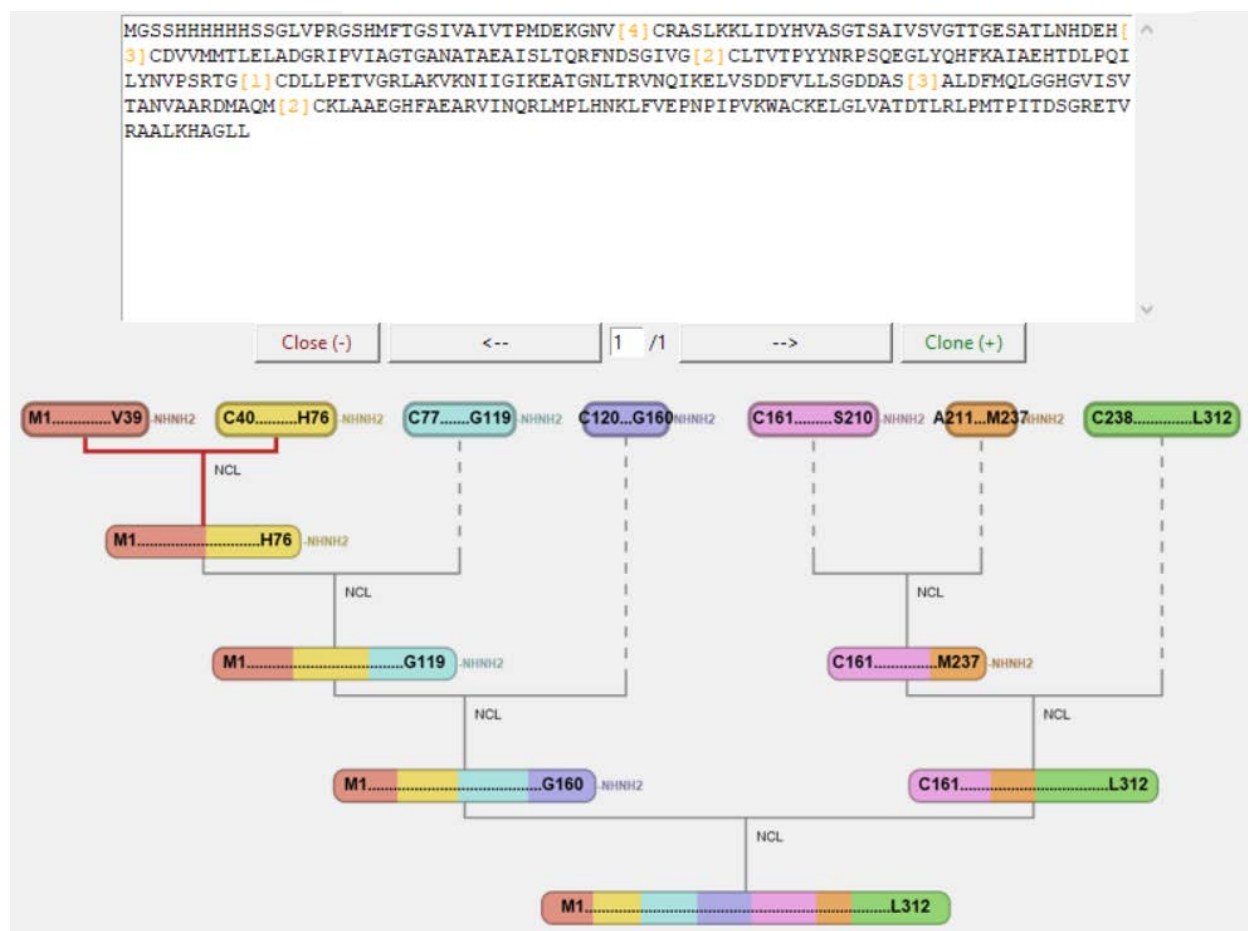


We have inserted 6 ligation junctions to split the protein into 7 segments, which are shown in the preview image as colored sections with accurate relative widths

After inserting ligation junctions, the **Auto-Bracket** tool can be used to complete the bracket with a suggested ligation order. See *Section 6 – Automatic Bracket Assembly* for more information on this feature.

If using Auto-Bracket, the rest of Section 2 may be skipped. The remainder of this section describes how to build a custom bracket manually.

Next, insert numbers into the square brackets to specify the **order of ligation**. As numbers are added, the bracket image preview will change to match. Junctions are numbered from the bottom (final reaction) up, and the final junction is always [1].



The order of the junctions in this bracket is [4] [3] [2] [1] [3] [2]. In CPS, ligations will be performed from the top (highest number) down. When creating a new bracket, it is usually easiest to start from the final reaction [1] and build upwards.

Note that the leftmost NCL reaction is highlighted in red, warning about the use of a poor thioester-aa (Val) at this junction. This program behavior, along with several other image features, can be modified in the Figure Settings menu (see Section 5 – Modifying the Image).

Experiment with different ligation orders to determine the optimal synthetic plan for your protein. At any point in the bracket-building process, use the “Clone (+)” button to make a copy of the sequence and return to it later (see Section 4 – Comparing Multiple Brackets).

2.3 (OPTIONAL) MARK SPECIAL AMINO ACIDS AND SIDE-CHAIN REACTIONS

2.3.1 Non-Canonical AA

Within the sequence, any text surrounded by parentheses represents a single amino acid; use parentheses for non-canonical aa or any other special aa that cannot be represented as a single letter. For example, (Nle) in a sequence is interpreted as a single amino acid with the name “Nle”.

