

SimpleSynteny Command Line Manual Vr. 1.3.1

By: Daniel Veltri (dan.veltri@gmail.com)

December 11, 2017

Contents

	Page
1 Introduction	3
2 Required Dependencies	3
3 Script Setup for Your System	4
4 Running SyntenyMapper to Generate CMAP Files	5
4.1 Running the Script	5
4.2 File Output	5
5 Running SyntenyDrawer to Generate a Figure	6
5.1 Naming and Ordering Genomes	6
5.2 Running the Script	6
5.3 Changing the DPI (Dots Per Inch) of a Figure	7
5.4 Customizing Gene Target Colors	7
6 The CMAP File Format	8
6.1 Mapping a Single Contig	8
6.2 Query Names and Direction	8
6.3 Query Numbering	9
6.4 Contig Numbering	9
6.5 Valid Numbers	9
6.6 Removing Gene Directions	9
6.7 Customizing CMAP Files	10
7 Acknowledgements	10
8 Recent Changes	10

1 Introduction

SimpleSynteny provides two Ruby scripts to help you generate figures on your own computer. The first script **SyntenyMapper.rb** takes a separate gene target and genome FASTA file as input and produces “Contig Mapping” (CMAP) lines for all hits as output. After removing any unwanted lines (which represent contigs, supercontigs, scaffolds, etc.) from a CMAP file, this is fed as input to the **SyntenyDrawer.rb** script to generate a final figure. Details on the CMAP file format are provided in section 6 at the end of this document.

If you have any questions, suggestions or detect any errors with the **SimpleSynteny** scripts (or server) please contact Dan Veltri at: dan.veltri@gmail.com. **SimpleSynteny** is freely released under a GNU GENERAL PUBLIC LICENSE v3. Please see the LICENSE.txt file that should have been distributed with the scripts.

2 Required Dependencies

SimpleSynteny depends on the following programs and libraries to work († = required for **SyntenyMapper**, ‡ = required for **SyntenyDrawer**):

- Ruby ($\geq 1.8.7+$)^{†‡}. Freely available at: <https://www.ruby-lang.org>
 - BioRuby Gem[†]. Freely available at: <http://bioruby.github.io>
 - RMagick Gem[‡]. Freely available at: <https://rubygems.org/gems/rmagick>
- ImageMagick ($\geq 6.6.9$)[‡]. Freely available at: <http://imagemagick.org> (Note: RMagick may encounter issues with ImageMagick vr.7+)
- NCBI-BLAST Binaries[†]. Freely available at: <https://blast.ncbi.nlm.nih.gov>
 - * BioRuby requires **blastall** from the legacy BLAST release. You can either use **blastn** and **tblastn** from the legacy v2.2.9 binaries *or* have the script point to the newer BLAST+ programs. Be aware that some default **blastall** options may differ from the BLAST+ standalone programs (e.g. **wordsize**). The **legacy_blast.pl** script that comes with BLAST+ may be helpful to check your parameters if this is a concern.

For dependency installation, while we show some example installation procedures below, *we cannot provide support outside of the SimpleSynteny scripts. Please direct dependency questions to the relevant project.*

Example Install with Ubuntu 14.04+ (with root access)

1. **sudo apt-get install imagemagick libmagickcore-dev libmagickwand-dev**
Check that ImageMagick installed by typing in the command line ‘**convert -version**’ to ensure a version number prints out.
2. **sudo apt-get install ruby**
Check that both Ruby and Gems installed by typing in the command line: ‘**ruby -v**’ and ‘**gem -v**’ to ensure version numbers print out.
3. **gem install rmagick** (may require sudo access)
4. **gem install bio** (may require sudo access)
5. Download NCBI-BLAST+ and install locally.
6. Download **blastall** from the legacy BLAST release and save with other BLAST executables.

Example Install on Mac OS X with Homebrew

Mac OS X typically comes with Ruby already installed. We recommend using Homebrew to make installation easier.

