2020 Year End Report for The Stardust Project:

A Sample-based, Perceptually- and Cognitively-driven Visual Analysis of Massive Scientific Data Using an Asynchronous Tasking Engine

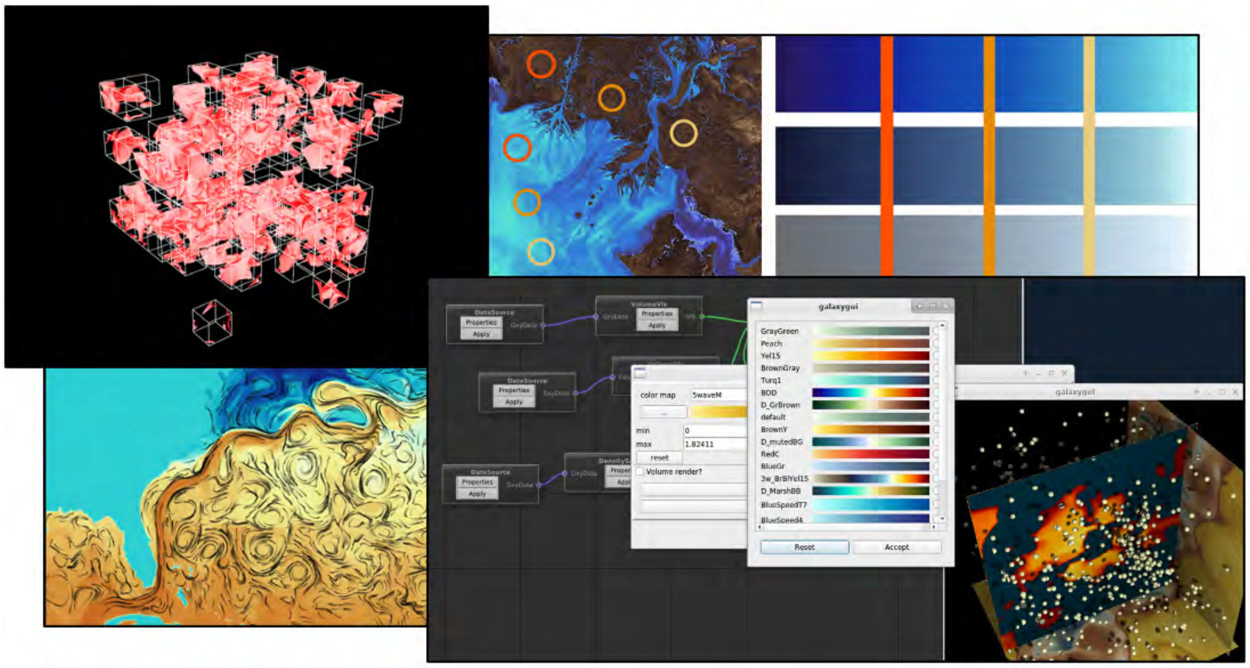
James Ahrens	LANL
David Rogers	LANL
Francesca Samsel	UT Austin
Greg Abram	UT Austin
Paul Navrátil	UT Austin
Colin Ware	UNH

October 1, 2020

LA-UR-20-27090 004

Contents

1 Executive Summary



We have successfully completed the final year of the Stardust project, as defined in the project proposal *Sample-based, Perceptually- and Cognitively-driven Visual Analysis of Massive Scientific Data Using an Asynchronous Tasking Engine*. Over the course of the project, a variety of research, prototypes and techniques were explored and evaluated for inclusion in the `galaxy` engine. Open source releases of `galaxy` include promising techniques and algorithms for exploration and visualization of massive scientific data. We have completed deliverables for the third year, as evidenced by the work detailed in this report and the artifacts noted below.

1.1 Project Overview

Scientific data from simulation and experiments are increasing in size by orders of magnitude each year. This is due to the advancing quality of scientific simulations driven by the Department of Energy's multi-decadal simulation software, system software and hardware acceleration programs. In addition, sensor quality has improved exponentially due to Moore‘s law producing massive amounts of experimental and observational results. Scientific discovery is dependent upon understanding these massive simulation and sensor-based results.

Our cross-disciplinary team of system, algorithm, perceptual and cognitive experts will work together to produce a tasking engine, sampling and ray-tracing engine prototypes and algorithms. We will evaluate our work's performance and perceptual and cognitive value and get feedback on our approach from our simulation and experimental science partners. The result of this proposal will be:

- An architectural solution that fully utilizes the node architecture of today and future supercomputing platforms using tens to hundreds of computational cores and limited memory per core.
- A generalized data reduction solution that supports intelligent sampling/data reduction at multiple stages of the analysis process.
- A quality perceptual and cognitive solution through the evaluation and comparison of approaches. Demonstrations of our proposed approach enabling scientific discovery for a variety scientific domains.

2 Summary of FY2020 Deliverables and Results

Here we summarize the work done in support of the project deliverables for this year. More detail is available in the slide deck appendix, code repositories and publications cited in the text.

2.1 Engines

We have developed the `galaxy` engine as a platform to explore asynchronous rendering and sampling algorithms in the *Stardust* project. We are continuing to develop sampling and rendering methods to explore the asynchronous approach to data analysis and visualization.

Deliverables

- **Implement the statistical sampling and analysis algorithms**. In 2018 we developed *density-based* sampling, in which a scalar field can be sampled such that the samples are placed probabilistically based on a user-defined density function of the scalar values. In Figure 1 we show an image containing samples a scalar field in samples are selected based on a density function that prefers the high values in the data set. This work led to the development of an adapted Markov Chain Monte Carlo algorithm based on on the Metropolis-Hastings algorithm.

  In 2019 we turned to developing sampling based on user-specified events occurring along rays cast through a data set. Figure 2 shows the use of this algorithm to generate sample points where sampling rays cross a specific value (which we call *isovalue sampling*). Figure 3 shows streamlines seeded by those samples.

  In 2020 we have returned to the idea of density-based sampling to address issues that arise in irregular datasets and datasets that change over time. This work, described in depth in Section 3.2, produced Figure 4. Here we have used several multi-variate density functions - of temperature, density and salinity, to produce samples of five *water masses* of interest to the ocean scientists.

  In addition to these sampling operations, we have implemented operations within the `andromeda` engine (See Section 3.5), to explore statistical sampling in the context of a partition-based engine.

- **Evaluate the effect of sampling operators on rendering, visualization and analysis**. Sampling operations on large scale data can have many impacts on rendering, visualization and analysis of the resulting data. See Section 3.2 for a discussion of this work.

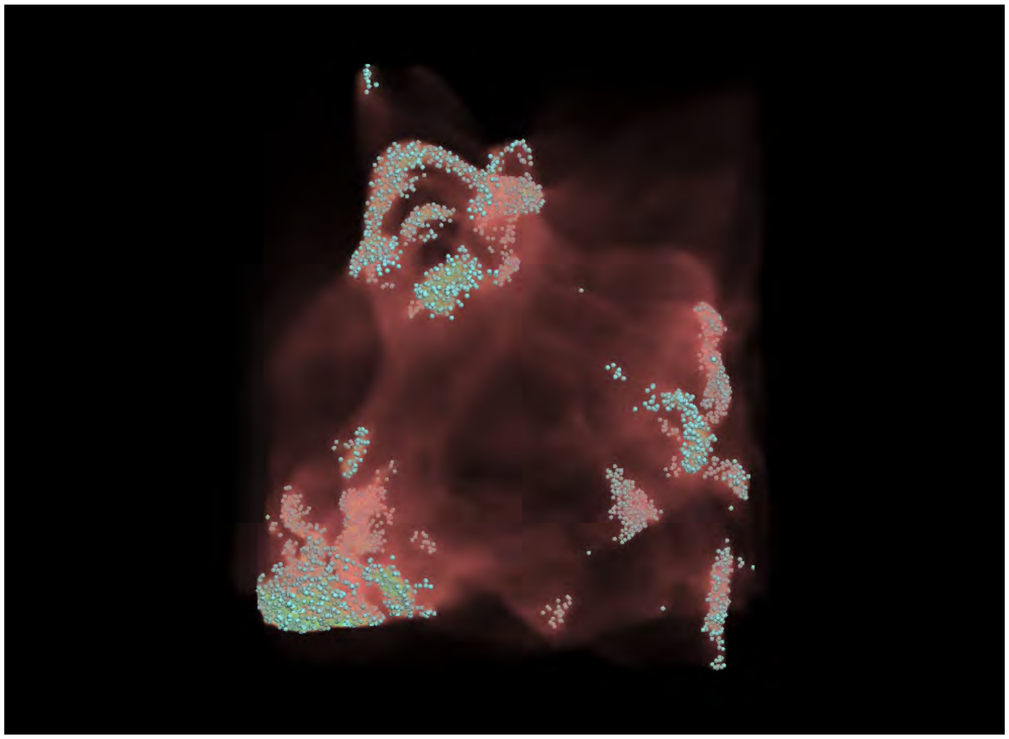


Figure 1: `galaxy` rendering of a density-based sampling of the Cosmo-Water dataset, in which samples are selected randomly based on a user-specified function of the temperature variable.

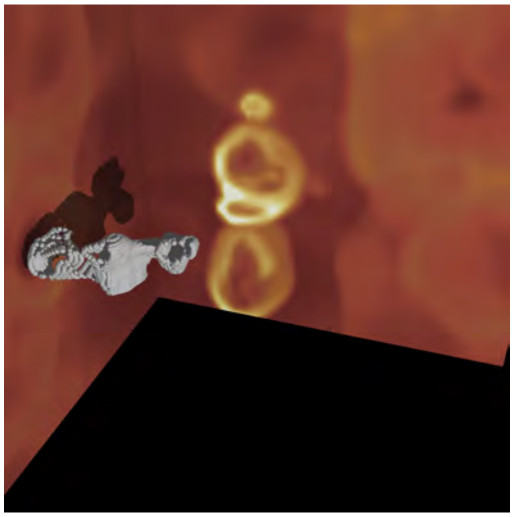


Figure 2: Isovalue-based samples of the Cosmo-Water dataset

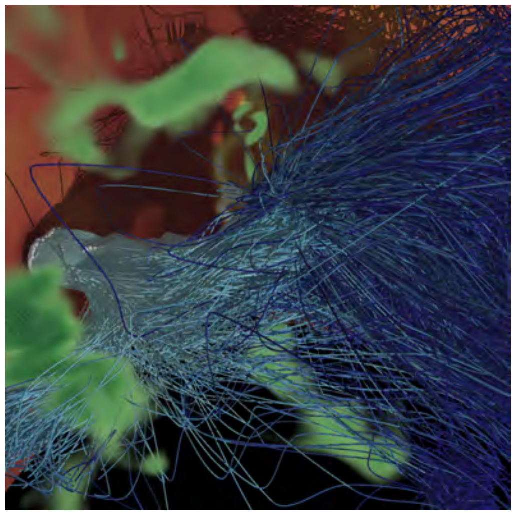


Figure 3: streamlines seeded by the sample shown in Figure 2

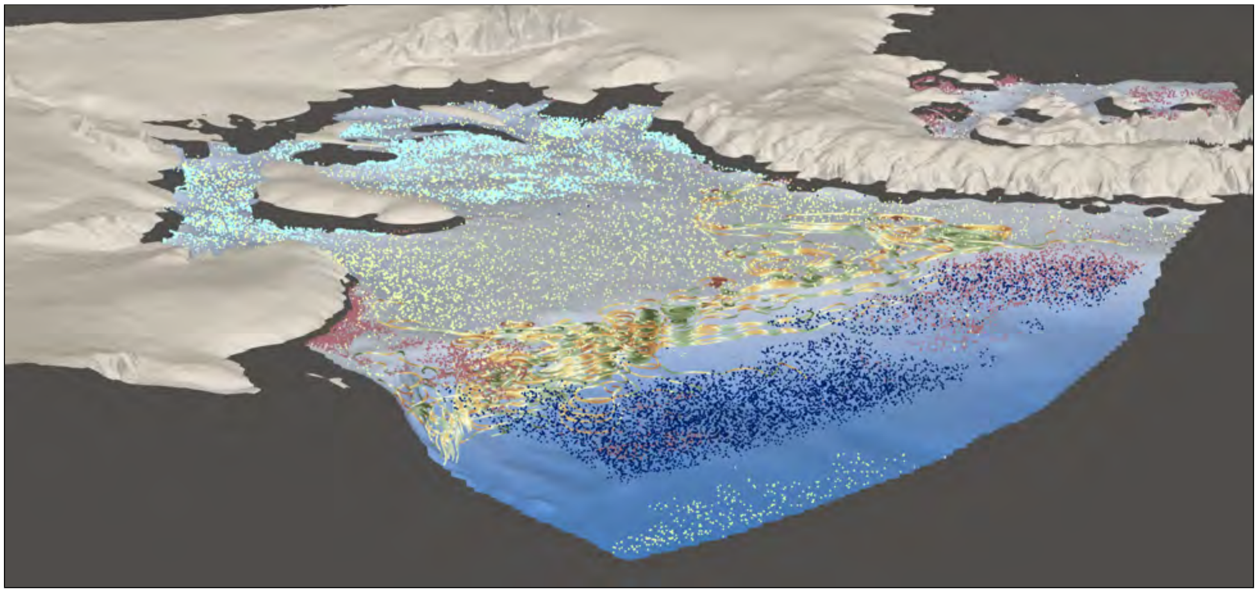


Figure 4: Density-based sampling the ocean under and near the Ronne-Filchner ice shelf in Antarctica (data from DOE E3SM). Here samples are used to help visualize complex water masses.