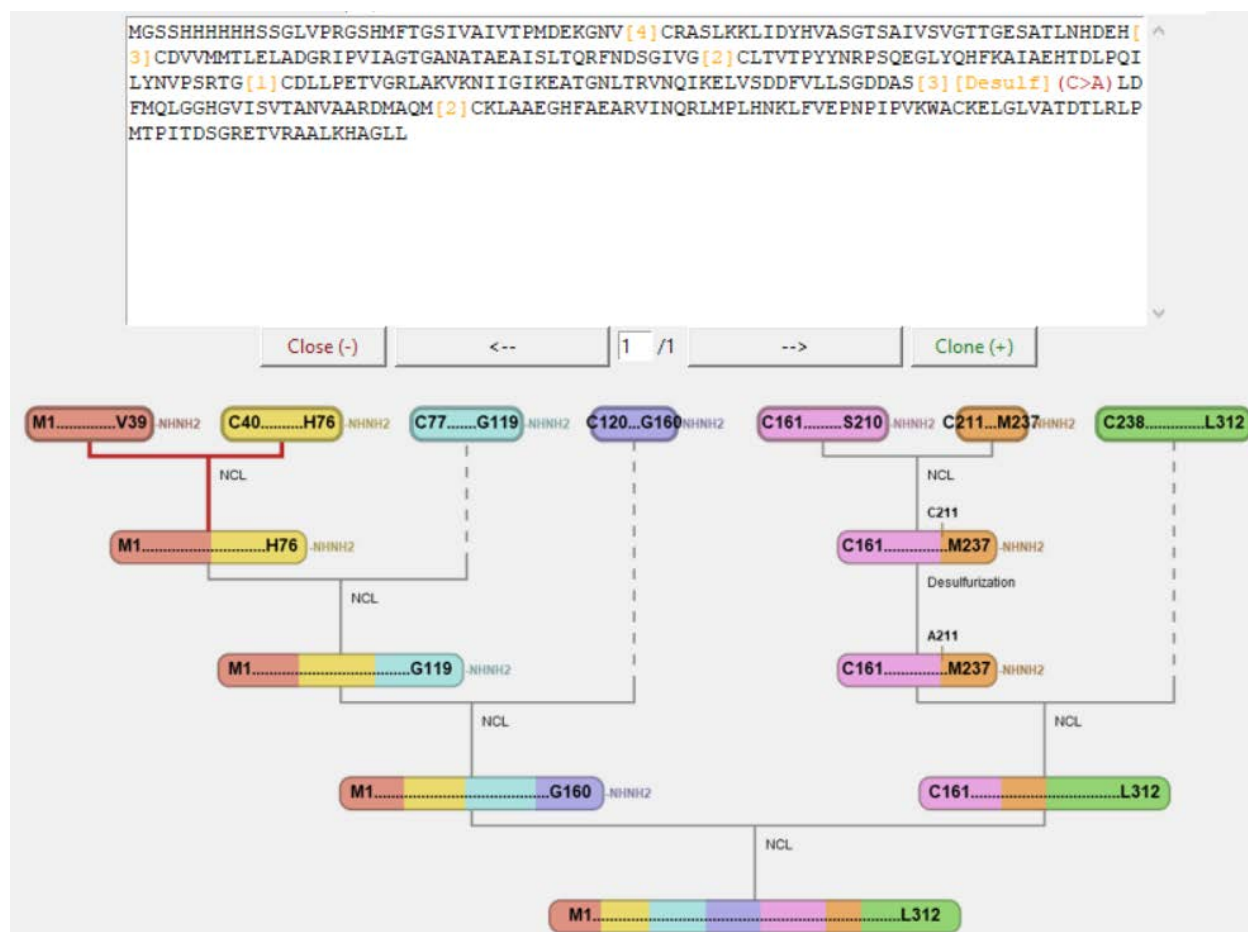
Except for the first and last aa of each segment, special aa will not be visible on the bracket (unless they undergo a side-chain reaction; see below). However, brackets saved in SVG format (see Section 3 – Saving/Exporting a Bracket) will include a color-coded reference sequence above the bracket. In the reference sequence, all special aa are highlighted and assigned a single-character code with an accompanying key.

2.3.2 Side-Chain Reactions (e.g., Desulfurization)

Some amino acids may undergo a side-chain transformation during a synthesis; a common example is the use of desulfurization after NCL to convert one or more Cys to Ala. To denote an amino acid whose identity changes during a synthesis:

- a) Replace the aa in the sequence with the special aa format (**X>Y**), where X is the initial aa in the starting segment, and Y is the final aa in the product protein.
For example, (C>A) indicates a Cys transformed to an Ala, and (Pen>V) indicates a penicillamine transformed to Val.
- b) Add a matching reaction step immediately following a ligation step. A reaction step is indicated by text within square brackets, such as [**X>Y**], placed adjacent to a ligation step.
The keyword [Desulf] is a special reaction step that may be used in place of [C>A] or [Pen>V], as in the example below. BracketMaker interprets [Desulf] as a desulfurization step causing both of these transformations. All other reactions must use the format [X>Y] and must match the aa exactly.

These text annotations are illustrated in the below example:



In our example, A_{211} (in orange segment) must initially be made as Cys to enable NCL, then converted to Ala in the final protein. We have included a [Desulf] step after the ligation number [3] between pink and orange segments, and we have replaced the A at this junction with (C>A). As a result, the bracket now shows a single-segment desulfurization reaction after NCL in which C_{211} is converted to A_{211} .

Note that the modified aa is only labeled in the segments prior to and immediately after transformation. All segments below these are assumed to contain A_{211} , matching the final protein reference sequence included in the output SVG (see Section 3 – Saving/Exporting a Bracket).