1. `brew update && brew upgrade` (update homebrew and repositories)
2. `brew install imagemagick`
Check that ImageMagick installed by typing in the command line '`convert -version`' to ensure a version number prints out.
3. `gem install rmagick` (may require root access)
4. `gem install bio` (may require root access)
5. Download NCBI-BLAST+ and install locally.
6. Download the Mac OS version of `blastall` from the legacy BLAST release and save with other BLAST executables.

Example Install on Windows XP/7/8/10

Note, installing ImageMagick and RMagick for Windows can be a frustrating process. If a native Windows install does not work, it may be easier to install through Cygwin.

1. Install an appropriate (32 vs 64bit) 'RubyInstaller' and 'Development Kit' (DevKit) package for your system from:

<http://rubyinstaller.org/downloads>.

Check that both Ruby and Gems installed by typing in a command prompt window '`ruby -v`' and '`gem -v`' to ensure version numbers print out.

2. Follow the instructions for installing ImageMagick and RMagick with an appropriate Windows binary release for your system according to the instructions here:

<https://github.com/rmagick/rmagick/wiki/Installing-on-Windows>

Check that ImageMagick installed by typing in a command prompt window '`convert -version`' to ensure a version number prints out.

3. `gem install bio` (from command prompt window)
4. Download NCBI-BLAST+ and install locally.
5. Download `blastall` from the legacy BLAST release and save with other BLAST executables.

3 Script Setup for Your System

The `SyntenyMapper` script requires the user to adjust a few variables near the top of the file using a text editor. `$LOCAL_BLASTALL_LOC` and `$LOCAL_MAKEBLASTDB_LOC` should be set to the full path to where the `blastall` and `makeblastdb` programs are installed, respectively. `$BLAST_VR` should be set to the version of BLAST you are using so that printed reports match.

```
$LOCAL_BLASTALL_LOC = "../my_blastall_install_location"
$LOCAL_MAKEBLASTDB_LOC = "../my_makeblastdb_install_location"
$BLAST_VR = "BLAST_VrX.X.X"
```

BLAST database files will be created in the current working directory- by default they are deleted after the program runs but they can be kept by setting `$CLEAN_DB` to `false`. The `SyntenyDrawer` script should not require any changes if ImageMagick and RMagick were installed correctly and CMAP files are properly formatted.

4 Running SyntenyMapper to Generate CMAP Files

The **SyntenyMapper** script requires two FASTA files as input: one containing your genome of interest, and another containing your gene target(s) of interest (as either DNA *or* protein sequences- but not both). **Please see the paper for important details on ‘Discovery versus visualization’ (Materials and Methods) when choosing an appropriate gene target file for your project.** The output of the program will be properly formatted CMAP lines (see Section 6 for format details), with each line corresponding to a contig/scaffold with a significant hit found by BLAST.

4.1 Running the Script

SyntenyMapper is run from the command line as follows:

```
ruby SyntenyMapper.rb targets.fasta genome.fasta output_base_name [options]
```

Here, ‘targets.fasta’ corresponds to the FASTA file with your query targets, and ‘genome.fasta’ is the FASTA file with your genome. ‘output_base_name’ is the base name of the CMAP file and other reports you create (see Output below for details). The following option flags can be used with the script, which, you can review by calling the ‘-help’ flag with the script on the command line:

- -e <num> : Adjust the E-value cutoff for BLAST (default: 0.001)
- -n <num> : Number of CPU cores on your system for BLAST to use (default: 1)
- -gap <T/F> : Have BLAST perform gapped alignment? True/False (default: T)
- -cov <num> : Adjust the Minimum Query Coverage / Identity % (default: 30)
(Use 0 to disable - see the **SimpleSynteny** FAQ/Help Page for details.)
- -f : ‘Flip Contigs’ displays forward and flipped/reversed CMAP line for each hit (default: off)
- -gene <string> : Only print results for gene name provided (default: off)
- -debug : Print extra information other than CMAP lines (default: off)
Note remove debug output from generated file before calling **SyntenyDrawer** if used!
- -help : Show help dialog

IMPORTANT: Before running the script, you need to adjust the lines (detailed above in Section 3) so the program can find **blastall** and **makeblastdb** on your system.

