LLNL Climate First Quarter Report 2018

POP: Jan - Mar 2018

Summary

This quarter we have fixed several major and minor issues affecting both VCS and vcs.js. One large improvement in vcs.js was the updating of all web module dependencies and the reworking of automated testing. We now have image-comparison based testing running automatically on Travis for all vcs.js PRs. A big improvement within VCS itself involved fixing axis scaling for plots involving vectors, streamlines, and glyphs. Previously the vectors/streamlines/glyphs themselves were scaled to fit within the viewport before plotting, but this is incorrect for these kinds of graphics. Instead we now scale the datasets upstream in the pipeline. This caused us to also need to rethink and fix how markers (implemented using glyphs) are sized throughout the codebase. As a result, we now have markers which are sized (and shaped) consistently, regardless of plot size/aspect or world coordinates range.

Description of work performed

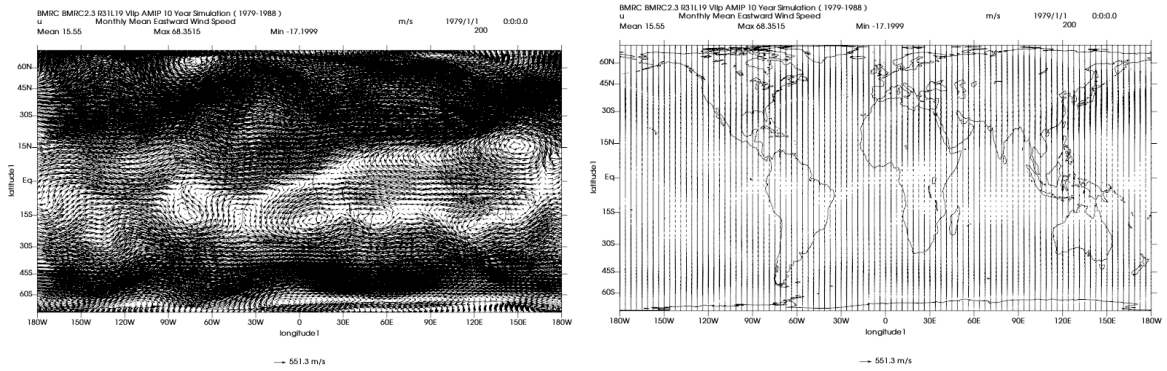
During the first quarter, good progress was made on several items from the SOW. A first pass on error bars for 1D plots has been implemented and is in the software review process. In addition, a complete automated testing infrastructure for vcs.js was implemented and deployed so that new PRs are run through the complete test suite on Travis and the results submitted back to github as a “check”. This automated testing framework includes the launching of a full-feature test server to provide end-to-end testing of the web tools, including requests to generate images on the server and do image comparison of the results in the browser (headless Chrome provided by Travis).

Also many new bugs and issues were discovered and fixed during the quarter. One of the more significant bug fix efforts involved broken vector/streamline/glyph plots in the presence of axis scaling, the details of which are described (and depicted with screenshots) below.

