VAPOR Product Roadmap

Visualization and Analysis Software Team

October 2017

VAPOR

Introduction

In 2015 the VAPOR team began a major refactoring of the VAPOR codebase aimed at addressing a myriad of limitations in the original design, while simultaneously supporting and developing the VAPOR version 2.x code stream. The refactoring effort, heretofore referred to as "VAPOR3", is still underway and expected to be completed in 2018. The most significant goals of VAPOR3 are:

Extensible architecture: While VAPOR2 was largely a monolithic collection of code, the aim of VAPOR3 is a clean separation of well-defined components and APIs that follow modern software engineering best practices, and facilitate extensibility by VAPOR team members as well as community contributors in areas of development such as:

- data readers for new file formats, model outputs, etc.;
- data operators (e.g. visualizers, computational modules, etc.); and
- alternate user interfaces (e.g. scripting, web based).

Flexible data model: VAPOR2's support for computational meshes was limited to regular grids only, and had no formal support for coordinate systems or units of measure. The VAPOR3 data model takes inspiration from the widely used Climate Forecast (CF) conventions, and will be capable of supporting a significant subset of the defined CF mesh types; horizontal, vertical, and temporal coordinate systems; and units of measure.

Improved usability: New functionality had been added to VAPOR2 in a somewhat ad hoc fashion resulting in a GUI that is often overloaded with features and inconsistent in its behavior and operation. The result is that the GUI is often difficult to use, particularly for new users. The VAPOR3 GUI will be restructured to follow best practices for user interface design such as separating commonly used and advanced features, and striving for consistency throughout the application.

Development plans

The most significant areas of new development are discussed below. Many of these areas are enabled by, or closely tied to, the VAPOR3 refactoring effort.

VAPOR Data Collection (VDC)
The VDC is the cornerstone of VAPOR's large data handling capability, providing the ability to compress and/or progressively access gridded data. VAPOR3 will offer version 3 of the VDC. Version 3 of the VDC is inspired by the CF conventions, and is intended to allow representation

of most CF compliant data. Much of the development work on version 3 of the VDC is now complete, and the new file format is capable of storing the same grids as VDC2, but also provides formal support for horizontal and vertical coordinates as defined in the CF conventions, as well as units of measure as supported by Unidata's UDUNITS2 package. Work is now underway to add support for unstructured grids via the UGRID conventions. Future work on the VDC includes:

- Adding support for alternate data compression schemes, and advanced wavelet encoders (e.g. SPIHT and SPECK).
- Adding progressive access representation scheme(s) for unstructured data
- Providing an MPI-compatible, parallel API with C and Fortran bindings, thus enabling numerical models to write (or read) data directly to (or from) the VDC format.
- Developing VDC readers/writers for other data analysis applications. E.g. NCL, VisIt, and ParaView.

Scripting interface
Though VAPOR is primarily an interactive application for exploratory data analysis, a frequent request from the VAPOR user community is the ability to control their analysis operations via a scripting language that would facilitate batch and repeated operations. Though a final decision has not been made, at present Python is the most likely candidate for such an interface.

OpenGL 4.0 support
All rendering in VAPOR is performed with OpenGL. Presently VAPOR uses the venerable version 2.x OpenGL API, which has reached EOL. Migrating to a more recent version of this graphics library will be necessary in the future.

Performance optimization
A number of components of the VAPOR tool suite are computationally intensive and could benefit from optimization and parallelization. VAPOR's wavelet transform and flow integration modules, for example, could readily take advantage of current/emerging many-core architectures.

Distributed memory parallel backend
Thus far VAPOR has been able to satisfactorily support large data sets by relying on the VDC's progressive data access capability. The long term viability of this approach to increasingly large data sets is not known, and at some point it may be necessary to refactor VAPOR's backend data operators and visualizers to support distributed memory parallelism.

Unstructured grid rendering
Supporting unstructured grids (e.g. MPAS, ICON, HOMME) will require substantial changes to at least two of VAPOR's current premiere visualization algorithms: isosurface display and direct volume rendering.

Expanded python support

VAPOR currently relies on Python, and in particular the NumPy, matplotlib, SciPy modules, for performing numerous tasks such as deriving new quantities, basic 2D plotting, and quantitative analysis. However, the interaction with the Python interpreter is highly limited. We plan for a much tighter coupling between Python and VAPOR in the future, allowing users to more seamlessly interact between the two. Such expanded capability would open the door for VAPOR to invoke NCL Python calculation and plotting modules, and vice versa, for example.

In situ visualization

There is a much interest in the HPC communities in *in situ* visualization. Though the viability of this approach is uncertain due to its numerous limitations (e.g. supporting exploratory science, scheduling challenges, etc.), and the lack of maturity of the supporting technologies, it is nevertheless a direction that we must track and ultimately may need to support.

VAPOR3 Development Roadmap October 2017					
Capability	2017 V3.0 (beta)	2018 V3.0	2018 V3.1	2019 V3.2	Future V3.x
VAPOR Version 2 enter EOL			✓		
Direct Volume Rendering (regular grids)*		✓			
Isosurfaces (regular grids)*		✓			
Statistics*	✓				
Line plotting*	✓				
Contour lines*	✓				
2D data*	✓				
Time animation*	✓				
Geo-referenced images*	✓				
Unstructured grids (2D)	✓				
Multi-dataset visualization	✓				
VDC importer for NCL			✓		
Direct Volume Rendering (unstructured grids)			✓		
Isosurfaces (unstructured grids)			✓		
Flow visualization*			✓		
Key frame animation*			✓		
Python calculation engine*				✓	
VDC support for general unstructured grids via UGRIDs				✓	
OpenGL 4.x support				✓	
Python scripting interface				✓	
Accelerator support				✓	
Parallel (MPI) VDC API				✓	
Parallel (MPI) rendering backend					✓
Progressive data access / compression (unstructured grids)					✓
Advanced wavelet encoders and/or alternate compressors					✓
Web interface					✓
Expanded python support					✓
In situ visualization					✓

Anticipated release schedule for major new VAPOR components. Starred (*) items are those that already exist in VAPOR2 and must be migrated to the VAPOR3 architecture.