2.2 Perception and Cognition

Perception work this year concentrated on prototyping capability for the `galaxy` engine, and integrating color work into the engine, as well as a complete redesign of the `sciviscolor.org` web presence.

Deliverables

- **Deliver smart particle solutions (design and evaluation)**. For the operation of a sample-based engine, both rays and particles can be used to sample positions within the data. Particle position and behavior through time can be motivated by a number of algorithms, but a smart particle approach may be the best way to find informative positions within a dataset, and to use those as positions for evaluating sampling, visualization or analysis operations. We have implemented a smart particle solution, and evaluated it on a canonical scientific data analysis problem (see Section 3.3). This design and algorithm are now a candidate for inclusion in the Galaxy engine.
- **Deliver evaluated sets comprised of multiple colormaps and discrete colors for 3D data that provide perceptual differentiation and clarity of individual features**. Selecting colormaps to visualize multivariate datasets is a difficult and time consuming process. Our research provides a range of results that can help scientists select sets of colormaps that most effectively present the data (see Section 3.4).

3 Detailed Results

In this section, we describe results and deliverables for FY20, and other work called out in the proposal that have not been detailed in previous reports. References to relevant deliverables are included in deliverables section above.

3.1 Galaxy UI

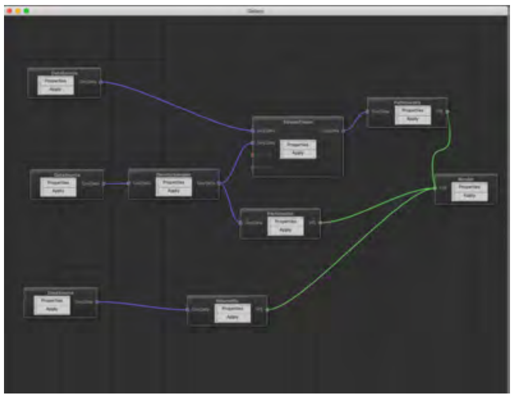


Figure 5: The `galaxy` GUI in use

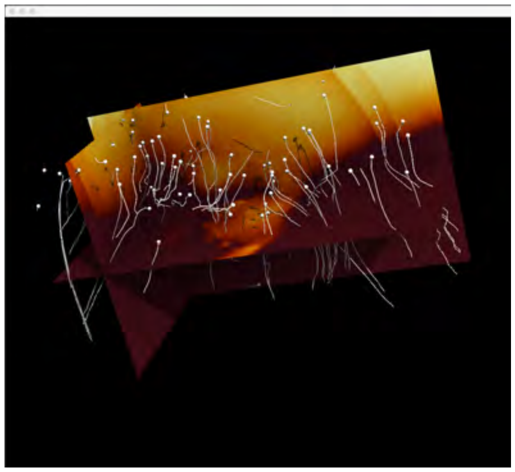


Figure 6: Image window generated by the `galaxy` network in figure 5

We have developed a graphical user interface for the `galaxy` system to make the capabilities of `galaxy` accessible to a wide range of users. Figures 5 and 6 shows screen captures of the `galaxy` GUI in action.

The `galaxy` GUI is built as a `galaxy` *multiserver client* and enables users to develop sophisticated visualizations as networks of *filters*. All actual data resides on the `galaxy` server; the `galaxy` GUI manipulates these via annotated *proxies* for the data objects, which carry limited information about the data (its spatial extent and data range, for example).

In general, filters accept proxies for operand data objects on the left and produce proxies for results on the right. Source filters, in particular the DataSource filter, do not accept input proxies; instead they provide an interface that enables the user to select among datasets resident on the server (and to import data, if necessary), and produce proxies for the selected data. Sink filters, in particular the Render filter) do not produce proxies; instead they represent an output of the visualization, either as an image to an interactive display window (as does the Render filter) or output to storage (as the to-be-implemented Export filter).

Some filters represent actual operations that will take place on the `galaxy` server. The `galaxy` GUI causes these operations to occur by transmitting an order (a JSON document containing operation-specific identifiers for operands and parameters) to the server; the server then carries out the operation, potentially creates server-side temporary datasets, and returns operation-specific results again as a JSON document. Such filters then create proxies for the newly created server-side data objects and produce them at right-hand output ports.

Other filters are solely client-side operators that receive proxies for datasets, annotate them based on parameters specified via their property page and produce them for input to subsequent filters. In particular, each renderable data type in `galaxy` (volumes, particle sets, triangle sets and path line sets) corresponds to a *Vis* filter in the GUI. The role of these is to annotate proxies for data objects with how they are to be rendered in the resulting visualization. Volume datasets, for example, can be rendered using volume rendering, requiring a transfer function, or using slices, requiring planar equations and/or isosurfaces, requiring isovalues. Since these are *implicit* operators carried out during the rendering process, no server-side action is required; instead, appropriately annotated proxies are sent to the Render filter.

As an example, consider the visualization in Figure 5. One path through the visualization uses the DataSource on the bottom left to interact with the `galaxy` server to access the temperature dataset, producing a proxy. This proxy is passed to the *VolumeVis* filter (center bottom) which creates an annotated proxy with a reference to the dataset plus two planar equations and a color map. This result is then passed to the Render filter, which will render two slices of the temperature dataset appropriately colored, along with any other Vis results it receives.

A slightly more complex path is immediately above this. A DataSource instance produces a proxy for the density dataset, and is is passed to a DensitySampler filter. Sampling is *not* done implicitly in the rendering step, so the DensityFilter causes the `galaxy` server to perform a sampling function on the dataset associated with the filter's input proxy and d property-page parameters), produce a temporary Particles dataset, and return the necessary information to the client side to enable the DensityFilter to create a proxy for this result. This proxy, for a Particles dataset, is then transferred to a ParticlesVis filter which adds annotations that inform the renderer how to size and color the particles in the resulting visualization.

Finally, the particles dataset is also used as seeds for tracing streamlines through the flow field selected by the top left DataSource. Both proxies for the particles (from the DensitySampler filter) and the flow field (from the DataSource filter) are attached to the StreamTracer filter. Again, streamlines cannot be done implicitly in the renderer, so the StreamTracer causes the server to advect streamlines, creating a temporary PathLines dataset, and return information to the StreamTracer filter enabling it to create a proxy for the path lines. This is then transmitted to a PathlinesVis filter that creates an proxy annotated with how to render the streamlines which, finally is passed to the Render filter.

The result, containing the two slices of temperature, the particles and the pathlines is shown in Figure 6.

The `galaxy` UI, by providing UI components that can be easily assembled into pipelines by users unfamiliar with the underlying mechanisms within the engine. Thus, it provides a way for many different types of users to take advantage of the technology developed in the *Stardust* project.

3.2 Sampling Evaluation

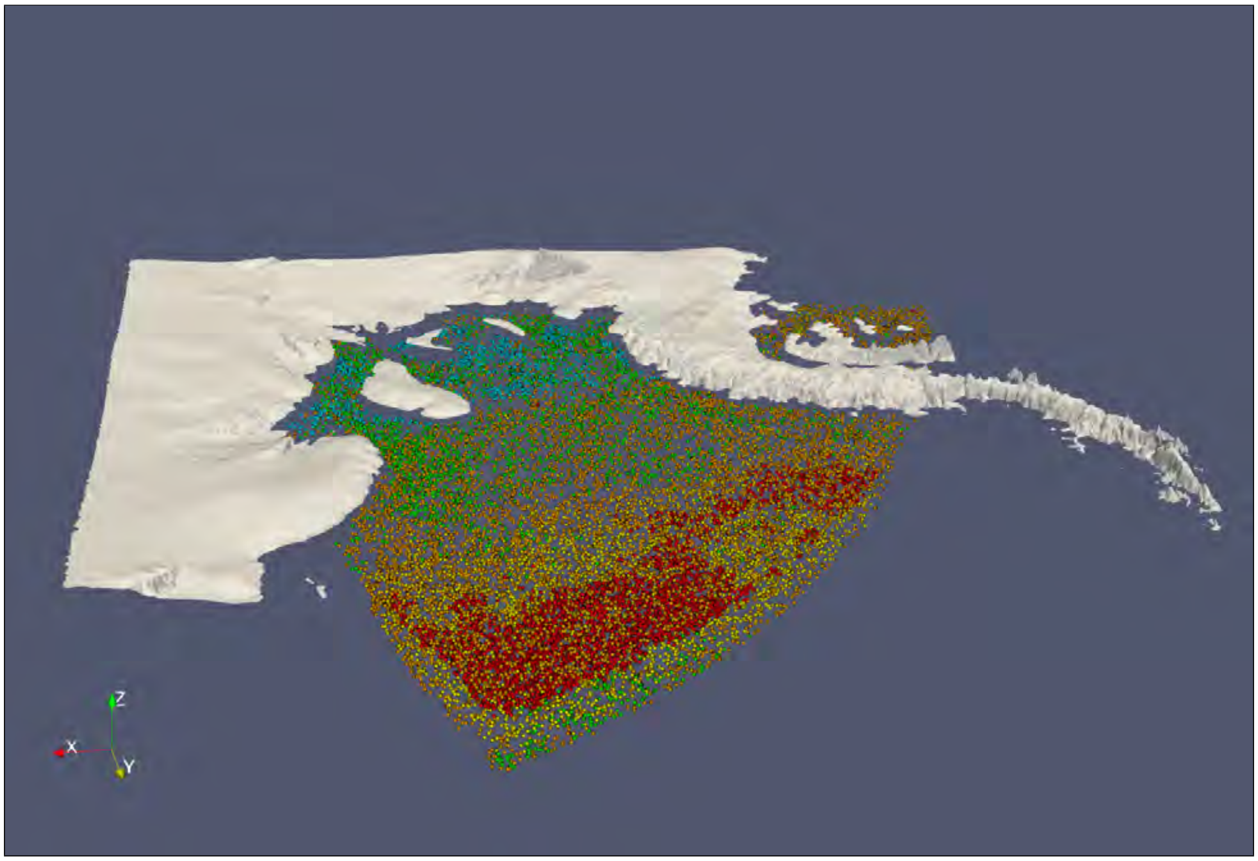


Figure 7: Density-sampled water masses.

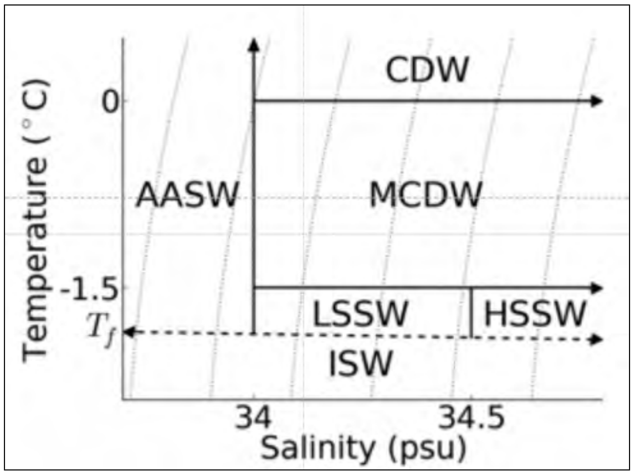


Figure 8: Water masses as a function of temperature, salinity and depth(T_f)

Recently we have investigated the use of density-based sampling to visualize modeled ocean data near and under the Ronne-Filchner ice sheet in Antarctica (Figure 4). In particular, our ocean science collaborators are interested in understanding the movement of six distinct *water masses*, which are as shown in Figure 8, from [10] . Density-based sampling is well suited to visualizing these water masses since because it avoids the obscuration and color-blending problems that arise when traditional opaque or translucent surface or volume rendering techniques are used.