To indicate **one-pot reactions** (i.e., sequential reactions without intermediate purification), sequential steps may be included in a single set of square brackets, separated by a comma. For example, [3,Desulf] would indicate the same transformation as in the above example, but the intermediate pink/orange segment with C_{211} would not be shown.

For clarity and accurate assessment of CPS strategies, segments should only be shown on a bracket if they are isolated (i.e., purified) after reaction.

2.3.3 Protected Side Chains and Deprotection Reactions

In many strategies, side chains must be protected to avoid problematic side reactions during NCL and/or desulfurization, and these protecting groups may be removed later in the synthesis to expose the reactive side chain. Other strategies may include the use of temporary solubilizing tags on side-chains, which are removed in the final synthesis. To indicate any transformation involving the removal of a protecting group, tag, or label from an aa side chain:

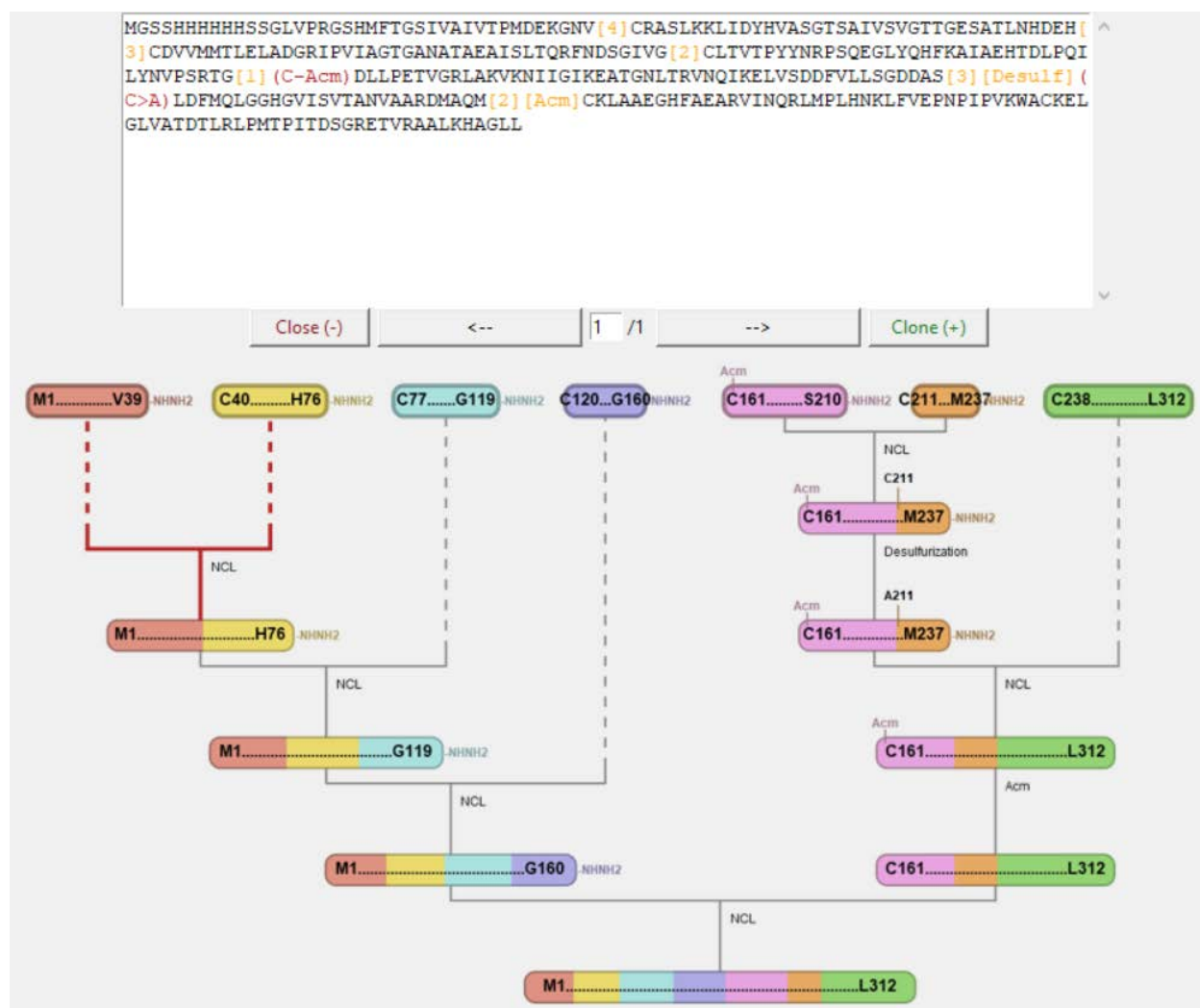
- a) Replace the aa in the sequence with the special aa format **(X-Abc)**, where X is the “true” side chain and Abc is the labile group.

For example, (C-Acm) indicates a Cys protected with the Acm group.

- b) Add a removal step matching the labile group immediately following a ligation step. A removal step is indicated by text within square brackets, such as **[Abc]**, placed adjacent to a ligation step.

Any reaction that does not contain a right-arrow is assumed to be a removal step. For example, [Acm] represents the removal of “-Acm” from any aa with this protecting group.