4.2 File Output

By default, files will be output to the current working directory, or a path prefix given with the “output_base_name” argument.

CMAP files are generated using the base name plus a “_fwd.cmap” suffix to denote a forward direction. If the **-f** flag is called, an additional file ending in “_rev.cmap” will be generated to denote the reverse or “flipped” version. You should look at both files and choose one to include in a working project directory for the **SyntenyDrawer** script. Once you have placed the file in that directory, remove the “_fwd” or “_rev” from the filename so that it does not appear as part of the genome name in the figure. Typically, hits for common genes like aldolase may appear on many contigs throughout a genome so you will likely want to review your CMAP file in a text editor and delete any unwanted lines to make figures more readable.

In addition to CMAP files, the script will generate base name files ending as “_skipped.genes.txt” (containing a list of any gene targets with no significant hits found), “_MASKS.txt” (showing query hits tiled onto target sequences when calculating the Minimum Query Coverage option), and “_BLAST.REPORT.csv” (containing a CSV file of your BLAST results).

5 Running SyntenyDrawer to Generate a Figure

The **SyntenyDrawer** script takes a folder location containing one or more valid CMAP files (with .cmap extensions) as input which are processed in alphabetical order and drawn top-to-bottom. We highly recommend creating a separate folder to contain the CMAP files for a particular project. Note that **SyntenyDrawer** does not do any strict checking for proper formatting- if you manually edit files generated by **SyntenyMapper** and run into problems you may want to cut and paste your CMAP lines into **SimpleSynteny**'s Advanced Mode Page to check for errors.

5.1 Naming and Ordering Genomes

To customize the order in which genomes are drawn, you should append numbers to the front of your filenames (01,02,etc.). Filenames are used to label/name your genomes but numbers at the front (and the .cmap extension) of a file will be ignored. Underscore characters will be converted to spaces and the text will be italicized. Currently, gene targets may not contain spaces and see Section 6 for CMAP format details.

5.2 Running the Script

SyntenyDrawer is run from the command line as follows:

```
ruby SyntenyDrawer.rb path_to_cmap_directory [options]
```

Here, 'path_to_cmap_directory' is the path to the folder where you have placed your valid CMAP files generated with **SyntenyMapper** (or typed manually). By default, the script will create 'SimpSynFig.png' as output in the same location as the CMAPs. The following option flags can be used with the script, which, you can review by call the '-help' flag with the script on the command line:

- -o | -out <string> : Name of figure including extension (default: SimpSynFig.png)
Available extensions: .png .jpg .gif .eps .tiff and .pdf
- -width <num> : Width of image in pixels (default: 2600)
- -height <num> : Height of image in pixels (default: 2000)
- -dpi | -d <num> : Set image DPI- see next section for more details! (default: 600)
- -genome-font <num> : Font size of genome name (default: 28)
- -genome-space <num> : Adjust buffer between genome name and first contig as percent of width. Positive numbers will slide contigs right while a negative number will slide them left. (default=1.0)
- -gene-font <num> : Font size of gene names (default: 14)
- -contig-font <num> : Font size of contig names and BP numbers (default: 20)
- -contig-height | -height <num> : Height of contig in pixels (default: 50)
- -optimize | -opt <no/dist/flip> : Optimize genome order relative to first genome? (default: no) Here 'dist' tries to minimize Euclidean distance of lines between genes, 'flip' tries to minimize the number of flip-arrows. Note, this uses a greedy algorithm which will slow down quickly as more genomes are added.
- -shade <blast/both/full> : Shade gene boxes? (default: blast)
See the **SimpleSynteny** FAQ/Help Page for details.
- -lines <straight/curved> : Style of lines connecting genes (default: curved)
- -arrows <num> : Distance to move overlapping arrows in pixels (default=30)