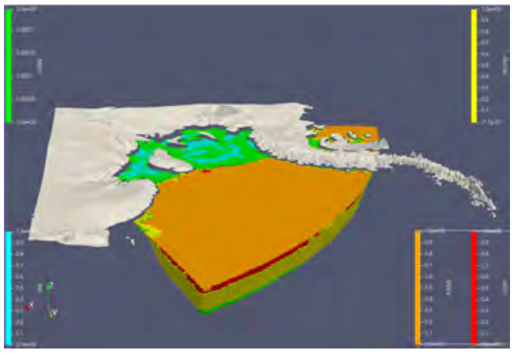


Figure 9: Ocean Model segmented among water masses.

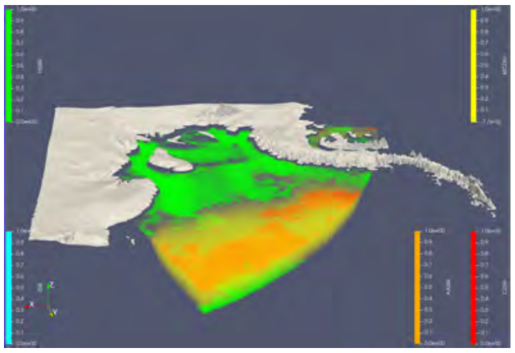


Figure 10: Water masses volume rendered.

Figures 9 and 3.2 show visualizations of this water mass data. Figure 9 shows the dataset segmented into the 5 regions and rendered as opaque surfaces. Obviously, here the issue is obscuration—the observer can't see the extent of water masses below the predominant AASW mass. Figure 3.2 uses volume rendering to address this issue. While this image shows the extent of some of the masses well, particularly the green HSSW and red CDW masses, the image does not show the orange AASW or cyan ISW masses at all, due to their shallow optical depth—even with a transfer function that maximizes the opacity of these masses, they are simply too thin to be visible. Further, the color blending that occurs in volume rendering embedded scalar fields is itself confusing. In Figure 3.2 the yellow MCDW mass changes the red CDW mass to an orange shade, potentially causing it to be confused with the AASW mass. We note that this issue may be alleviated by a more intelligent choice of colors.

In Figure 7 we have addressed this visualization by using our density-based sampling as a primary visualization technique. By distributing samples according to the water-mass density functions we can visualize the data as point clouds representing the water masses. By varying the density of the sampling and the size of the rendered spheres we can control the apparent opacity of the masses and, we believe, it is possible to see the interplay among the masses. Even though the orange AASW and cyan ISW masses are very thin, they are clearly visible.

Time-varying data introduces a particular challenge to the use of samples as a primary visualization technique. Since this sampling is probabilistic based on a field that varies even slightly over time, the sampling results in successive timesteps may vary significantly, producing sets of samples that—while reflecting the density distribution of each timestep correctly, are independent of one another, resulting in disturbing artifacts in animation over time. Instead, sampling over time calls for *coherence* between timesteps. The sampling of each timestep should inform the sampling of subsequent timesteps to minimize these artifacts.

Unfortunately, our prior work using a random-walk Markov Chain algorithm is poorly suited to this application, for several reasons. First, unlike the data used in that work (Figure 1), this ocean data is very irregular in shape—large in the latitude and longitude dimensions and very thin in the vertical dimension. This presents a problem for a three-dimensional Markov Chain algorithm which uses a normally-distributed stepsize in its walk. With a sufficiently large variance to produce steps that will explore the full extent of the dataset, very many of the steps will exit the computational space. Further, there is no obvious way to preserve coherence between successive timesteps. These issues led us to investigate alternatives to the Markov Chain algorithm.

3.2.1 Cell-Probability Sampling

We have investigated the use of the cells of the computational grid as a basis for random sampling. Our *cell-probability* uses the likelihood that a random sample will be chosen in any particular cell. The likelihood of a random sample of a density function f(X) = y, for $X \in R^n$ occurring in any particular partition of the domain $X_i \in X$ is $\int_{X_i} f / \int_X f$, in effect the area under the curve over the partition X_i divided by the area under the curve over all of X. Given the subdivision of the computational space into cells, we use the n-dimensional trapezoidal rule as an approximation for the integral, multiplying the volume of the each cell by the average of its vertex densities.

Once we have this piece-wise approximation to the probability density function, we can use it to generate samples. Our current method is to compute a running sum of the cell probabilities, resulting dividing [0,1] into a sequence of intervals, with the width if the i^{th} interval the probability that a sample will land in the i^{th} cell. We then iteratively select a cell for sampling by choosing a random number in the range [0,1], determine which interval it falls in and sample randomly inside the corresponding cell.

The quality of the results of this approach are tied to the division of the computational space into cells. If the cells are chosen so to capture the detail of the ground-truth scalar field to linear (or low-order) approximation, the results will be good. The trapezoidal approximation to the integral will be close and the random sampling within the cells will be approximate the relatively constant PDF over the cell. This is generally the case with direct simulation results. However, this may *not* be the case for derived scalar fields, such as the boolean water-mass fields in our example dataset. In particular, this results in artifacts where cells straddle the boundaries. While the number of samples in such cell may be (approximately) correct, sampling inside the cell without regard for the PDF *inside* the cell may make the cell boundaries visible.

We therefore are looking at smarter methods for sampling inside the cells, and two methods have been considered: a Markov Chain algorithm inside the cell, and further cell subdivision. However, we have chosen an alternative method for the binary water-mass visualization; we use an *iso-volume* algorithm to trim away the zero-valued parts of the domain and uniformly sample the remaining part.

Coherence between timesteps remains an issue with this approach. We have investigated two methods of "salting" this algorithm using samples from previous timesteps. We could, for example accept earlier prior samples in subsequent sets if the likelihood of their occurring in the subsequent step exceeds some threshold. Alternatively, if the above example determines to sample from a given set, we can instead choose a sample from the prior set if it is sufficiently close. Each was shown to result in unacceptable artifacts.

3.2.2 Poisson-Disk Sampling

Finally, we have investigated an algorithm based on Poisson-Disk sampling, a technique used to place samples randomly based on a separation rule: random samples are accepted if they are outside some minimum distance from nearby points, producing a natural-appearing distribution where there is no "crowding". We extend this algorithm by using the probability density function to vary the separation distance of samples. In effect, where the PDF is high, we use a small separation distance, resulting in close packing, but where the PDF is low, we enforce a high separation, resulting in widely separated sampling. We then select random points in the domain, interpolate a PDF value and use it to determine a necessary separation value at the point. We then search the pre-existing samples to see if any prior samples lie within this radius of the candidate, and if not, accept it.

In order to prioritize our efforts where samples are most likely to exist, we use an algorithm much like *cell-probability* sampling in our selection of candidate points. This prevents us from wasting time testing samples in large areas of low probability

This approach leads to a distribution of samples that is æsthetically appealing, and reveals the varying density of the underlying data and, importantly, provides a means to incorporate samples from prior timesteps: we simply begin the iteration with those prior samples. Since they will only be accepted if they meet the separation requirements of the subsequent timestep, samples from densely sampled regions of the prior time step will be preserved where possible and discarded elsewhere.

3.2.3 Further Work

Inspired by astonishing video of flocking behavior among starlings (we recommend searching YouTube for "Starling flocking"), we are beginning to investigate further use of sampling to represent time-varying density fields. This behavior has been simulated quite effectively in the computer graphics community. The *boids* algorithm [14] uses a simple set of rules to produce remarkable results. Reynolds' rules include separation - quite similar to our Poisson-Disk approach, and other work has added goal-oriented behaviors. To extend this work for density-based time-varying sampling of scalar functions we need to derive a vector field based on the difference between successive timesteps and then advect our samples accordingly, while obeying a separation rule. We think this will be a fruitful area of research resulting in a novel method of visualizing time–varying density fields.

3.3 Smart Particle Solutions

Animated pathlets are perhaps the best way of displaying patterns in 2D vector fields [25]. Pathlets are usually seeded displayed on a uniform pseudo random plane, from whence they are advected forward (in space and time) for a designated time span, they leave a trace of their passage that becomes transparent towards its tail. When a pathlet dies it is replaced by another at the original seed point.

Sometimes, we need to emphasize certain patterns in the data at the expense of others and so non uniform seeding is required. For example, in the case of the global ocean transport network (sometimes called the 'conveyor belt'), scientists are interested in flow pathways defining the major circulation patterns. To address the seeding problem, we developed a smart particle solution—the Ants and Fireflies algorithm—and we chose the long standing problem of visualizing ocean transport patterns from computed ocean flow models as an example of a visualization challenge. The solution is a general purpose sampling method applicable to revealing features in vector fields through animated visualizations. This design and implementation are now candidates for capabilities within `galaxy`.

The method has been implemented and evaluated in a collaboration with Mathew Maltrud, a modeler who is part of the Model for Prediction Across Scales [15] Ocean team (MPAS-Ocean). On Maltrud's recommendation the region chosen for investigation is the North Atlantic between the Equator and 70 deg N. Maltrud, provided data resampled from MPAS model output for a 50 year period beginning in 1930.

3.3.1 Ants and Fireflies Algorithm

The Ants and Fireflies algorithm incorporates a two stage process. First ants lay down scent trails according to certain criteria designed to bring out particular features in the data. Next fireflies, are attracted by the scent trails, live for a short interval and glow, revealing those features. To bring out transport pathways ant are seeded into the flow pattern and advected. An individual ant leaves a scent trace if it passes through more than one transect according to designated rules. The transects are situated so as to bring out particular flow pathways. An example of a set of Ant trails is given in Figure 11. Ant traces are color coded based on the transects that define them. Shown as blue in the background is the southerly flow pattern of mid water. The foreground shows different shallow water circulation patterns color coded to represent their defining transects. For example, the North Atlantic Gyre shown in orange.

Although the ant traces clearly reveal major flow pathways, they are generated anew for each timestep of the simulation and they show little frame to frame coherence. As a result, an animated time series based on ant traces is characterized by unpleasant flickering.

Fireflies provide the solution, giving visual continuity because they persist from frame-to-frame. Metaphorically speaking, fireflies are attracted by the scent trails left by ants. They descent into the vector field where they pick up the color of the scent and are swept along, leaving a colored trace. Fireflies have a lifespan, when they die a new one is born and like its predecessors it only lives if it lands near a scent trail. In the final animation only firefly traces are shown.

Figure 12 shows a still at year 30 of the simulation. The foreground traces reveal three distinct circulation patterns. The North Atlantic gyre characterized by clockwise rotation is

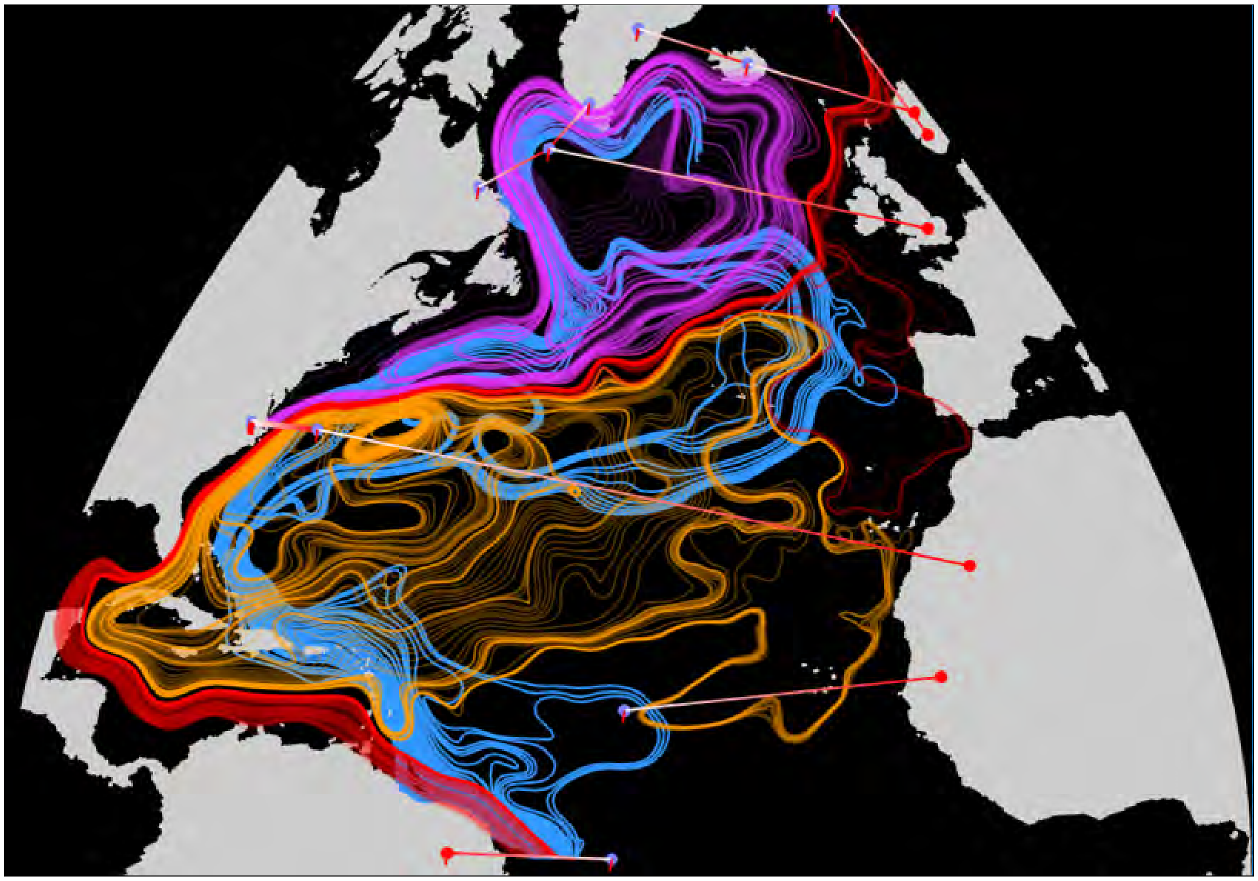


Figure 11: Traces defined by advected particles passing through two or more transects. Blue: mid water. Cerise, orange red: surface water

shown in orange. North of this is a smaller gyre rotation counter clockwise shown in cerise. To the south is a less stable pattern of currents mostly tracking northwards and shown in red. A very different midwater flow pattern is drawn in the background in dark blue: This is defined mean transport vectors between 1500 and 4000 m. and is characterized by southerly flow. Also shown in the visualization are graphs showing flow volumes associated with particular transects.