These text annotations are illustrated in the below example:



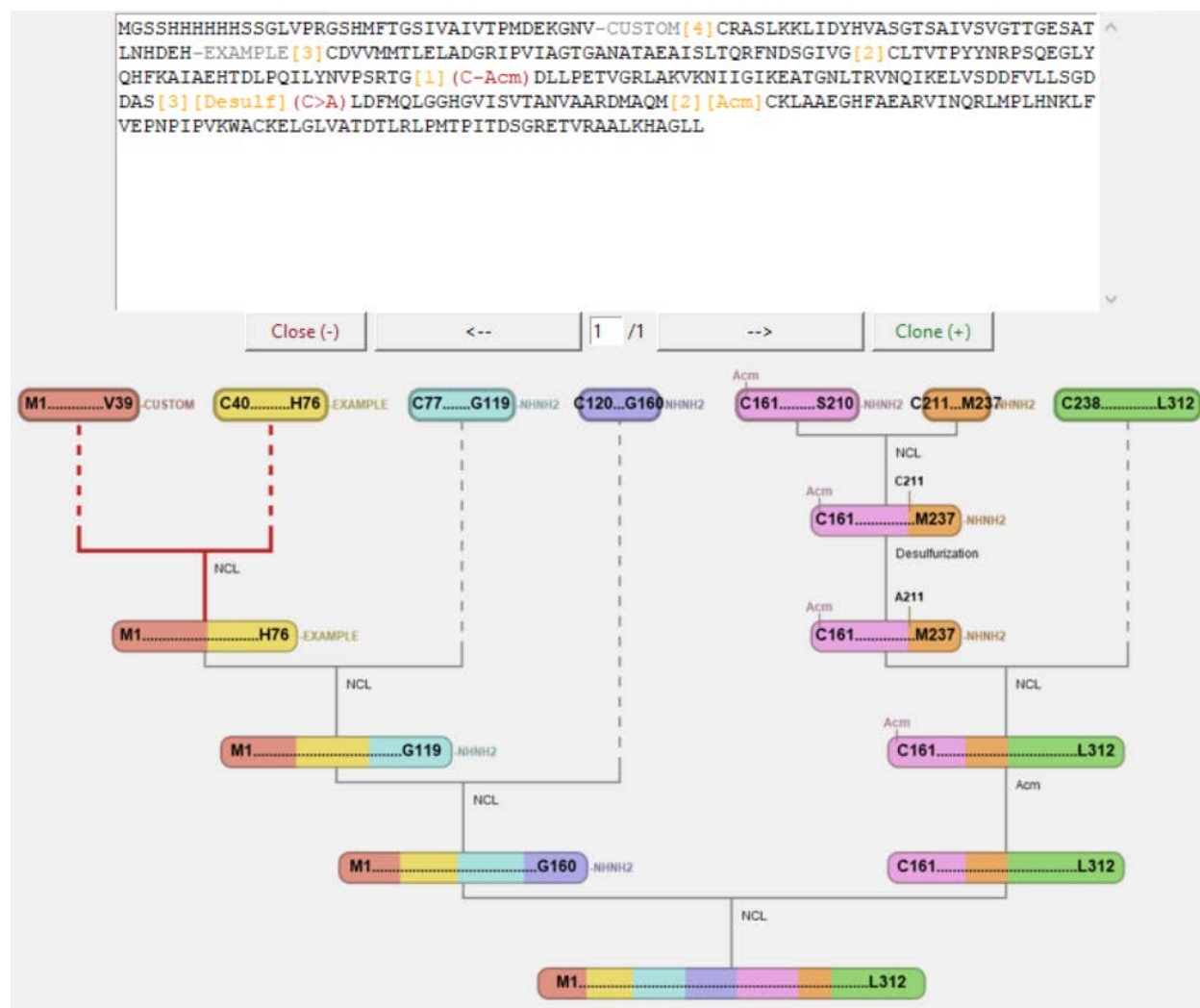
In our synthesis, C_{161} must be protected during desulfurization and deprotected before its ligation. To indicate this transformation, we have replaced the amino acid in the sequence with (C-Acm), and we have included an [Acm] removal step after the ligation number [2] between orange and green segments. The "Acm" label is shown on each relevant segment until it is removed.

See the previous sub-section (Side-Chain Reactions) for the syntax of one-pot reactions.

2.4 (OPTIONAL) ADD CUSTOM THIOESTER LABELS

By default, BracketMaker labels each segment (except the last) with a “-NHNH₂” group at its C-terminus. The default label for C-termini of segments be specified in the Figure Settings menu (see *Section 5 – Modifying the Image*).

In some strategies, segments may not all use the same C-terminal reactive group. Custom labels for each segment may be specified in the sequence text following a hyphen, as in the below example (in grey):



The first two segments use custom C-terminal labels, and all other segments use the default.

3 SAVING/EXPORTING A BRACKET

Brackets can be saved in a variety of formats: plain text (.txt), standard vector graphics (.svg, recommended image output), and portable network graphics (.png, non-editable preview image). When saving, use the check boxes below the “Save As” button to specify the desired format(s).

3.1 SAVE TEXT FOR FUTURE EDITING

Save a bracket in text (.txt) format to reopen it in BracketMaker later. The text file will have two lines, the first containing the “Protein Name / File Header” field (preceded by a right arrow “>”) and the second containing the contents of the sequence text box.

Note that this only specifies the bracket sequence, and contains no information about segment colors or other image settings. We therefore recommend to save figure settings as well (see Section 5.1 – Descriptions of Figure Settings). After reopening a saved bracket from text, you may reopen a saved figure settings profile.

