SOFTWARE TOOL ARTICLE

CyAnimator: Simple Animations of Cytoscape Networks

[version 1; peer review: 1 approved, 1 approved with reservations]

John H. Morris ¹, Dhameliya Vijay², Steven Federowicz³, Alexander R. Pico ⁴, Thomas E. Ferrin ⁵

¹Resource for Biocomputing, Visualization and Informatics,, University of California, San Francisco, CA, 94143, USA

²Dhirubhai Ambani Institute of Information and Communication Technology, Gujarat, India

³Intrexon Inc., San Diego, CA, 92121, USA

⁴Gladstone Institutes, San Francisco, CA, USA

⁵Resource for Biocomputing, Visualization and Informatics, University of California, San Francisco, CA, USA

v1 First published: 05 Aug 2015, 4:482
<https://doi.org/10.12688/f1000research.6852.1>
 Latest published: 30 Dec 2015, 4:482
<https://doi.org/10.12688/f1000research.6852.2>

Abstract

CyAnimator (<http://apps.cytoscape.org/apps/cyAnimator>) is a Cytoscape app that provides a tool for simple animations of Cytoscape networks. The tool allows you to take a series of snapshots (CyAnimator calls them frames) of Cytoscape networks. For example, the first frame might be of a network shown from a "zoomed out" viewpoint and the second frame might focus on a specific group of nodes. Once these two frames are captured by the tool, it can animate between them by interpolating the changes in location, zoom, node color, node size, edge thickness, presence or absence of annotations, etc. The animations may be saved as a series of individual frames, animated GIFs, MP4 movies, or H.264/MOV movies. CyAnimator is available from within the Cytoscape App Manager or from the Cytoscape app store.

Keywords

Cytoscape , CyAnimator , network , animation

This article is included in the Cytoscape gateway.

Open Peer Review

Approval Status

	1	2	3
version 2			
(revision)			
30 Dec 2015		view	view
		↑	
version 1			
05 Aug 2015	view	view	

1. **Giovanni Scardoni**, University of Verona, Verona, Italy
2. **Nathan Salomonis** , Cincinnati Children's Hospital Medical Center Research Foundation, Cincinnati, USA
3. **Piet Molenaar**, Cincinnati Children's Hospital Medical Center Research Foundation, Cincinnati, USA

Any reports and responses or comments on the article can be found at the end of the article.

This article is included in the **Bioinformatics** gateway.

Corresponding author: John H. Morris (scooter@cgl.ucsf.edu)

Competing interests: No competing interests were disclosed.

Grant information: JHM was funded by NIGMS grants P41-GM103504 and P41-GM103311. TF is funded by NIGMS grant P41-GM103311. AP is funded by NIGMS grant P41-GM103504.

Copyright: © 2015 Morris JH *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. Data associated with the article are available under the terms of the [Creative Commons Zero "No rights reserved" data waiver](#) (CC0 1.0 Public domain dedication).

How to cite this article: Morris JH, Vijay D, Federowicz S *et al.* **CyAnimator: Simple Animations of Cytoscape Networks [version 1; peer review: 1 approved, 1 approved with reservations]** F1000Research 2015, 4:482 <https://doi.org/10.12688/f1000research.6852.1>

First published: 05 Aug 2015, 4:482 <https://doi.org/10.12688/f1000research.6852.1>

Introduction

Biological networks are typically represented as nodes and edges (node-link diagrams) that might represent the pathways, signaling cascades, interactions between proteins, and other relationships between biological entities. The problem with this representation is that it makes it seem like these relationships are static, but it is well known that biological networks are dynamic, adapting and changing in response to the cell cycle, environmental conditions, development, and, over longer periods of time, evolution. One of the best methods to represent these changes is to take advantage of motion, showing changes by interpolating between the two states. This use of motion is one way to address “change blindness”¹, which makes it difficult to detect the difference between two images when they are shown in succession. This is important both for presentation of results to collaborators and the broader scientific community and for the exploration of data by individual researchers.