This algorithm is one example of a smart particle approach to finding areas of interest within complex data, and intelligently exploring data through visualization. Such algorithms clearly show promise for use on large data.

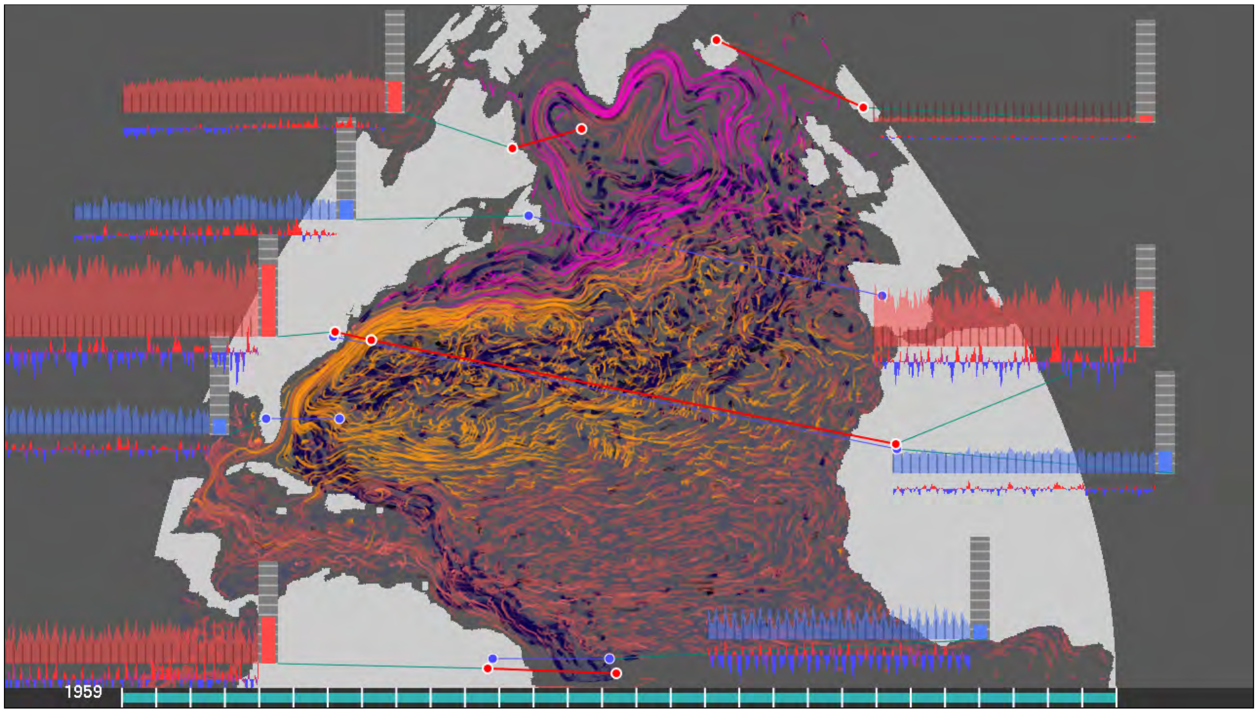


Figure 12: A single frame of a 50 year animation illustrating the North Atlantic transport pattern

3.4 Evaluated Colormap Sets

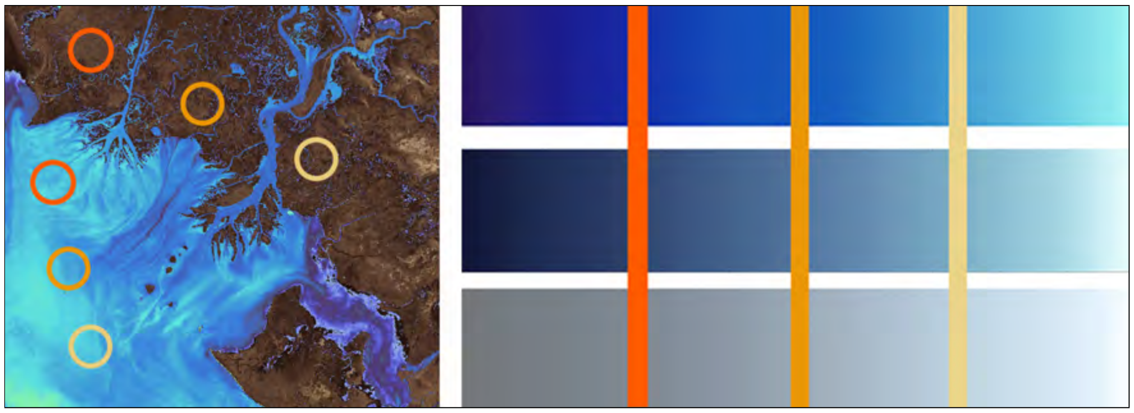


Figure 13: Blue - low, med and high saturation. Complimentary orange and yellows, also vary in saturation. All six components are designed to be distinct, interchangeable and provide the ability to direct attention.

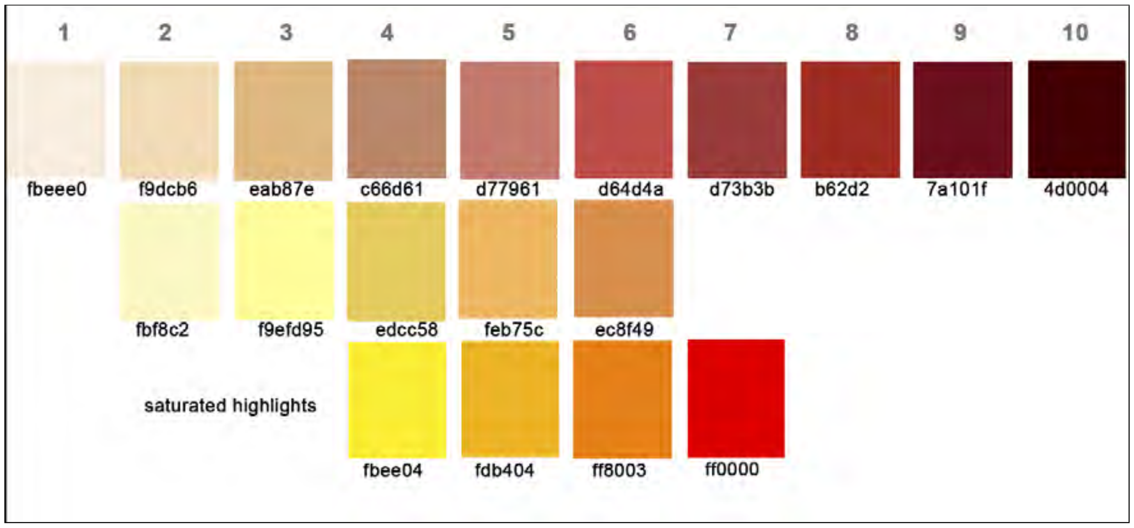


Figure 14: Warm discrete hues create a palette from which we can build color sets that include all seven types of color contrast.

To enable scientists to quickly and easily apply hues and colormap combinations to their data, we have developed a series of color sets.

Colormap Sets provide guidance to scientists when selection color encoding for multivariate data. Selecting colormaps for multivariate visualization is a difficult and time consuming task because color combinations, driven by the data, are often cacophonous and uneven, misdirecting attention and causing distraction and eye fatigue.

A color set is a combination of discrete hues and continuous colormaps from which scientists can select to create a harmonious and effective visualization, a simple version shown in Figure 13. Figure 14 illustrates the range of contrasts that are used to build color sets comprised of easily discernible hues. The hues and contrast types within a color sets such as shown in Figure 15 are designed to enable scientists indicate areas of importance and to avoid distracting color interactions.

The hue selections within each colormap as well as within each color set, are driven by traditional artistic color theory principles—the seven types of contrast. Figure 16 uses cool warm contrast to distinguish between the scalar fields and discrete hues, contrast of saturation to distinguish between the color scales and contrast of analogous hues to identify discrete variables. By controlling the level and type of contrast, we are able to create color sets that are not only perceptually distinct, but also direct attention to indicate the level of importance for specific regions of data.

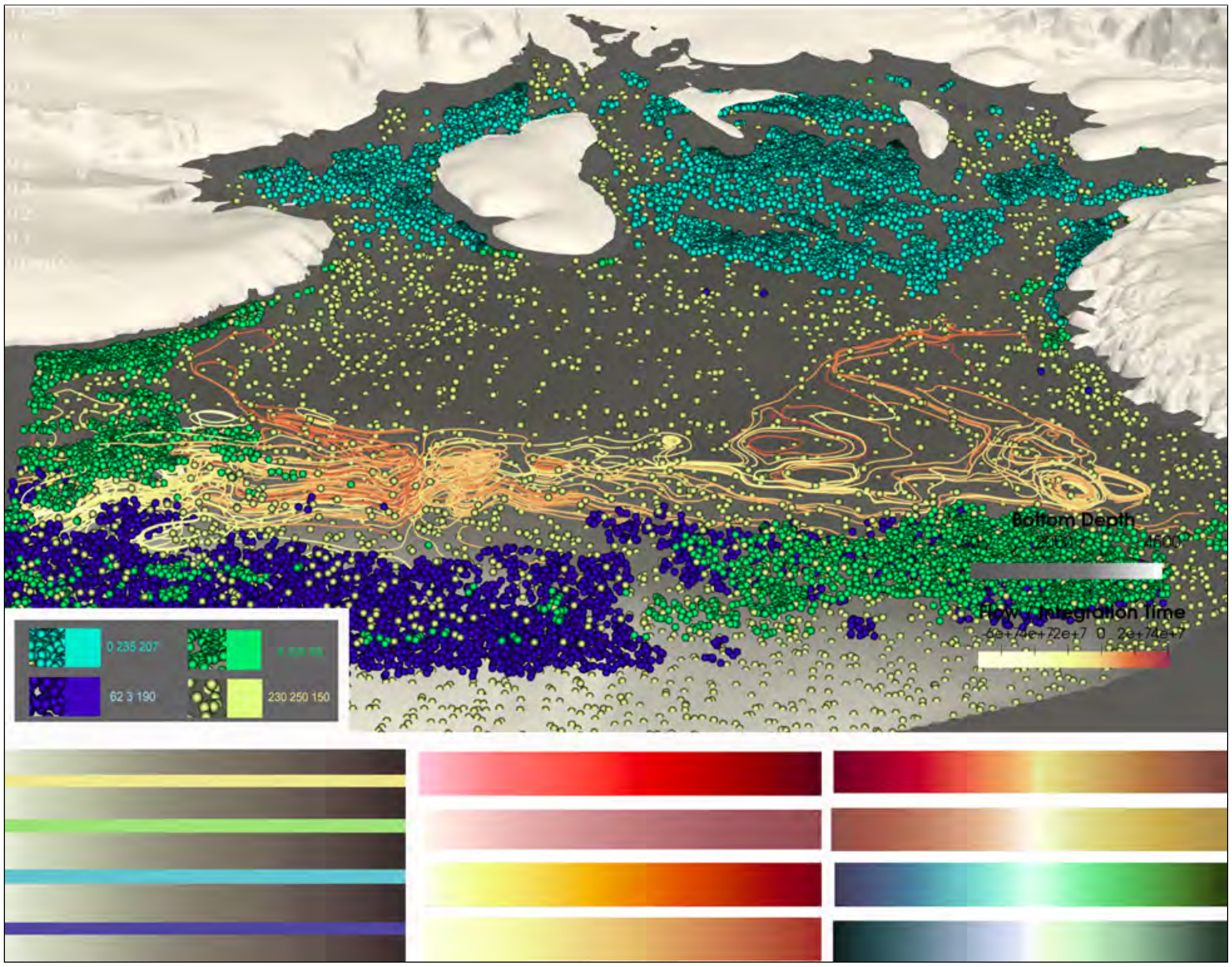


Figure 15: Here, volumetric data defining the water masses in the Southern Ocean has been sampled using a technique developed for Galaxy, and rendered in a color set developed here. Below, left to right: discrete hues on a colormap; warm colormaps, two sets of saturated and unsaturated; four divergent colormaps, two warm and two cool. The visualization above uses the discrete hues to identify four water masses, a warm colormap to contrast with the cool water masses to represent the flow and a muted brown colormap to render the ocean floor. All colormaps are drawn from the colormap set using the priority recommendations.