3.2 EXPORT IMAGE

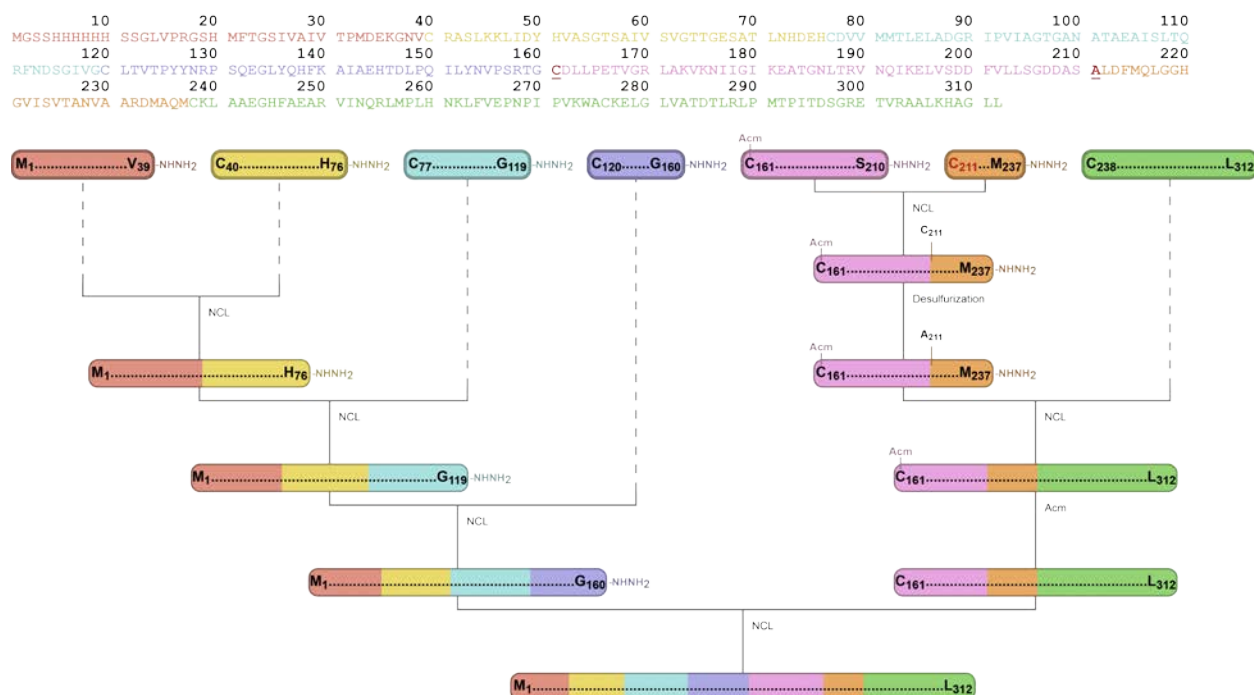
The recommended format for exporting images is standard vector graphics (.svg). This image format is viewable in any web browser and can be fully edited in other vector graphics software (e.g., Adobe Illustrator, Inkscape, Gravit). This format also contains a color-coded reference sequence, not shown in the preview image, as well as text formatting for extra clarity such as subscripted aa numbers and color-highlighted special aas.

SVG files cannot be re-opened in BracketMaker! To edit a bracket later, save as plain text; see above for the note about saving figure settings.

The preview image shown in the BracketMaker GUI may also be saved in .png format; however, this lacks certain image features and cannot be edited later. In most cases, we recommend saving in .svg format, then using a vector graphics program such as Adobe Illustrator to convert this to a .png.

The Python package used by BracketMaker to convert SVG to PNG does not account for text formatting, but most vector graphics software does.

The final DapA bracket built in Section 2.3.3 is shown below:



This image was created using default BracketMaker settings, except a) Highlight Poor Thioesters was disabled, and b) AA Width was set to 4 px to make room for all segment labels. The image was saved in .svg format and converted to .png in outside software.

4 COMPARING MULTIPLE BRACKETS (NAVIGATION PANE)

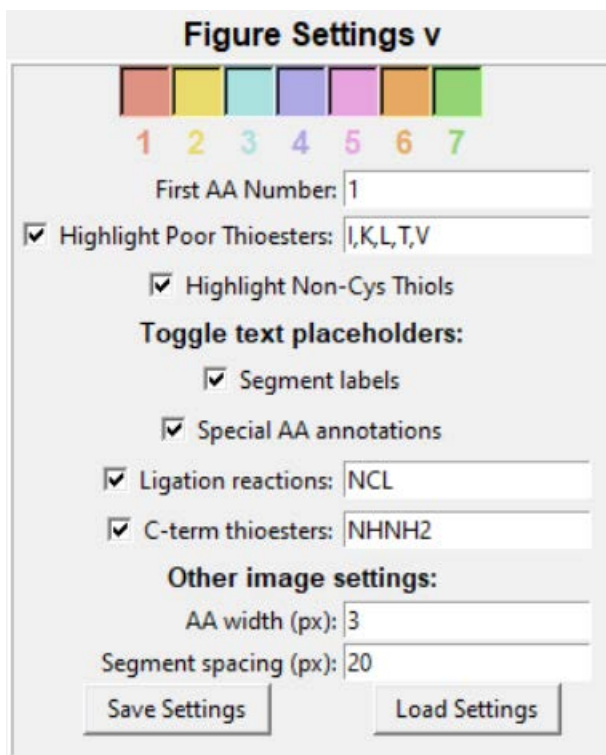
Multiple brackets may be loaded into an active BracketMaker session; the number of open brackets is shown below the text-editing window. The navigation buttons may be used to cycle between open brackets. Only one bracket is displayed at a time, and any brackets not saved (see *Section 3 – Saving/Exporting a Bracket*) will be lost when BracketMaker is closed.