- `-alt` | `-alt-labels <true/false>` : Alternate gene labels top/bottom on the same contig? (default: true)
- `-box` | `-b <true/false>` : Draw a white box around gene names? (default: true)
- `-no-gname` | `-n <true/false>` : Ignore displaying text for gene names? (default='false', often used with `-box` set to false)
- `-grayscale` | `-g <true/false>` : Make figure in grayscale? (default: false)
- `-colors <dir/color_list.txt>` : Full path to optional `color_list.txt` file to assign gene colors (default: random assignment)
- `-debug` : Print debug output as figure is generated to terminal
- `-help` : Show help dialog

5.3 Changing the DPI (Dots Per Inch) of a Figure

On some systems with some versions of RMagick, the `-DPI` flag mentioned above for the script appears to only change the DPI listed in the image information data, but not the actual image DPI. Accordingly, if this is the case we recommend running the `convert` program which comes with ImageMagick on your figure to change the actual DPI of your image. For example, to change the DPI of an image to 100 use the following command:

```
convert my_original.png -density 100 my_new100dpi_image.png
```

Above, 100 can be replaced with the desired DPI in pixels per inch. To check the actual DPI of an image use ImageMagick's `identify` program:

```
identify -format "%x x %y" my_new100dpi_image.png
```

Which should return "100 PixelsPerInch x 100 PixelsPerInch" as output. The `convert` program has a number of additional features such as trimming, centering, adding a border to your figure, etc. See the program documentation for more details.

5.4 Customizing Gene Target Colors

While normally colors are randomly assigned to genes each time the program is run, they can (optionally) be customized if the script is called with the `-colors` flag and a path to a file named `color_list.txt`.

Inside `color_list.txt`, each gene name should be listed on it's own line, just as it is in your CMAP files, followed by two colons and a color name. Color names should either be in hex format or use the names recognized by the ImageMagick library. For a full list of available colors and names, please see the ImageMagick Colors Page: http://www.imagemagick.org/script/color.php#color_names. An example with three genes is as follows:

```
GENE1::CornflowerBlue
GENE2::green
GENE3::MediumOrchid1
```

IMPORTANT: Note `color_list.txt` must contain *all* unique gene names from *all* CMAP files used to generate a figure; otherwise, an error may occur.

6 The CMAP File Format

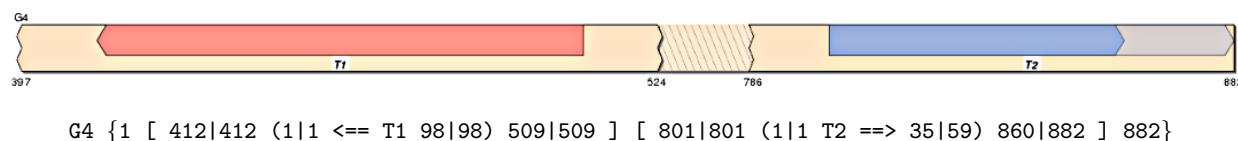
SyntenMapper uses BLAST to map your gene targets onto a genome. It stores these coordinates in a human-readable format called a ‘CMAP’ (Contig-Mapping) file. All of the CMAP files in the folder location provided to **SyntenDrawer** are used to generate a final figure. Each CMAP file represents a single genome, drawn as a single row of one or more contigs in a figure. **SyntenDrawer** processes each CMAP in alphabetical order, drawing genomes in the image from top-to-bottom. To specify a specific order for your genomes, rename the CMAP files with a number at the front (e.g. 01, 02, etc.) which will be ignored when printing the genome name. Here’s how the file format works:

6.1 Mapping a Single Contig

Each line of a CMAP file represents a single contig/supercontig which is drawn from left-to-right. The format of a basic contig line starts with the name of the contig, followed by a space, and then data contained within curly-braces as follows:

```
MyContigName {FirstBasePair# ...LastBasePair#}
```

Note there is one space after ‘MyContigName’ above. The numbers just inside the { } curly-braces represent the first (typically 1) and last base pair (BP) numbers of the contig, respectively. Between these numbers reside one or more query genes or proteins to map onto the contig. Here’s an example figure and the contig line used to make it:

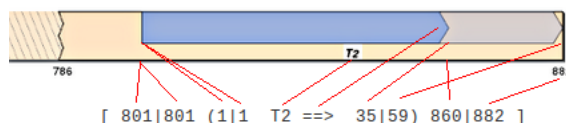


The contig is named ‘G4’, it is 882 BP long and contains two genes ‘T1’ (facing upstream) and ‘T2’ (facing downstream).

We can see the contig (shown in light beige) is named in the top-left corner as ‘G4’ and contains two query protein sequences (DNA sequences could also have been used). The CMAP line denotes the contig stretches from BP’s 1 – 882, however, to optimize space the program removed long stretches of the contig in the figure (BP’s 1 – 398 and 525 – 785) since they did not contain any query sequence information. Anytime the program removes a section of a contig to save space, the contig is drawn with a jagged edge and corresponding BP numbers are indicated beneath the contig. If the end of a contig is reached, a straight edge is shown as seen above BP 882. The query protein ‘T1’ is shown in red and faces upstream, while protein ‘T2’ is shown in blue and faces downstream. Every query sequence marked on a contig line is placed in square brackets [] and separated by two spaces. For a given query, the information inside the brackets is used to decide where on the contig to draw a box to represent it, if and where it should be shaded, what direction it faces (5’-to-3’ vs 3’-to-5’), and what it is named. Using ‘T2’ from the example above, let’s work our way from the query name outwards to see what everything means.

6.2 Query Names and Direction

Query names are automatically capitalized and italicized by the program for figures even if they are spelled in lowercase on a CMAP line. The spelling of names is important, as the program looks for matching identical names between genomes and, if it finds them, draws a connecting line between them. To the left or right of the gene name will respectively be a <== or ==> direction arrow (separated from the name by one space). These are used to draw the gene direction as found by BLAST when using **SyntenMapper** or you can set it manually. Two spaces are used to separate the gene name and direction arrow from any numbers.



6.3 Query Numbering

The figure for T2 above shows how numbers inside the () parenthesis correspond to the query's amino acid (AA) or BP numbers, while those outside correspond to the BP's of the contig itself. The innermost AA/BP numbers, closest to the query name, are used to indicate where box shading starts and ends. The numbers on the other side of the | symbol corresponds to the first or last AA/BP of the full-length query. In the example above, we can see T2 is 59 AA long, however, shading is only applied between AA's 1 – 35. If we were not interested in the shading option, we could just set the “end shading” number to 59 so that the entire box was filled with the same color.

6.4 Contig Numbering

Continuing outside the () parenthesis, a single space separates similar sets of numbers used to indicate where on the contig to draw the box. These are simply the contig BP numbers that correspond to the start and stop of the shading and full-length query numbers as discussed above. Again, the numbers closest to the query name refer to shading, while those on the other side of the | symbol correspond to the full-length query sequence. When using DNA query sequences, the difference between the first and last shading and full-length numbers should be the same for the contig space as it is for the query space. Since our example query sequences are proteins, the difference between the first and last contig DNA coordinates have been multiplied by three to account for codons. It is important to note that **SyntenDrawer's** **shade** option assumes there are no introns in coding sequences. If you need to take introns into consideration, you will need to manually edit your CMAPs using a text editor and represent each exon as a separate gene entry (you may want to append a '.1', '.2', etc. to the name of each segment). One more space in each direction is used to separate the final [] square brackets which denote the boundary of the query entry. We can now visualize this full gene as the following line:

```
[ CONTIG_BP_START|CONTIG_BP_START_SHADING (FULLGENE_BP_START|GENE_BP_START_SHADING <== GENE_NAME GENE_BP_END_SHADING|FULLGENE_BP_END) CONTIG_BP_END_SHADING|CONTIG_BP_END ]
```