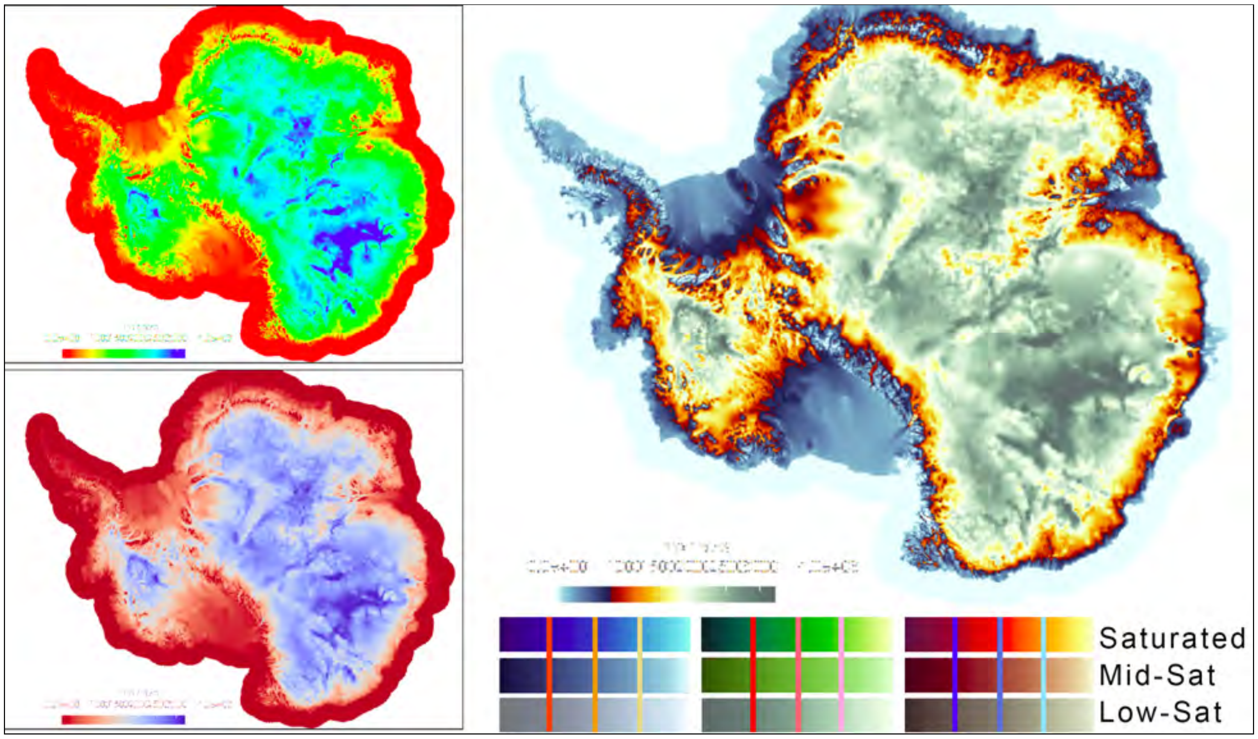


Figure 16: This figure demonstrates how saturation and hue can direct attention to appropriate points in the data set. Here, the coastline of the Antarctic is the area of greatest interest, so it is highlighted using a saturated red colormap. The surrounding ocean floor is an area of secondary interest, therefore we used a mid-saturation blue to highlight this region without overpowering the red. Finally, the interior of Antarctica is rendered in a muted green in order to provide context without losing detail or interfering with the other hues.

Ocean scientists are interested in the distribution and movements of six water masses in the Antarctic ocean, defined by specific ranges of temperature and salinity. Showing these concurrently is difficult using traditional visualization methods; opaque boundary surfaces obscure one another while translucent methods (both surface and volumetric) result in confusing color combinations. In Figure 15, we have sampled the water masses to produce a set of well-distributed points in each water mass, then to render these as small spheres. The water masses are clearly distinguishable simultaneously, over time, without ambiguous color mixing. Thus the relative motion of the different masses is apparent.

Figure 14 illustrates the building blocks that are the foundation of the color sets. Here, warm hues are divided into three levels of saturation and luminance while alternating between cool and warm versions of the hue. With this range of discrete hues we are able to build colorsets that take advantage of the discriminatory power provided by multiple types of contrast simultaneously.

The work on sets has lead us to a new series of work incorporating the color theory from sets with the scientists requests for more in-depth guidance, a section added to the website. Here we are in the process of testing *wave colormaps*, colormaps that move through multiple luminance ranges in order to increase feature resolution and enable isolation and focus on regions of interest. Figure 17 shows comparisons explored in the development stages of this work to begin to identify patterns of hue and saturation levels that increase clarity on specific types of datasets.

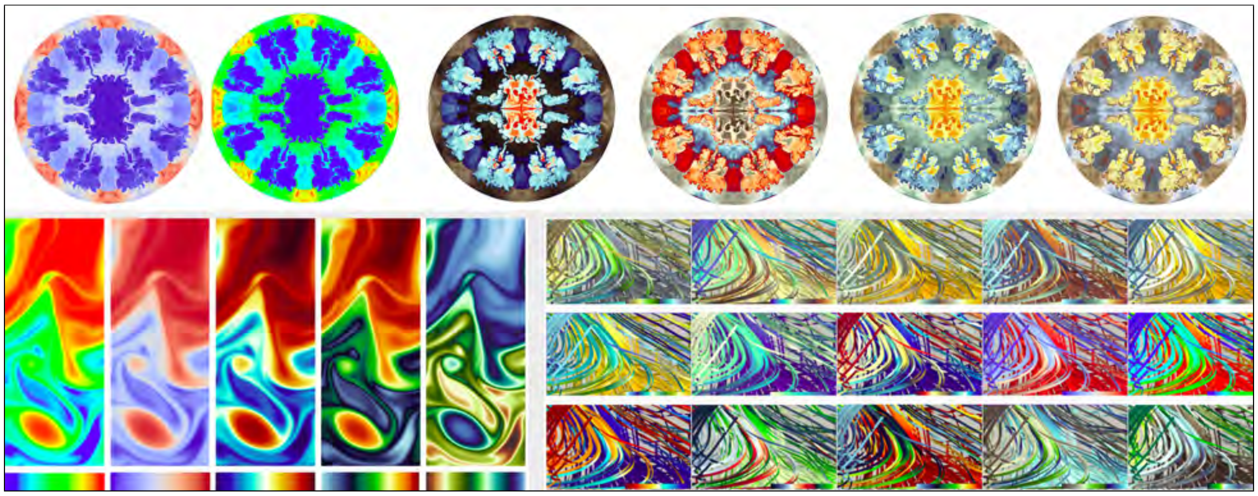
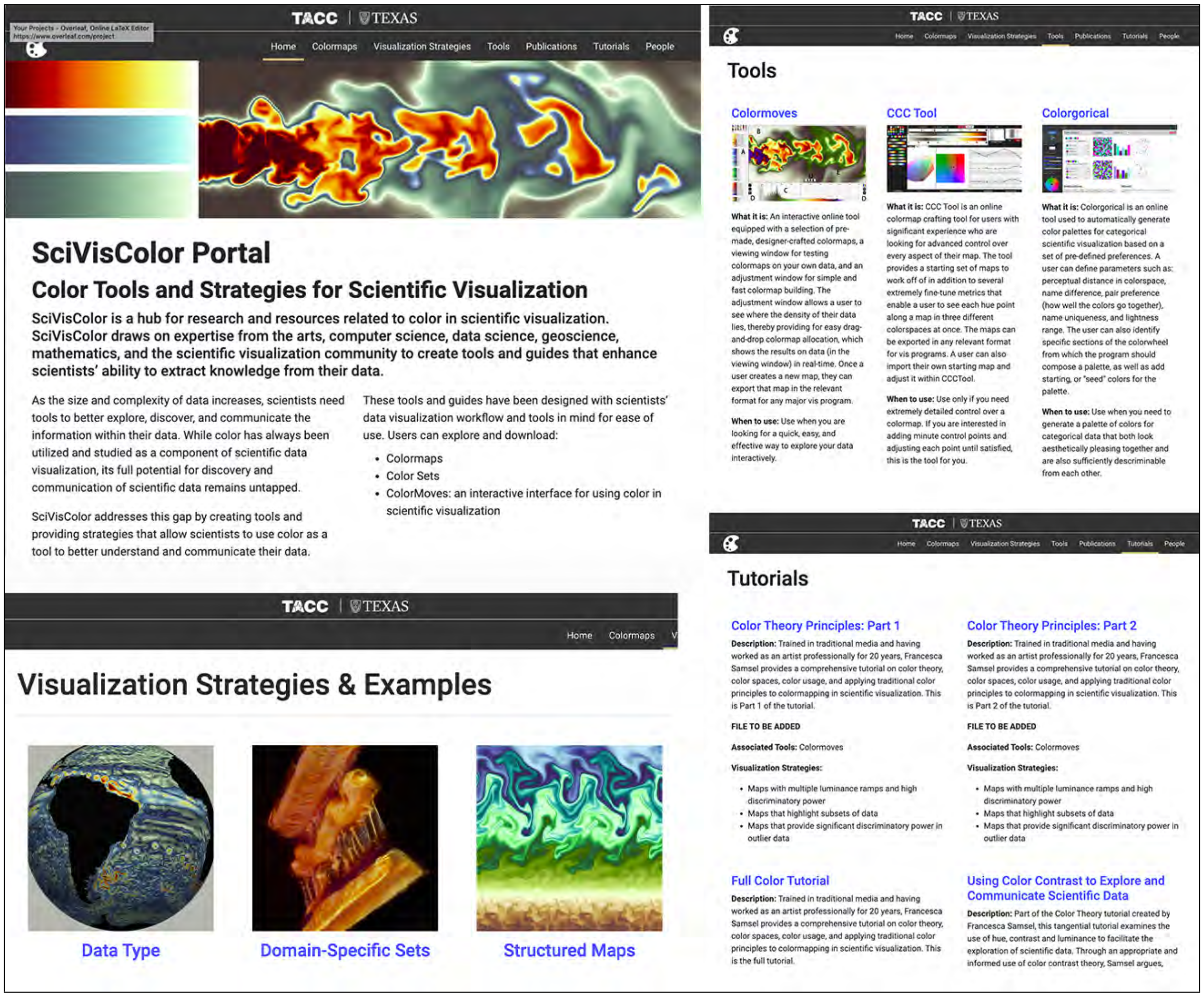


Figure 17: By comparing the results of wave maps on a range of data types we are able to hone in on examples of successful applications which are demonstrated in our Visualization Examples section of the updated `sciviscolor.org` .



SciVisColor Portal

Color Tools and Strategies for Scientific Visualization

SciVisColor is a hub for research and resources related to color in scientific visualization. SciVisColor draws on expertise from the arts, computer science, data science, geoscience, mathematics, and the scientific visualization community to create tools and guides that enhance scientists' ability to extract knowledge from their data.

As the size and complexity of data increases, scientists need tools to better explore, discover, and communicate the information within their data. While color has always been utilized and studied as a component of scientific data visualization, its full potential for discovery and communication of scientific data remains untapped.

SciVisColor addresses this gap by creating tools and providing strategies that allow scientists to use color as a tool to better understand and communicate their data.