The navigation buttons are described below:

- **Previous (<--)** and **Next (-->)**
 - Switch between loaded brackets. The current position in the “queue” of loaded brackets, as well as the size of the queue, is shown between these buttons.
- **Close (-)**
 - Clear the currently displayed bracket and remove it from the list of loaded files.
Remember to save before closing!
- **Clone (+)**
 - Copy the currently displayed bracket and switch to the new copy. Both copies may be edited and saved separately.
This is useful for exploring multiple synthetic routes, i.e., multiple ways to complete a partially finished bracket.

5 MODIFYING THE IMAGE (“FIGURE SETTINGS” MENU)

Click the “Figure Settings >” button to expand the Figure Settings menu, which contains several user settings that change the look of the output image. Note that default settings are loaded each time the program is run! To save figure settings for a future session, see **Save/Load Settings** below.



The Figure Settings v dialog box contains the following elements:

- Color Swatches:** A row of seven colored squares (red, yellow, cyan, blue, magenta, orange, green) with numbers 1 through 7 below them.
- First AA Number:** A text input field containing the value 1.
- Highlight Poor Thioesters:** A checked checkbox followed by a text input field containing the sequence I,K,L,T,V.
- Highlight Non-Cys Thiols:** A checked checkbox.
- Toggle text placeholders:** A section header followed by two checked checkboxes: Segment labels and Special AA annotations.
- Ligation reactions:** A checked checkbox followed by a text input field containing NCL.
- C-term thioesters:** A checked checkbox followed by a text input field containing NHNH2.
- Other image settings:** A section header followed by two text input fields: AA width (px) containing 3, and Segment spacing (px) containing 20.
- Buttons:** Two buttons at the bottom labeled Save Settings and Load Settings.

Note: The vector graphics output format (.svg) can be fully edited in any vector graphics software such as Adobe Illustrator, Inkscape, or Gravit, so all of these image transformations could be performed after export. However, the figure settings within BracketMaker control repetitive elements that are tedious to edit manually.

When exporting an image, it is recommended to save figure settings as well, in case future edits are desired. To edit a previously saved bracket, load both the bracket text (.txt) and the figure settings (.bmsettings) to recreate the image in BracketMaker. SVG files cannot be re-opened in BracketMaker!

5.1 DESCRIPTION OF FIGURE SETTINGS

Settings apply to the active BracketMaker session, and affect all open brackets. Each setting is described below:

- **Color Swatches:** Square buttons representing colors of segments. These will auto-populate to match the number of segments detected in a bracket. The default setting is a muted rainbow

color scheme with 8 different colors. Click any swatch button to bring up a color-picker window and set the color of the relevant segment.

- *Note: In Mac version, swatch buttons are all white (this is a known issue with the tkinter package). For this reason, a number with the same color is displayed beneath each swatch button.*
 - *This is one of the most tedious settings to change afterward in other vector graphics software, as it involves editing every color gradient in intermediate segments! Remember to save figure settings (see below) if you wish to return to a bracket later.*
- **First AA Number:** The number of the first amino acid in the protein sequence (default = 1). This determines the numbers displayed on all segments, as well as the numbering of the color-coded sequence in the output SVG.
- **Highlight Poor Thioesters:** If checked (default = enabled), NCL reactions involving sub-optimal thioesters with poor reactivity will be marked with a red reaction line. The list of poor thioesters can be modified in this text box (default = I,L,K,T,V); they do not need to be separated with a comma, but are written this way for clarity. This feature is useful for comparing multiple brackets, as these reactions usually have poor yield and should come earlier in a synthesis; see our BracketMaker publication for more information on this subject. When exporting images for publication, it is recommended to disable this setting.
 - *Note: The AA in this text box will also be counted as poor thioesters for bracket scoring and the Auto-Bracket building algorithm (see Section 6 – Automatic Bracket Assembly).*
- **Highlight Non-Cys Thiols:** If checked (default = enabled), NCL reactions involving thiols other than Cys or Ala will be marked with a blue reaction line. This feature is useful for comparing multiple brackets, as these reactions usually have poor yield and should come earlier in a synthesis; see our BracketMaker publication for more information on this subject. When exporting images for publication, it is recommended to disable this setting.
- **Segment Labels:** If checked (default = enabled), segment labels including identity and number of each segment's first and last AA will be displayed on the output image. Disable this if custom segment labels (e.g., Segment A, Segment B, etc.) are desired instead.
- **Special AA Annotations:** If checked (default = enabled), any AA that are modified during a synthesis (see *Section 2.3 – Label Special Amino Acids*) will be marked above each segment with a line and a text label at the position of the AA. Disable this if custom labels (e.g., shapes or chemical structures) are desired instead.
- **Ligation Reactions:** If checked (default = enabled), a placeholder text label will be included next to each reaction line to indicate the type of reaction. The label used for ligation reactions may be modified in this text box (default = "NCL"); other reactions are labeled as they appear in the annotated protein sequence (see *Section 2.4 – Add Side-Chain Reaction Steps*).
 - *Note: For publication, this text is typically a description of the reaction conditions; thus, this label is only included as a placeholder, and should be edited in outside software prior*

to publication.