Cytoscape² is one of the most common tools used to visualize and analyze biological networks. It provides very powerful visualization tools to map a variety of categorical and numeric data into visual attributes associated with the nodes and edges of node-link diagrams. Unfortunately, Cytoscape does not provide any inherent animation capabilities, making it difficult to use interpolation to detect changes between two states, however, Cytoscape does provide a rich infrastructure for extending its core functionality through “apps”³. Several Cytoscape apps provide some animation capabilities. For example, VistaClara⁴, 3DScape⁵, and clusterMaker⁶ provide support to animate through the columns of a heat map. DynNetwork (<http://apps.cytoscape.org/apps/dynnetwork>) reads specially constructed input files that specifically encodes changes to the network over time. Of these, only clusterMaker2 and DynNetwork are available in Cytoscape 3, and offer relatively limited restrictive animation capabilities (in the case of clusterMaker2) or require construction of special input files in advance (in the case of DynNetwork). None of the available tools allows for the animation of arbitrary changes to the network topology or visualization.

CyAnimator attempts to fill this gap by providing a tool that supports animation by allowing the user to designate particular network views as “key frames”. These frames represent the state of the network at particular moments in time including the current list of nodes and edges and the visual attributes of those nodes and edges. The user may then arrange those frames in a desired sequence and CyAnimator will interpolate between the frames resulting in a smooth animation between the states of the network. The resulting animation may be saved as a movie.

Methods

Implementation

The main object in CyAnimator is a CyFrame, which contains all of the information about the visual attributes of the network background, annotations, nodes, and edges. This information is stored in a series of maps indexed either by the internal unique identifier (SUID) of the node or edge or by a hashCode of the annotation object, if the stored object is an annotation. CyAnimator makes

heavy use of the visual property system in Cytoscape and the pseudocode for the general loop for populating a CyFrame is:

```
getNetworkVisualProperties();
foreach Node:
    getNodeVisualProperties();
foreach Edge:
    getEdgeVisualProperties();
foreach Annotation:
    getAnnotationVisualProperties();
getImage();
```

The captured image is used to show thumbnails to the user in the CyAnimator dialog (see below). In general, getting the visual properties results in a series of maps, one map for each property. These maps are used to restore the state of the network or to form the basis for the interpolations. In hindsight, this was an unfortunate way to implement the storage of the individual visual properties. In the current implementation, when we want to add support for a new visual property, we must create a new map for it, add code to populate the map, add code to explicitly set that visual property on a network view, and finally add the appropriate code to interpolate between two values of that visual property. This would have been much easier if we had instead stored a map of the form:

```
Map<Long, Map<VisualProperty, Object>>
```

where the Long key would be the SUID of the object. We could then maintain a list of VisualProperties, or even a map linking VisualProperty to Interpolator. This would greatly facilitate adding new visual properties to the system. We plan on refactoring CyAnimator in a future release to implement this new approach.

When the user clicks on a frame in the CyAnimator dialog or during the animation, the stored visual properties must be mapped onto the current network. This is a straightforward mapping except in circumstances where the current network does not have the node, edge, or annotation that the stored CyFrame had or the CyFrame did not have a node, edge, or annotation that the current network has. In the first case, the missing object is added to the network and styled using the stored information. This approach depends on the fact that deleted nodes and edges are not removed from the CyRootNetwork, so the topology of the network at the time the CyFrame was stored may be recreated. In the second case, the CyFrame sets the visibility of the object to **false**, resulting in a network view that is consistent with the saved visual properties.

Interpolation between CyFrames is handled by the Interpolator, which maintains lists of interpolators for node visual attributes, edge visual attributes, annotation visual attributes, and network visual attributes. The Interpolator *makeFrames()* method takes as input the list of CyFrames, the user saved (key frames) and returns an array of frames that includes interpolations between each pair of key frames. So, if the user has requested 30 frames between each key frame, and created 5 key frames, *makeFrames()*, will return 121 frames, which is the number of key frames to interpolate between (key frames - 1), multiplied by the number of interpolations between each key frame, plus the initial key frame ((keyframes - 1)*interpolations + 1).

CyAnimator currently supports 5 different interpolation types:

- Linear:** Values are linearly interpolated between the two key frames
- Bezier:** Used for transparency to fade in slowly and fade out slowly
- Instant:** No real interpolation – switch between values 1/2 way through the set of frames
- Fade-in/Fade-out:** Linearly fade out then fade in. Essentially this is as close as we can get to a crossfade effect.
- None:** This visual attribute is not interpolated.

Table 1 shows the interpolated visual attributes and the type of interpolation. In the current version of Cytoscape, there is no mechanism to get the list of node custom graphics for a particular node,

so charts, gradients, and other custom graphics can not be interpolated. CyAnimator also doesn't currently support interpolation of arrow colors. This is an oversight that will be rectified in a future version.