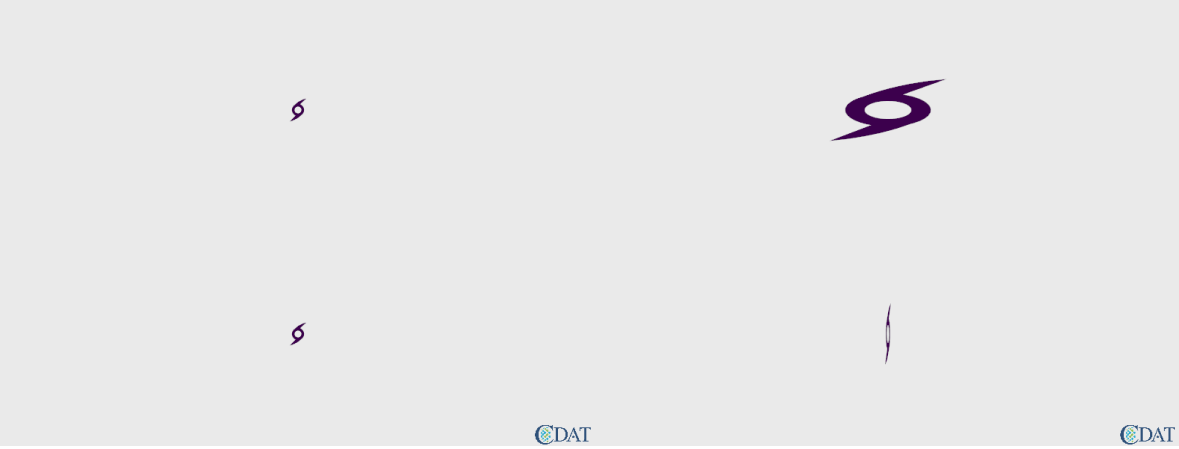
Bug fixes and enhancements to VCS

- BUG: PR #305 (merged) - Do not mark attribute as VECTORS so that are not removed. The issue was that vtkTransformPolyData changes a VECTORS attribute from double to float, then vtkAppendPolyData removed that attribute because of non-matching attributes (double and float).
- ENH: PR #247 - Error bars 1D plots. While this is still a work in progress, it provides initial API and working implementation for X and Y error bars on 1D plots.

- ENH: PR #319 (merged) - Added test for concave fill areas
- BUG: PR #306 (merged) - Fix various issues with resizing plots. This fix also got rid of the Canvas bgX and bgY variables, so there is now just a single source of truth for the size of a plot. It is stored in Canvas as 'width' and 'height', VTKVCSBackend no longer maintains its own separate notion of the size.
- ENH: PR #304 (merged) - Sometimes no streamlines were drawn if startseed was not inside the domain. To reduce the burden on users, we implemented a default startseed of “None” to mean the middle of the domain.
- BUG: PR #331 (pending merge) - Previously vectors, glyphs, and streamlines were all scaled by the “fitToViewport” method, which produced incorrect results when scaling axes. This fix changes things so that the input data is scaled rather than the vectors/glyphs/streamlines themselves, which results in correct plots under axis scaling. However, the fix broke many other things like animations, pattern scaling, and markers across the entire repository. These side-effect errors were also corrected.



In the images above, the right picture used “area_wt” scaling on the y-axis, and because the vectors themselves were scaled, they all appear to point straight up and down. The left image above shows the new (correct) behavior, where the input data is scaled rather than the vectors.



The two images above depict hurricane markers placed into plots sharing (vertically) an 800 x 600 canvas. In both plots, the top half of the canvas has been used to plot a hurricane with default world

coordinates [0, 1, 0, 1], while the bottom half of the canvas was used to plot another hurricane with world coordinates [-10, 10, 0, 1]. The plot on the right (representing the previous, incorrect behavior) gave both hurricane markers a size of 1, but as evident from the image, the size and aspect ratio of the markers was dependent on the world coordinates (and in fact on the canvas size as well). The plot on the left (representing the new, correct behavior) gives both hurricane markers a size of 20, which results in a final screen size of approximately 20 pixels, regardless of world coordinates or canvas size.

Bug fixes and enhancements to vcs.js

- ENH: PR #33 (merged) - Infrastructure update. This enhancement updated all of our module dependencies to the latest version, and re-worked the testing infrastructure. We now have image-based comparison tests run automatically on Travis for each new PR.
- BUG: PR #23 (merged) - Add guard to make sure graphics methods have any attributes we're going to expose as properties.
- BUG: PR #24 (merged) - Fixed bug which resulted in not being able to clear plots.
- BUG: PR #25 (merged) - Fixed bug which caused vcs server to crash in the presence of variables with lon or lat, but not both.
- BUG: PR #27 (merged) - Fixed bug in web api which resulted in broken resize functionality.
- BUG: PR #28 (merged) - Fixed bug in web api which resulted in not being able to make plots on top of each other (e.g. vector or streamline on top of boxfill).
- ENH: PR #29 (merged) - Created a new rpc method ('getgraphicsmethodvariablecount') which exposes underlying vcs method to tell how many variables are required by any graphics method.
- BUG: PR #37 (merged) - Provided a workaround which allows plotting isolines from the web. Until numeric default values like `1e-20` (and others) are systematically removed from vcs, we need to do some special checks from the web to prevent round-trip communication from changing float values to int (which causes numpy to raise exceptions).

Work planned for next quarter

The major task for the next quarter will include the update to use the modern OpenGL2 rendering backend to VTK, as described in the SOW. As this will require changes to all of the VTK-based pipelines, it may make sense to additionally perform the SOW tasks involving analysis of the pipelines from a performance standpoint. Additional improvements to vector and projected plots are also planned.

Reimbursable costs incurred

Not applicable

Labor effort expended by labor category

Not applicable