These tools and guides have been designed with scientists' data visualization workflow and tools in mind for ease of use. Users can explore and download:

- Colormaps
- Color Sets
- ColorMoves: an interactive interface for using color in scientific visualization

Visualization Strategies & Examples

Tools

Colormoves

What it is: An interactive online tool equipped with a selection of pre-made, designer-crafted colormaps, a viewing window for testing colormaps on your own data, and an adjustment window for simple and fast colormap building. The adjustment window allows a user to see where the density of their data lies, thereby providing for easy drag-and-drop colormap allocation, which shows the results on data (in the viewing window) in real-time. Once a user creates a new map, they can export that map in the relevant format for any major vis program.

When to use: Use when you are looking for a quick, easy, and effective way to explore your data interactively.

CCC Tool

What it is: CCC Tool is an online colormap crafting tool for users with significant experience who are looking for advanced control over every aspect of their map. The tool provides a starting set of maps to work off of in addition to several extremely fine-tune metrics that enable a user to see each hue point along a map in three different colorspaces at once. The maps can be exported in any relevant format for vis programs. A user can also import their own starting map and adjust it within CCCTool.

When to use: Use only if you need extremely detailed control over a colormap. If you are interested in adding minute control points and adjusting each point until satisfied, this is the tool for you.

Colorgorical

What it is: Colorgorical is an online tool used to automatically generate color palettes for categorical scientific visualization based on a set of pre-defined preferences. A user can define parameters such as: perceptual distance in colorspace, name difference, pair preference (how well the colors go together), name uniqueness, and lightness range. The user can also identify specific sections of the colorwheel from which the program should compose a palette, as well as add starting, or "seed" colors for the palette.

When to use: Use when you need to generate a palette of colors for categorical data that both look aesthetically pleasing together and are also sufficiently descriminable from each other.

Tutorials

Color Theory Principles: Part 1

Description: Trained in traditional media and having worked as an artist professionally for 20 years, Francesca Samsel provides a comprehensive tutorial on color theory, color spaces, color usage, and applying traditional color principles to colormapping in scientific visualization. This is Part 1 of the tutorial.

FILE TO BE ADDED

Associated Tools: Colormoves

Visualization Strategies:

- Maps with multiple luminance ramps and high discriminatory power
- Maps that highlight subsets of data
- Maps that provide significant discriminatory power in outlier data

Color Theory Principles: Part 2

Description: Trained in traditional media and having worked as an artist professionally for 20 years, Francesca Samsel provides a comprehensive tutorial on color theory, color spaces, color usage, and applying traditional color principles to colormapping in scientific visualization. This is Part 2 of the tutorial.

FILE TO BE ADDED

Associated Tools: Colormoves

Visualization Strategies:

- Maps with multiple luminance ramps and high discriminatory power
- Maps that highlight subsets of data
- Maps that provide significant discriminatory power in outlier data

Full Color Tutorial

Description: Trained in traditional media and having worked as an artist professionally for 20 years, Francesca Samsel provides a comprehensive tutorial on color theory, color spaces, color usage, and applying traditional color principles to colormapping in scientific visualization. This is the full tutorial.

Using Color Contrast to Explore and Communicate Scientific Data

Description: Part of the Color Theory tutorial created by Francesca Samsel, this tangential tutorial examines the use of hue, contrast and luminance to facilitate the exploration of scientific data. Through an appropriate and informed use of color contrast theory, Samsel argues,

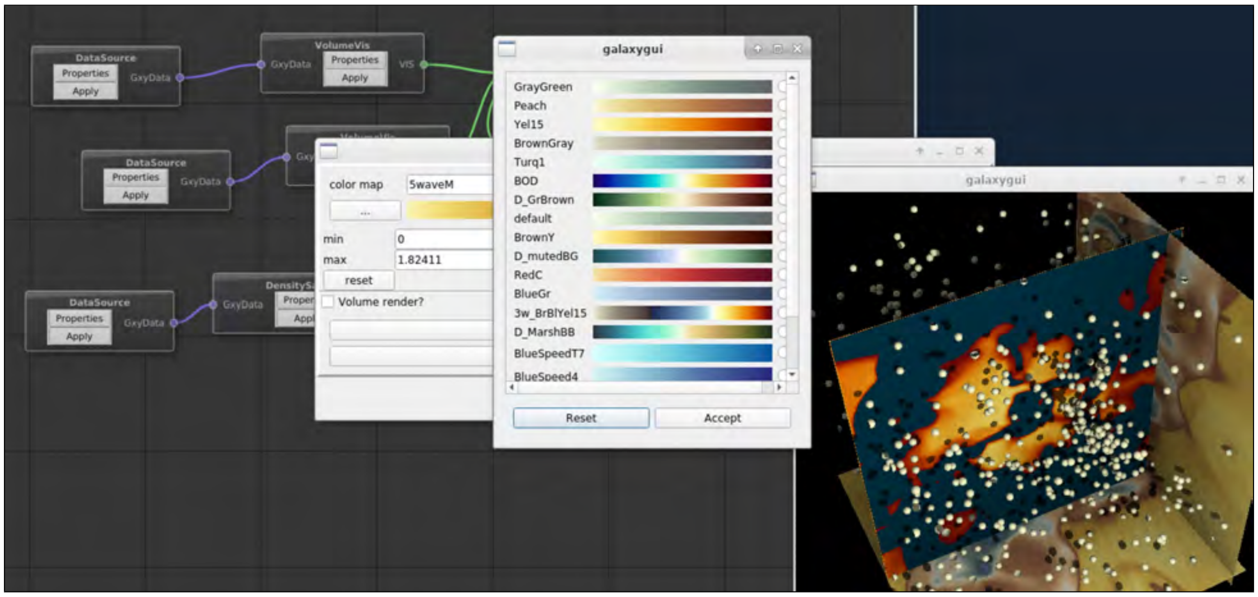
Figure 18: SciVisColor.org has been expanded to include several new sections: an extensive Visualization Strategies and Examples section; Tools which includes other color resources; categorical tutorials; and video tutorials.

3.4.1 SciVisColor.org

`sciviscolor.org` is the online home for research, resources and applications for color tools and strategies for scientific visualization. Most of the work funded by this project is available at this site. In addition, the most frequently used colormaps from `sciviscolor.org` have been installed in `galaxy` (See Figure 19). The site had over 5700 users last year. To ensure its growth and continuity the SciVisColor website is undergoing a complete update, as shown in Figure 18 and being transferred to a permanent, TACC-hosted server. In order to better serve our audience of scientists and visualization professionals, we created a task-based system for organizing and presenting content to users. This content includes a list of tools and applications from across disciplines, categorized by usage and linked to their relevant publications [16, 30, 9], tutorials [17, 18] ranging from color theory principles in visualization to tool usage; color sets with usage recommendations; the full colormap database with file downloads available; and examples of color map and color theory applications, organized as "visualization strategies" [30, 9, 1]. Visualization Strategies allow a user to search for an appropriate colormapping methodology by the type of data they are visualizing, the domain of their data, specific types of pre-structured colormaps, or artistically-inspired colormaps [19]. The user will find specific recommendations, along with examples of suggested colormaps on data, relevant tutorials, and relevant tools. This reorganization and update will provide more complete and accessible resources to our constituents.

3.4.2 Colormap Database

The colormap database coalesces years of colorwork across the team. Amalgamating all linear, divergent, convergent, and "wave" (multiple luminance-ramped) colormaps, the colormap database categorizes each individual map by hue, characteristic, use, and semantic association, as well as providing values for the upper and lower luminance bounds of the map. This database with form the foundation of upcoming work on automating the creation of colormaps based on scientist needs and data density.



color map 5waveM
min 0
max 1.82411
Volume render?

Figure 19: The most frequently used colormaps from have been installed for use in Galaxy.

3.5 Andromeda Prototype Engine

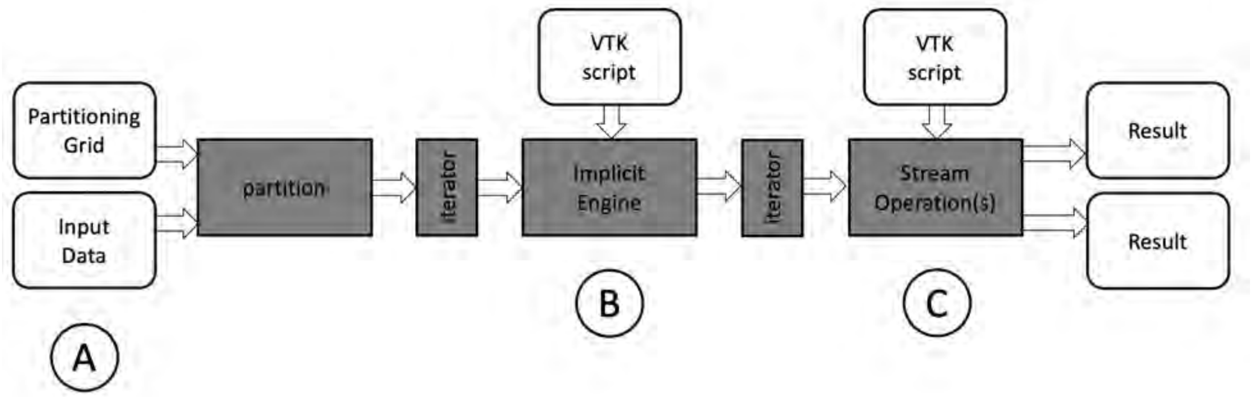


Figure 20: A diagram of the `andromeda` pipeline. A partitioning grid is used to partition the input data into a set of data partitions (A). This set of partitions is then evaluated by an implicit engine (B) that filters out data that will not need to be evaluated by an input VTK script. The results of that filter operation are then streamed to the VTK script (C) to produce the correct output.

The `andromeda` engine is a prototype for analysis and visualization of massive data that is based on evaluation of data partitions through the use of implicit functions, based on operations requested by the user (through a script). The Python-based prototype is designed with data streaming in mind, so that each data partition can be evaluated independently. Processes can add or delete partitions, add metadata to the partitions, evaluate algorithms on the partitions, and render them to some final form. An overview of the prototype is shown in Figure 20.

The prototype takes advantage of existing capabilities in VTK to evaluate data partitions and determine if they contain data that must be processed, based on the operations in an arbitrary VTK pipeline. This approach has the advantage of utilizing the existing capabilities of the VTK framework, while still providing a sampling-based approach (through partitioning of the dataset), that is not currently implemented within that framework. Thus, the prototype provides an additional method for analysis and visualization of extreme-scale data. Key to this approach is an input partitioning grid, which can be computed by a number of statistical or sample-based approaches, and is independent of the subsequent processing of the data as demonstrated by `andromeda`. In fact, the data can be constantly re-partitioned (and the partitioning refined) at any point along the pipeline.

As shown in Figure 20, the `andromeda` engine takes as input a partitioning grid and a dataset, producing a set of partitions of the data. These partitions are then operated on by streaming the partitions through a set of operations. The partitions are controlled by partition iterators, which provide a flexible way of processing the data. The partition iterators can be used to filter the partitions, but in general simply provide access to the members of a set of partitions. The Implicit Engine (See Section 3.5.1), takes as input a

VTK pipeline (or, more generally, a script) that defines a set of operations on the data. The engine determines which partitions must be evaluated in order for the operations to be performed, filtering out those partitions that do not need to be evaluated. In this way the streaming operations are optimized to operate only on data critical to the input script. The implicit engine thus produces a set of partitions that *must* be evaluated to produce a correct result, leaving out those that can be ignored.

An additional benefit of this approach is that operations can continuously add metadata to partitions, for use later in the pipeline. Importantly, in many cases, a partition can be processed at its own pace, independently of all others, allowing the overall processing of the data to proceed asynchronously. For the purposes of this prototype, we did not implement the asynchronous processing, but we believe the work here lays the foundation for a later asynchronous engine based on this work.

3.5.1 Implicit Engine and Operators

In VTK, implicit functions are real valued functions defined in 3D space, where they are used to evaluate the function at a certain point, usually on a surface of geometry. Previously only limited to vtk geometry filters, We extend this query-based functionality to general VTK filters, e.g. Threshold, Contour, Glyph, Cut, Clip, and Extract subset. By doing so, we are able to evaluate generic VTK filters by either a partition's value-range (range-based query) or by its spatial size and dimensions (spatial-based query). This process maps many VTK operations down to a very small set of easily evaluated operators.

The implicit engine uses this inherited information to decide the ordering and processing of these VTK filters at a per-partition basis.