To implement the creation of a movie, three different mechanisms are used. In each case, an image file is created for each interpolated frame. One option the user has is to just write out the individual frames and allow them to use whatever movie making tool they desire. We have also implemented an animated GIF writer using the native Java ImageIO library. Finally, for creating MP4 or H.264 movies, we utilize the Xuggler (<http://www.xuggle.com/xuggler/>) Java library, which includes the necessary video codecs.

Operation

To bring up CyAnimator select **Apps→CyAnimator**. This will bring up an empty CyAnimator dialog (Figure 1). Note that

Table 1. Visual Property Interpolations.

Object Type	Visual Attributes	Interpolation
Network	Background Paint, Scale Factor, Size, Center X Location, Center Y Location Title	Linear None
Node	Position, Border Width, Label Font Size, Size, Fill Color, Border Color, Label Color Transparency, Border Transparency Label Font Face, Shape, Border Line Type Label Position, Custom Paint	Linear Bezier Fade-out/Fade-in Instant None
Edge	Stroke Color (Unselected), Label Color, Label Font Size, Width Transparency Line Type, Source Arrow Shape, Target Arrow Shape Label Source Arrow Unselected Paint, Target Arrow Unselected Paint	Linear Bezier Instant Fade-in/Fade-out None
Annotations	Position, Size, Color, Text	Linear Instant

Figure 1. The CyAnimator Dialog showing two key frames.

CyAnimator is only able to animate between networks in the same network collection. Starting CyAnimator on a new network collection will create a new, empty, CyAnimator dialog. Once a CyAnimator dialog is open, the general workflow would be to manipulate the network to what you want it to look like at the start of your movie, then select **Add Frame** to add the frame to CyAnimator. Once the frame has been added, you can modify your network to what you want it to look like in the next frame of your movie and then again select **Add Frame**. Note that CyAnimator will do all of the interpolation to get from one frame to another, so the manipulations of the network can include a variety of changes, including changes in color, position, zoom, annotations, etc. Repeat this process until you are happy with your movie, then simply press the record (●) button. If you want more time between any two frames, use the context menu to change the **Interpolations** value.

CyAnimator dialog

The CyAnimator dialog provides the main interface to CyAnimator, including the following controls:

- Add Frame** Add the current network view as a frame to the animation.
- Play the animation by interpolating through each frame.
- Pause the currently playing animation.
- Stop the currently playing animation.
- Step backwards to the previous interpolated frame.
- Step forwards to the next interpolated frame.
- Bring up the Output Options dialog and record a movie or (optionally) save each of the interpolated frames.
- Speed slider** The speed slider controls the speed of the animation (but is ignored for the recorded movies, which uses its own Frames Per Second option).

Frame context menu

In addition to the menu bar controls discussed above, each frame provides a context menu that's available through a **Right-click** (or **Control-click** on Mac). These menu items are:

- Interpolate** Select the number of frames to interpolate between this frame and the previous frame. This allows you to control the duration of the interpolation—to increase the speed between this frame and the previous frame, select fewer interpolation frames; more interpolation frames will result in a longer animation.
- Delete** Delete the current frame.
- Move Right** Move this frame right (later in the animation).
- Move Left** Move this frame left (earlier in the animation).

Output options

The Output Options Dialog (see [Figure 2](#)) provides the controls to choose the type of video you want to produce, options about the resolution and speed of the video, and the location of the video output.

Video types

Frames. This is the simplest of the video types. This will output each frame as a “.png” file at the requested resolution (see ‘Resolution’ below) into the Video location directory specified. To make a movie, you could use any of the standard video packages that accept individual frames (e.g. iMovie on Mac). Note that the Frames Per Second option doesn't make sense for this output type since we're only writing individual frames (no time encoding), so this option is disabled (grayed out) in the interface.

GIF. This will output an animated GIF of the interpolated frames. Animated GIF files are easy to show on web sites, but are not the currently accepted standard format. On the other hand, animated GIF files are computationally very easy to produce.

Figure 2. The Output Options Dialog showing the production of an H.264 video file at three times the screen resolution and 50 frames/second.

MP4. This will output a raw MPEG4 file of the interpolated frames. Producing an MP4 (and a MOV/H264) can be computationally expensive, and will require more time and memory to complete.