- **C-Term Thioesters:** If checked (default = enabled), all segments except the last will be labeled with –X at their C-terminus, where X is the text in this box (default = NHNH₂). Numbers will be automatically subscripted in the output SVG. This represents the type of thioester chemistry used in the synthesis, and should be disabled if another representation (e.g., a chemical structure) is desired. By convention, this label depicts the C-terminus of segments as they appear at the start of each reaction.
- **AA Width (px):** The width of each segment is determined by the number of AA in that segment (default = 3 pixels per AA). Change this to a larger value if some segments are too small to fit their label.
- **Segment Spacing (px):** This determines the white space included as “padding” between segments on the same horizontal level (default = 20 pixels). Change this to a larger value to create more white space for your own segment C-terminal and N-terminal labels (e.g., chemical structures).
- **“Save Settings” button:** Save the current settings profile to a plain-text file (.bmsettings file extension). This file contains the values for all of the settings described in this section. This is highly recommended whenever you export an image, as .svg/.png files cannot be re-opened by BracketMaker, and .txt files do not contain information about the image.
- **“Load Settings” button:** Load a settings profile from a previously-saved .bmsettings file.

6 AUTOMATIC BRACKET ASSEMBLY (“AUTO-BRACKET”)

Click the “Auto-Bracket >” button to expand the Auto-Bracket menu.

Auto-Bracket v

Current Bracket Info:

Max Path	Avg Path	Rxn Steps
5	4.00	8

To auto-fill, insert [] between segments and click Build Bracket, or import an Aligator file.

☒ Save AutoBracket Results (.tsv)

How many different possible brackets? (0 for all)

AutoBracket Settings

☒ Penalize Poor Thioesters (+1 Path)

☒ Penalize Non-Cys Thiols (+2 Path)

Cys Protecting Group:

N-terminal Cys Protecting Group (if different):

Toggle one-pot steps: ☐ Desulfurization

☐ Internal Cys Deprotection

☒ N-Terminal Cys Deprotection

The Auto-Bracket tool automates the steps described in *Section 2 – Creating a Bracket*, enabling users to easily build brackets with standard NCL techniques. Auto-Bracket can also be used to compare multiple possible synthetic routes and find the most efficient order of joining segments together.

For a more thorough description of the bracket-building process and our definition of “standard NCL”, see our BracketMaker publication. Briefly, the program performs a retrosynthetic analysis of a protein using every possible ligation order, ranks each order by their bracket scores, and displays the best (i.e., lowest-scoring) bracket representing the most efficient predicted route.

To use Auto-Bracket, insert empty ligation junctions [] between segments (as illustrated in *Section 2.2 – Add Ligation Junctions*), modify settings as desired, and click the **Build Bracket(s)...** button. This will assemble the input segment list in every possible order and display the most efficient predicted order(s).

To explore multiple variations of possible segments as well as multiple orders, see Section 6.2 – Combining Aligator and Auto-Bracket.

The “**Current Bracket Info**” table displays scores about the current bracket that can be used to compare multiple brackets. For more information about these scores, see our BracketMaker publication.

- **Max Path:** The longest “path length” (i.e., number of sequential yield-loss steps) of any segment. Most reactions count as 1 path, but some poor-yielding reactions may count as 2 or 3 paths (see description of penalties below). This usually should be kept as small as possible to obtain the best protein yield.
- **Avg Path:** The average path length of all segments, including path penalties. This can help determine which bracket is preferred if multiple brackets have the same Max Path.
- **Rxn Steps:** The total number of individual reactions involved in the synthesis. This gives an idea of the complexity of the CPS project but is not necessarily relevant for yield.

Below the info table are the main buttons for running Auto-Bracket:

- **“Build Bracket(s)” button:** This button runs the Auto-Bracket function with the current user settings.
 - *Depending on the number of segments, this may take several seconds or minutes to run. When finished, resulting brackets will be automatically loaded into the active BracketMaker session.*
- **“Aligator Import” button:** This button is used to import results from our Automated Ligator (Aligator) program, performing Auto-Bracket with the current user settings on each line of an Aligator results file (see *Section 6.2 – Combining Aligator and Auto-Bracket* for a description of this feature).

6.1 DESCRIPTION OF AUTO-BRACKET SETTINGS

The rest of the Auto-Bracket menu contains user settings for the bracket-building function:

- **Save AutoBracket Results (.tsv):** If checked (default = enabled), the resulting list of possible brackets will be written to a tab-separated values (.tsv) file.
 - *This can be opened in a future BracketMaker session using the **Open From List** button in the main menu.*
- **How many different possible brackets?:** This determines the number of results that Auto-Bracket will display after it is finished, and the number that it will write to the .tsv results file (default = 1).
 - *Increase this number to see multiple possible orders of assembling the same set of segments, or set to 0 to see every possible order.*
 - *Ligation orders are ranked by their bracket scores (lowest Max Path, then lowest Avg Path, then lowest Rxn Steps), which generally results in convergent brackets being shown first.*