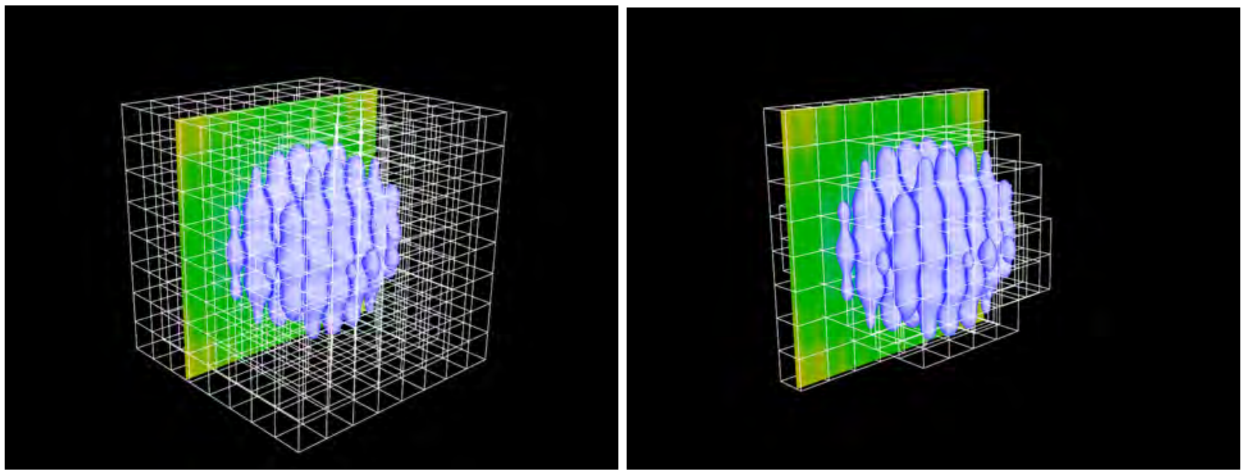


Figure 21: Implicit engine. Left: Full data. Right: VTK implicit operators only process partitions based on range-based information using the implicit engine.

3.5.2 Statistics operations

When datasets are partitioned, a set of standard statistics (mean, median, minimum, maximum, and standard deviation) are computed and stored as metadata in a JSON format for the partition, and can be accessed by iterators and processing filters later in the pipeline. This general execution method, of adding data during the streaming process, makes it easy to chain operations together, and supports the streaming evaluation of partitioned, independent data. These quantities can be easily computed as the data is streamed, utilizing common statistical operations found in Python modules. Such results can be saved as shown in this example:

```
{
    "variables": {
        "cell": {}'
        "point": {
            "RTData": {
                "mean": 83.343642028808594,
                "median": 84.11825561523438,
                "min": 44.801490783691406,
                "max": 107.83773803710938,
                "std": 11.768940925598145
            },
            "field": {}
        }
    },
    "ID": o,
    "vtkfile": "testing/scratch/data/data0000.vti"
}
```

Beyond the default set, any custom statistic can be computed as metadata to be passed through the implicit engine to evaluate partitions by a different metric.

The statistic operations, stored as metadata, can be used to perform partition filtering to enable a more focused workload, rather than operating on all partitions i.e. the entire dataset. An example of the mean operation can be see in Fig. 22.

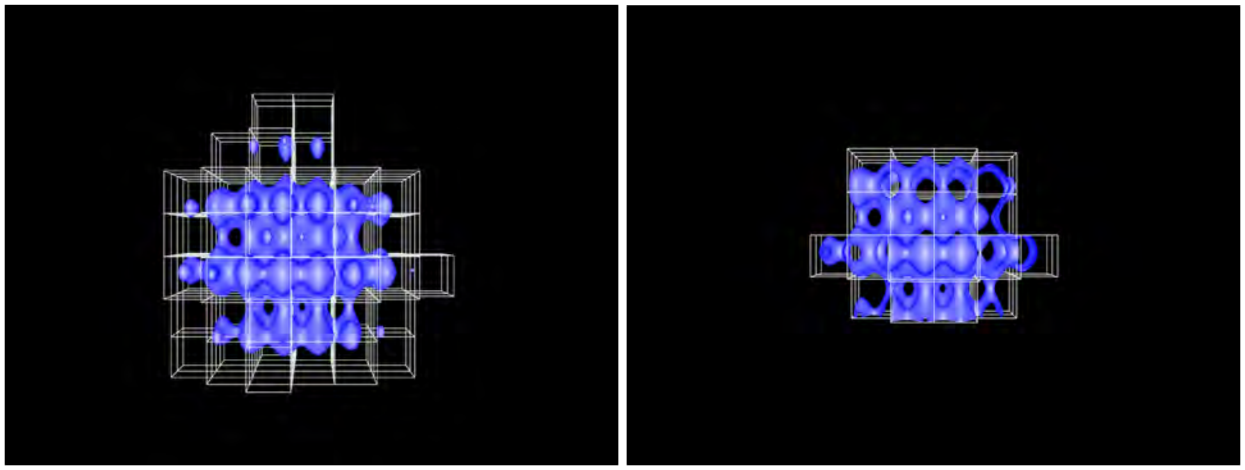


Figure 22: Filtering by statistics metadata. The left image is the original isosurface visualization of a wavelet. The right image is a set of partitions, filtered by the mean statistic. Only partitions with a mean larger than 210 are selected and processed by andromeda.

3.5.3 Sampling Operations

After data is partitioned, there is an option to perform different sampling schemes. Performing random sampling (seen in Fig. 23), stratified random (See Figure 24), or systemic sampling can be useful in situations where many partitions exists, or particle datasets are being used rather than regular grid.

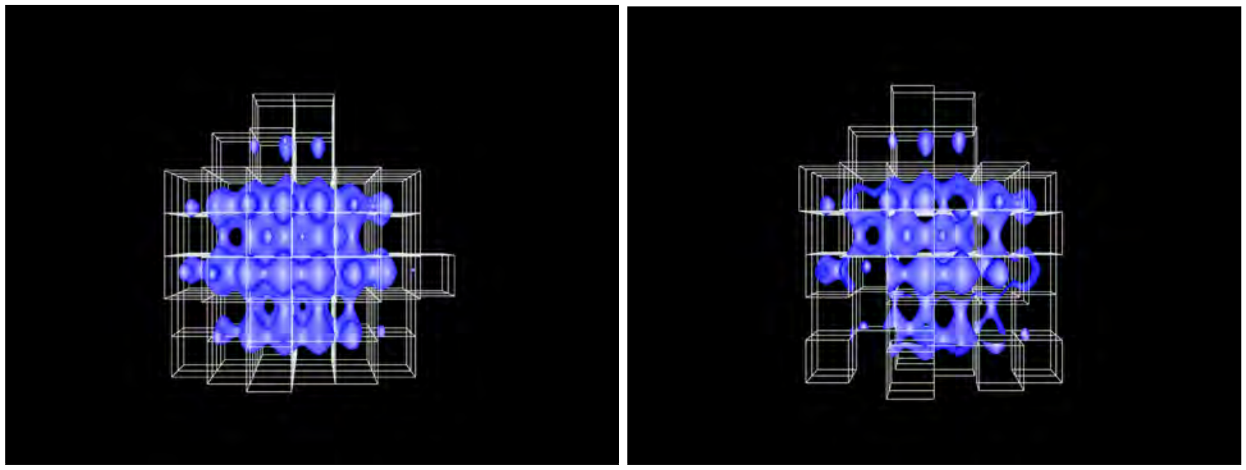


Figure 23: Example of the random sampling operator. The left image is the original isosurface visualization of a wavelet. The right image is a random sampled subset on the visible partitions.

3.5.4 Okubo-Weiss Analysis Algorithm

As part of this prototype, we implemented several example algorithms to demonstrate that the `andromeda` framework can usefully evaluated independently partitioned data. One such operation is the computation of the Okubo-Weiss [5, 23] metric, a value used to identify vortices in data.

The Okubo-Weiss criterion for identifying vortexes is $W < 0$, where W is defined as:

$$W = (s_n)^2 + (s_s)^2 - \omega^2 \tag{1}$$

$$s_n = \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}, s_s = \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \tag{2}$$

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}. \tag{3}$$

Here, s_n is the normal component of strain, s_s the shear component of strain, and ω is the vorticity.

When the Okubo-Weiss criterion is computed on a per-partition basis where velocity components are detected, a negative W quantity denotes a high likelihood vortex activity

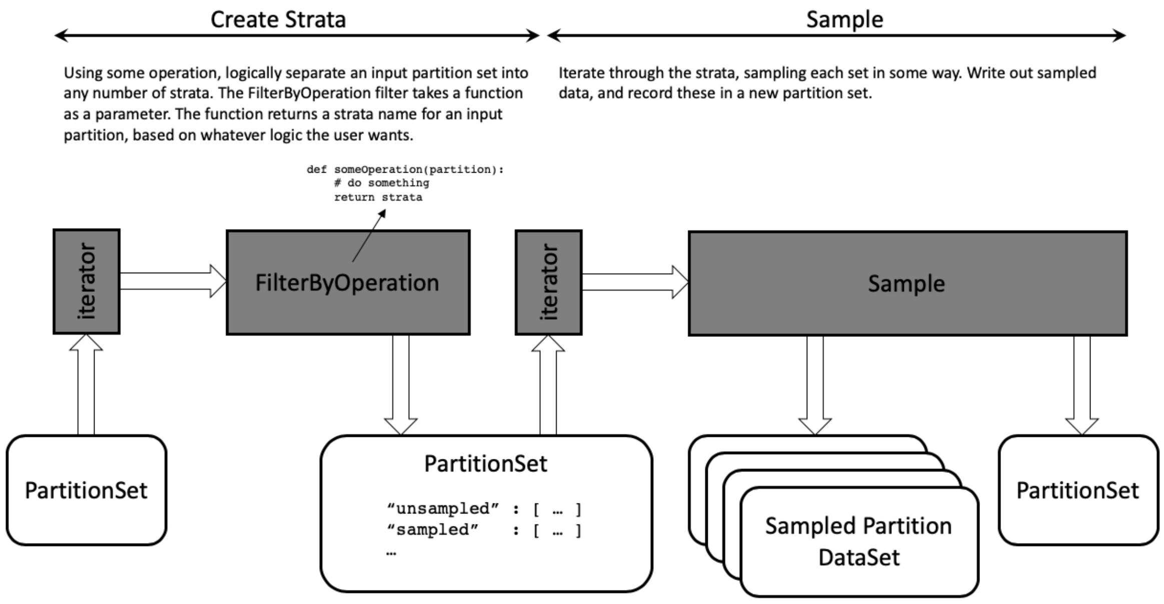

Create Strata
Sample
Using some operation, logically separate an input partition set into any number of strata. The FilterByOperation filter takes a function as a parameter. The function returns a strata name for an input partition, based on whatever logic the user wants.
Iterate through the strata, sampling each set in some way. Write out sampled data, and record these in a new partition set.
def someOperation(partition):
    # do something
    return strata
iterator
FilterByOperation
iterator
Sample
PartitionSet
PartitionSet
"unsampled" : […]
"sampled"   : […]
…
Sampled Partition DataSet
PartitionSet

Figure 24: A diagram of stratified random sampling operations within the `andromeda` pipeline. This is a two step process, in keeping with the streaming model. First, strata are created by computing some value on the partition, using a `FilerByOperation` filter. This creates tagged sets of partitions within a set of partitions. Second, are then accessed through partition iterators, and sampled, based on some criteria.

at that location. For example, by using a fixed negative threshold, we are able to filter all partitions to only enable computation in regions with high vortex activity as a feature of interest. This analysis operator is applied to an open-source, turbulence dataset from the John's Hopkins Turbulence Database [8], and shown in Fig. 25

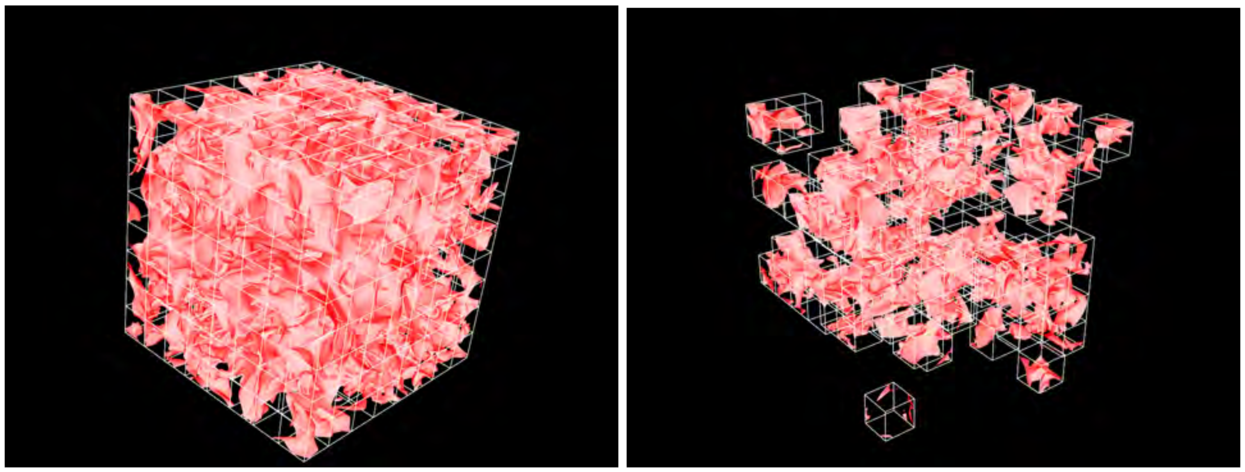


Figure 25: An Okubo-Weiss analysis operator. Left: Original turbulence dataset, rendered with an isosurface on the velocity-x scalar field. Right: only the regions with a negative W are filtered ($W < -20$), enabling the processing and visualization regions with high-vortex activity.