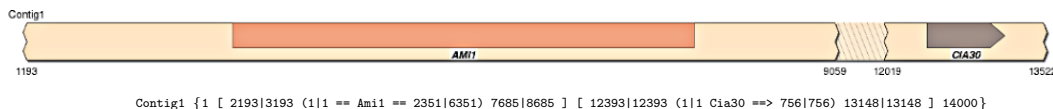
Multiple genes can be inserted inside a contig, they just need to be separated by two spaces on either side of them.

6.5 Valid Numbers

In order for a CMAP line to be valid, numbers to the left of the query name should be less than the corresponding numbers on the right. For example, the first contig BP number must always be less than the end BP number. Query sequences must not be longer than the size of their residing contig, and shading numbers must always be equal to or fit within the range of the full-length sequence.

6.6 Removing Gene Directions

You can remove directionality from a gene entirely by placing == on both sides of a gene name. The following example shows *Ami1* drawn as a rectangle without direction:



6.7 Customizing CMAP Files

Now that you understand the CMAP format, you can edit CMAP files to make the following adjustments to your image:

- Fix errors if you think BLAST made them, this can happen particularly with TBLASTN if you are mapping protein sequences.
- Adjust gene shading to highlight particular areas of interest.
- Manually insert genes of importance if you need to use a certain BLAST E-value cutoff but a gene of interest is being left out.
- Remove gene directions and display them as basic rectangles.
- Make manual images for teaching purposes etc.

7 Acknowledgements

Support for this project is made possible through funding provided by the 2013-2015 USDA-APHIS Farm Bill 10201 and 10007 Programs; The U.S. Department of Agriculture Agricultural Research Service (USDA-ARS); the USDA-ARS Floriculture and Nursery Research Initiative with the sponsorship of AmericanHort and the Society of American Florists; Rutgers University Department of Plant Biology and Pathology; and the ARS Research Participation Program administered by the Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy (DOE) and the USDA. ORISE is managed by ORAU under DOE contract number DE-AC05-06OR23100.

8 Recent Changes

- Version 1.3.1 (Dec 11th, 2017) Minor corrections to **SyntenDrawer**: Bug fixes to correct certain output flags (`-o`, `-b`) not being interpreted correctly. Special thanks to Praveen Vijayakumar for pointing this out.
- Version 1.3.0 (Feb 27th, 2017) Changes to **SyntenMapper**: Bug fix for case where gene overlap merges could get stuck in a loop. Changes to **SyntenDrawer** and CMAP data format: After you run **SyntenMapper**, you can now remove directions by replacing arrows with `==` on *each* side of a gene name. See the CMAP section for details. With **SyntenDrawer**, you now have the ability to remove gene names with the `-gene-names` flag set to false. This will leave empty white boxes so the user can add in names using external software. White boxes can also be removed if desired by also calling `"-box false."` Also fixed a bug where start and end BP numbers were not being displayed correctly after certain long contigs were split. Special thanks to David Roe for pointing out this error.
- Version 1.2 (May 27th, 2016) Changes to **SyntenDrawer**: Improved detection of overlapping flip-arrows. Determine distance to move overlapping arrows with `-arrows` flag, `-overlaps` flag removed. Added `-gene-font` and `-contig-font` flags to adjust gene and contig names font sizes. Merged `-font` / `-gname-font` flags to new name of `-genome-font`. Fixed an error with user-argument processing. Gene connections that pass through an adjacent genome are now drawn as translucent dashed lines.
- Version 1.1 (May 3rd, 2016) First public version of scripts and this file.