MOV/H264. This will output an MPEG4-encoded video of the interpolated frames. The MPEG4 video is then encapsulated in an H.264 container. This is considered the current standard for web deployment of video content. Producing an MOV/H264 can be computationally expensive, and will require more time and memory to complete.

Frame options

Frames Per Second. This controls the speed of the movie in terms of the number of frames per second. Smooth animations should be set to at least 30 frames per second.

Resolution. This controls the resolution of the output frames. The units are % expansion, so a Resolution of 300 would result in a 300% (or 3X) expansion of the output image. This is extremely important for high-quality videos. We recommend at least a 3X (Resolution of 300) expansion for any published video.

Example movie

A sample movie is provided (see [Supplementary_File_1.mov](#)) that was created with CyAnimator and Cytoscape 3.3. This movie used `galFiltered.cys`, which may be located in the `sampleData` subdirectory of the Cytoscape installation directory (e.g. `/Applications/Cytoscape_v3.2.1/sampleData` on a Mac). The movie consisted of 4 key frames:

1. the full network
2. a view focused on MCM1
3. the same view, but peripheral nodes were moved out of the way, and five annotations were added: three text annotations, a shape annotation around the three text strings, and an arrow pointing to MCM1
4. another capture frame of the same view

Supplementary material

Sample movie created with CyAnimator and Cytoscape 3.3. This movie used `galFiltered.cys`, which may be located in the `sampleData` subdirectory of the Cytoscape installation directory (e.g. `/Applications/Cytoscape_v3.2.1/sampleData` on a Mac).

[Click here to access the data.](#)

Conclusions

CyAnimator is an important addition to the suite of Cytoscape apps. It provides an easy tool to interpolate between different states of a network and may be used to animate changes over time, condition, or treatment. One missing feature of CyAnimator is the ability to save the key frames of an animation as part of a session. We will be adding support for this in a future release.

Software availability

Software available from

<http://apps.cytoscape.org/apps/cy animator>

Latest source code

<https://github.com/RBVI/CyAnimator>

Archived source code as at the time of publication

<http://dx.doi.org/10.5281/zenodo.206847>

License: Lesser GNU Public License 3.0

<https://www.gnu.org/licenses/lgpl.html>

Author contributions

JHM wrote the manuscript and enhanced the app. SF developed initial version of the app for Cytoscape 2.8. VD ported the app to Cytoscape 3. AP and TF supervised app development and provided input on the manuscript.

Competing interests

No competing interests were disclosed.

Grant information

JHM was funded by NIGMS grants P41-GM103504 and P41-GM103311. TF is funded by NIGMS grant P41-GM103311. AP is funded by NIGMS grant P41-GM103504.

Acknowledgments

The authors wish to acknowledge the support of the NRRB Academy.

References

1. Simons DJ, Ambinder MS: **Change blindness: theory and consequences.** *Curr Dir Psychol Sci.* 2005; **14**(1): 44–48.
[PubMed Abstract](#)
2. Shannon P, Markiel A, Ozier O, *et al.*: **Cytoscape: a software environment for integrated models of biomolecular interaction networks.** *Genome Res.* 2003; **13**(11): 2498–2504.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
3. Pico AR, Bader GD, Demchak B, *et al.*: **The Cytoscape app article collection [v1; ref status: not peer reviewed, <http://f1000r.es/3tv>].** *F1000Res.* 2014; **3**: 138.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
4. Kincaid R, Kuchinsky A, Creech M: **VistaClara: an expression browser plug-in for Cytoscape.** *Bioinformatics.* 2008; **24**(18): 2112–2114.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
5. Wang Q: **3DScape: three dimensional visualization plug-in for Cytoscape.** *Nature Precedings.* 2011.
[Reference Source](#)
6. Morris JH, Apeltsin L, Newman AM, *et al.*: **clusterMaker: a multi-algorithm clustering plugin for Cytoscape.** *BMC Bioinformatics.* 2011; **12**(1): 436.
[PubMed Abstract](#) | [Publisher Full Text](#) | [Free Full Text](#)
7. Dhameliya V, Wu A, Morris S, *et al.*: **CyAnimator: F1000 Release.** *Zenodo.* 2015.
[Data Source](#)

Open Peer Review

Current Peer Review Status:

Version 1

Reviewer Report 02 September 2015

<https://doi.org/10.5256/f1000research.7370.r10174>

© 2015 Salomonis N. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Nathan Salomonis

Cincinnati Children's Hospital Medical Center Research Foundation, Cincinnati, OH, USA

The authors present a useful Cytoscape app that fills in an important void not provided by the parent application: dynamically transitioning between distinct network states or associated visualized datasets. The application is thus novel and provides an important function in Cytoscape that could significantly assist with biological interpretation.