- When using the “Alligator Import” feature, this setting determines the number of possible orders per Alligator line.
- **Penalize Poor Thioesters (+1 Path):** When checked (default = enabled), this applies a penalty of +1 path length (total path length = 2) to reactions involving a thioester with poor reactivity, as these reactions are generally low-yielding.
 - The list of Poor Thioesters may be modified in the Figure Settings menu (see Section 5.1 – Description of Figure Settings). If enabled, Auto-Bracket results will be biased towards brackets in which these reactions occur earlier.
 - This setting will also affect the currently displayed bracket scores.
- **Penalize Non-Cys Ligations (+2 Path):** When checked (default = enabled), this applies a penalty of +2 path length (total path length = 3) to reactions involving a thiol site other than Cys (C) or Ala (A), as alternative thiolated AAs often have very poor reactivity and are low-yielding.
 - If enabled, Auto-Bracket results will be biased towards brackets in which these reactions occur earlier.
 - This setting will also affect the currently displayed bracket scores.
- **Cys Protecting Group:** When an A or V thiol site is used in NCL, Auto-Bracket will automatically add a desulfurization [Desulf] step after NCL; during this desulfurization, any unprotected Cys in the combined segment must be protected. The Cys protecting group may be specified in this text box (default = AcM).
 - With the default protecting group of “AcM”, any “C” in relevant segments will be replaced with “(C-AcM)”, and an “[AcM]” removal step will be added after NCL and desulfurization.
- **N-Terminal Cys Protecting Group:** In many NCL reactions, the left-half segment may contain an N-terminal Cys that is vulnerable to self-reaction (i.e., cyclization of this Cys with the segment’s activated thioester). This N-terminal Cys must be protected during NCL and deprotected afterward. The protecting group may be specified in this text box (default = Tfa-Thz).
 - With the default protecting group of “Tfa-Thz”, any N-terminal “C” in relevant segments will be replaced with “(C-Tfa-Thz)”, and a “[Tfa-Thz]” removal step will be added after NCL.
 - This can be the same as the Cys Protecting Group above, but it is helpful to use different protecting groups if N-terminal Cys deprotection can be performed one-pot with NCL (e.g., using the Tfa-Thz group) while deprotection of internal Cys residues cannot.
- **Toggle One-Pot Steps:** These checkboxes specify which steps are performed one-pot after NCL (i.e., without purifying the NCL product) and which must be performed in a separate reaction after purification. By default, “N-Terminal Cys Deprotection” is the only enabled one-pot step, meant to be paired with the default “Tfa-Thz” protecting group.
 - When assembling brackets, Auto-Bracket will add any multi-pot steps in separate reaction tags – for example, [2][Desulf][AcM] – and will include any one-pot steps within the same tag as the ligation number – for example, [2,Tfa-Thz].

- *One-pot steps are “invisible” on the bracket, and only the final product after all one-pot steps will be shown. One-pot steps do not add anything to Max Path, Avg Path, or Rxn Steps bracket scores.*

6.2 COMBINING ALIGATOR AND AUTO-BRACKET

Auto-Bracket may be used in conjunction with our Automated Ligator (Aligator) program to explore many possible routes of synthesizing a protein. We highly recommend using this method to explore possible synthetic routes for a new protein. The Aligator program and its user manual may be downloaded at the following URL: www.github.com/Kay-Lab/Aligator

For a full description of Aligator, see [Jacobsen, Erickson, & Kay 2017, Bioorg Med Chem 25: 4946-4952]. Briefly, Aligator processes a protein to give a list of possible segment sets divided at NCL junctions, with scores describing segment length, solubility, and the use of poor thioesters and desulfurization steps. BracketMaker can then be used to assemble each segment set into a bracket following standard rules of NCL.

6.2.1 Use Aligator to Explore Ligation Junctions

Run Aligator on your target protein following the Aligator user manual. Among Aligator’s file outputs is a text file named “(Protein Name) All Strategies.txt” containing the most highly-ranked strategies for the protein. Keep this text file to pass to BracketMaker.

6.2.2 Match Auto-Bracket Settings with Aligator Settings

In a new BracketMaker session, modify Auto-Bracket settings as desired (see *Section 6.1 – Description of Auto-Bracket Settings*). Pay special attention to the following settings to match those used in your Aligator run:

- **Poor Thioester Penalty**
 - Modify the list of Poor Thioesters within Figure Settings (see *Section 5.1 – Description of Figure Settings*) to include all “accepted” thioesters from your Aligator run. Do not include “preferred” thioesters in the Poor Thioesters list. Any thioesters marked as “forbidden” in Aligator are irrelevant, as they will not appear in any strategy.
 - In Auto-Bracket settings, check the box to penalize poor thioesters with a +1 path penalty.
- **Non-Cys Thiol Penalty**
 - If any thiols beyond C and A were allowed in your Aligator run, check the box in Auto-Bracket settings to penalize non-Cys ligation sites with a +2 path penalty.
 - Note that the current version of BracketMaker will only add desulfurization steps for A and V thiols (as C and Pen respectively); additional desulfurization pairs may be added in a future update.
- **Number of Brackets**
 - To avoid being inundated with data, we recommend setting the “How many different possible brackets?” field to 1. This will provide 1 bracket for each Aligator line.

6.2.3 Import the Aligator Results File into BracketMaker

Under the Auto-Bracket menu, click the **Aligator Import** button to find and open the “(Protein Name) All Strategies.txt” results file. Auto-Bracket will automatically run on each line of the file, and progress will be displayed in the terminal window. Depending on the size of the protein, this can take several minutes to complete.

When finished, the resulting brackets will be loaded in the active BracketMaker session. Their order corresponds to their original order from Aligator (i.e., their “Aligator Rank”), not necessarily to their order when ranked by bracket scores. To find the most efficient predicted brackets in the set, open the .tsv file of results (in a separate program such as Excel) and sort by Max Path, then by Avg Path.

Keep in mind that brackets with fewer segments will often have lower scores, but the segments may be large and difficult to synthesize. When interpreting bracket difficulty, consider the feasibility of individual segments as well as the bracket scores!

6.3 SAVING AND LOADING AUTO-BRACKET RESULTS

When running Auto-Bracket, a user may optionally save the list of results to a tab-separated values (.tsv) file. This file may be opened in BracketMaker later using the **Open From List** button in the main menu. Brackets are loaded in the order in which they are listed; keep this in mind when editing the .tsv file (e.g., sorting results by bracket scores).

Thank you for using BracketMaker! For any questions, comments, or suggestions for improvement, please reach out to the Kay lab at the University of Utah:

<https://kay.biochem.utah.edu/>