Major Point 1: The authors state: "Note that CyAnimator is only able to animate between networks in the same network collection."

Can a network collection be different networks that share some or all of the same nodes? If not, I understand how this could be a technical limitation of the app, but would be of particular use when curating distinct pathway states from an original network. Often, adding, subtracting, moving and re-coloring nodes in succession to different states in the same network instances is challenging, but storing these as distinct snap shots that can be animated together is extremely powerful. If not present, is this a future option?

Major Point 2: The shown movie is a reasonable simple demonstration of CyAnimator, however, this does not show of the more interesting dynamic options of the tools, such as transitioning between different developmental or temporal states (node color attributes, aka gene expression fold differences) or temporally distinct interactions for a given pathway (e.g., chain of metabolic events). While I realize this would be some work to do, this would more strongly demonstrate the capabilities of the software.

Minor Point: One of the powerful uses of this software would be via programmatic generation of animations, by defining network states and visual parameters. While I know that some of these capabilities are embedded in Cytoscape 3, are these possible within CyAnimator for the purpose of making animations outside of the UI? If so, can the authors provide an example set of code to do this? A link to documentation or a tutorial would also be helpful for the end-users to properly use the software.

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 14 Dec 2015

Scooter Morris

Thank you for your review. I have just submitted version 2 of the CyAnimator paper, which corresponds to the latest release of the App on the Cytoscape App store. The new app version is a significant rewrite and offers several advantages over the previous version, which I believe address most of your concerns. To answer point-by-point:

Major Point 1: *The authors state: "Note that CyAnimator is only able to animate between networks in the same network collection."*

Can a network collection be different networks that share some or all of the same nodes? If not, I understand how this could be a technical limitation of the app, but would be of particular use when curating distinct pathway states from an original network. Often, adding, subtracting, moving and re-coloring nodes in succession to different states in the same network instances is challenging, but storing these as distinct snap shots that can be animated together is extremely powerful. If not present, is this a future option?

Yes, a network collection in Cytoscape 3 includes multiple networks that share nodes and edges. Essentially, a network collection can be thought of as a set of projections of a single network, very similar to the Cytoscape 2 model. Cytoscape 3 allows users to create multiple network collections and the nodes and edges aren't shared between them. Both versions of the App support this capability.

Major Point 2: *The shown movie is a reasonable simple demonstration of CyAnimator, however, this does not show of the more interesting dynamic options of the tools, such as transitioning between different developmental or temporal states (node color attributes, aka gene expression fold differences) or temporally distinct interactions for a given pathway (e.g., chain of metabolic events). While I realize this would be some work to do, this would more strongly demonstrate the capabilities of the software.*

The sample movie has been updated as suggested to show three different expression fold changes and a heat strip representation showing all three at once.

Minor Point: *One of the powerful uses of this software would be via programmatic generation of animations, by defining network states and visual parameters. While I know that some of these capabilities are embedded in Cytoscape 3, are these possible within CyAnimator for the purpose of making animations outside of the UI? If so, can the authors provide an example set of code to do this? A link to documentation or a tutorial would also be helpful for the end-users to properly use the software.*

This version of the App introduces some simple commands that may be used by app developers to create frames. At this point, the capability remains somewhat rudimentary and doesn't allow programmatic changes to frame ordering or transition duration. This will be added in a future version and more detailed documentation will be written at that time.

Competing Interests: No competing interests were disclosed.

Reviewer Report 18 August 2015

<https://doi.org/10.5256/f1000research.7370.r9880>

© 2015 Scardoni G. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Giovanni Scardoni

Center for BioMedical Computing, University of Verona, Verona, Italy

The App is very useful and easy to use. As the article highlights It is very important to have a tool allowing the user to create animations from static frames of a network, and there are no other Cytoscape apps with the same characteristics of CyAnimator. The paper is well written and both implementation and features are easy to understand.

Minor changes:

- The sentence about the future implementation of the maps in the implementation section should be moved to the conclusions session or to the supplementary materials. So the reader can concentrate on the current features of the app.
- Similarly, the last sentence of the conclusions sounds better as "The ability to save the key frames of an animation as part of a session will be added".

Competing Interests: No competing interests were disclosed.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research