3.5.5 Halo Finder Analysis Algorithm

Another algorithm that can operate well within the `andromeda` workflow is a halo finder algorithm for cosmology data. The prototype halo finder computes a density statistic for each partition, then based on a threshold the partition will be filtered before running an actual halo finder. The halo finder is a histogram-based, density finder. Here we show the results of the algorithm running on Nyx cosmology simulation dataset, dark matter density [3], Figure 26

3.5.6 Results

The `andromeda` prototype demonstrates that a partition-based approach can optimize the data being evaluated in extreme scale datasets, and can also use VTK to analyze data, create data artifacts and produce visualizations. This hybrid approach may therefore serve as a useful option for asynchronous, massive data analysis.

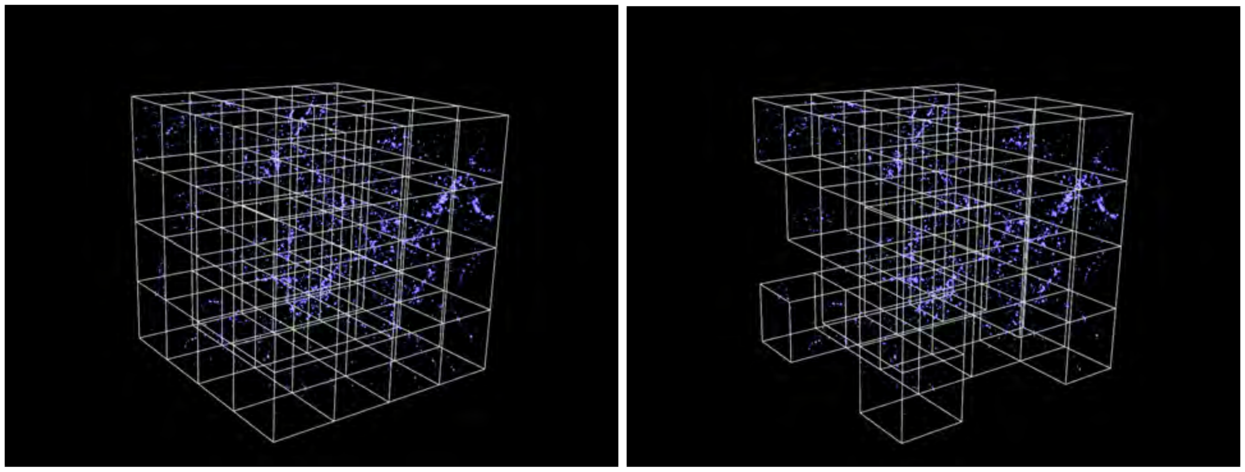


Figure 26: A density, histogram-based halo finder is used to pre-filter regions that may contain halo activity. Left: Original cosmology dataset rendering the Dark Matter Density regions. Right: A histogram-based, density finder that pre-filters regions with high density activity before finding halo regions.

3.6 Sampling and visualization of time-varying features

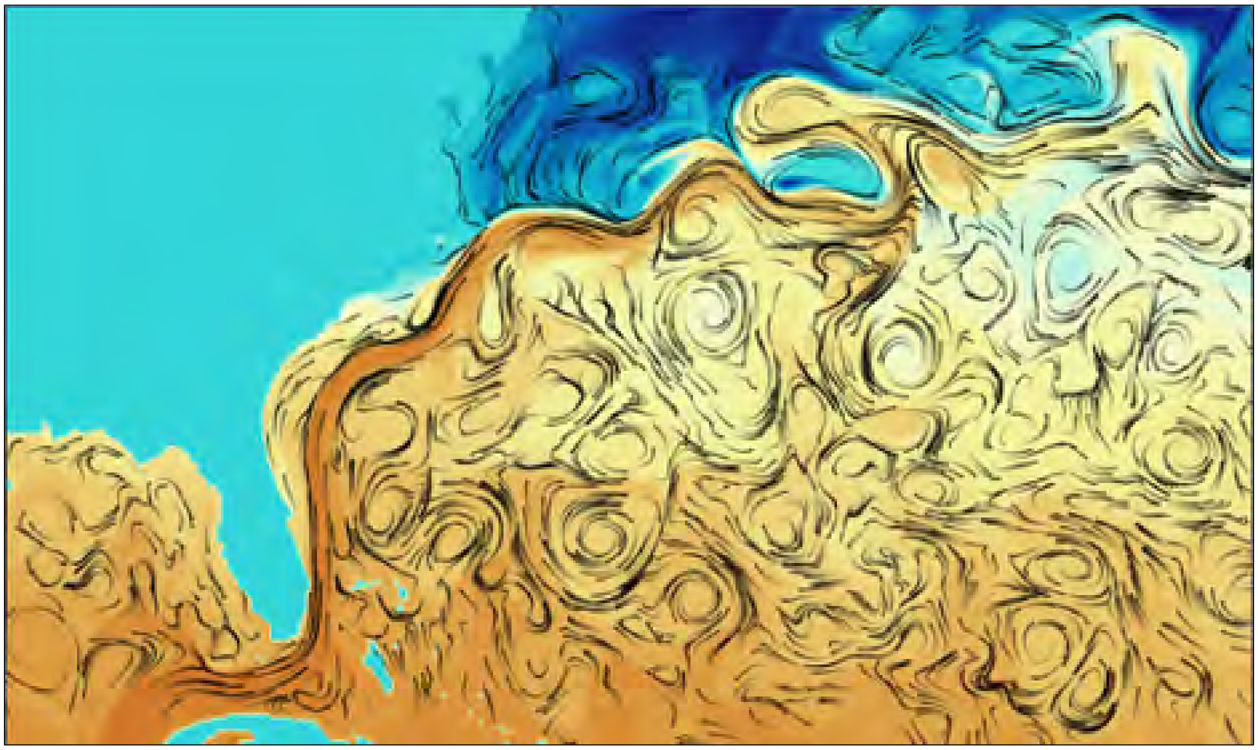


Figure 27: A frame from an animation of the Gulf Stream. Animated streamlets reveal the flow pattern. Colored background shows sea surface temperature.

Current practice is to save large scale computational models at every kth time step of the simulation. This is done because it enables re-starts should the system crash. These saved files are also used for visualization of model evolution but they have properties making them less desirable for this purpose. Because saves are done relatively infrequently this means that they can be unsuitable for visualizing the evolution of spatio-temporal structures, such as eddies and jets. They are also structured in such a way that makes extraction of visualization products relatively slow and this can have a major impact on cognition.

We investigated the proposition that given limited data budgets, simple sampling schemes together with basic compression can support visualizations much more likely to yield insights relating to ocean flow dynamics than saving every kth time slice. Specifically: dynamic aspects of eddy genesis will be visible with the new scheme, but they will not be visible with the kth time slice method.

Three sampling schemes were implemented. (1) Full resolution, every time slice (for comparison). 2) Full frame samples every 27 days. 3) Every 3rd sample in x, y and t. (A 27 times data reduction). Note that methods 2 and 3 require the same amount of data. Animations were produced based on data from each of the sampling methods. In the case of the 27 day samples simple linear interpolation was done between the key frames. In the case of the spatio-temporal sampling method three-dimensional linear interpolation was done.

The result show that the spatio-temporal sampling method clearly reveals the process whereby eddies are spun off the gulf stream; whereas this cannot be seen when the (interpolated) full frames are used. The results of spatio temporal sampling are virtually indistinguishable from the full resolution data, but are 27x times smaller.

In other work we have shown that simple compression schemes can further reduce data volumes with little or no loss in visualization quality. By combining spatio-temporal sampling with data compression it should be possible to produce much better animated visualizations with relatively small amounts of data and suggest that in the future, high performance computing simulations should save data images designed specifically designed for visualization.

3.7 Industry Collaboration

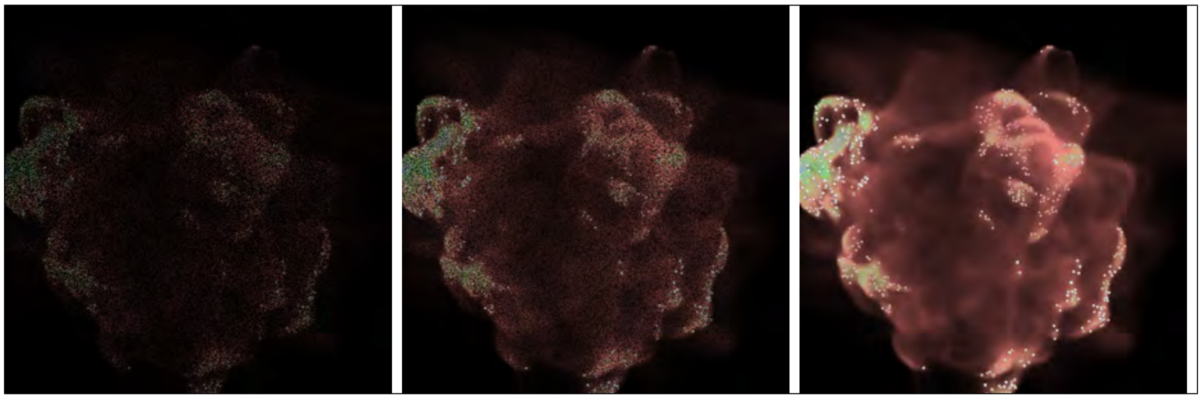


Figure 28: Ray-based particle sampling with asynchronous rendering within Galaxy, as demonstrated during a tutorial presentation at the Intel SC18 booth. Galaxy uses Embree and OSPRay from the Intel Rendering Framework to perform low-level ray operations to leverage the hardware-tuned performance provided by these packages.

The Stardust project team has engaged with industry in several key respects to increase the efficiency, reach and impact of the delivered work. We incorporated industry-developed libraries into Stardust software to provide hardware-optimized solutions for key low-level routines to enhance development efficiency and application performance. We also leveraged industry partnerships to amplify our dissemination efforts and to build community around ray tracing solutions for large data science.

Galaxy's low-level ray tracing operations use the Embree and OSPRay libraries from the Intel Rendering Framework, produced by the Austin-based Intel Graphics group managed by Jim Jeffers. These libraries are written using Intel's ISPC vector-parallel C++ extensions that provide high performance implementations for wide vector instructions (e.g., AVX2, AVX512) available on modern Intel processors. Using these libraries enables Galaxy to achieve high rendering and analysis performance without requiring specialized graphics hardware. Using these libraries also enables Galaxy to incorporate future features and optimizations developed by Intel with little additional development effort.

The Stardust team also leveraged the Intel Graphics and Visualization Institute of Xcellence at TACC (PI Navrátil) to enhance the reach of Galaxy and to build community around ray tracing for science. Through partnerships with Intel, Kitware, Intelligent Light, other Department of Energy labs, and universities in the US, UK and EU, the Stardust team led two efforts: the IXPUG In Situ Hackathon [12] (founded by Navrátil), which advances practical adoption and implementation of in situ analysis methods within simulation codes, and the SOLAR Ray Tracing Consortium [11] (founded by Navrátil and Ahrens), which advances the use of hardware-optimized ray tracing libraries for scientific analysis and for simulation acceleration. These efforts have led to adoption of Galaxy as the in situ rendering solution for a particle-based landslide simulation code (PI Krishna Kumar, UT-Austin) and ongoing work to incorporate Galaxy as the distributed ray tracing solution within OSPRay, ParaView and VisIt. Work begun at the initial SOLAR consortium meeting co-hosted by LANL and TACC resulted in the formation of a Khronos working group of industry, lab and academic partners to develop an interface standard for analytic ray tracing, codenamed “ANARI” [6]. Once the standard is finalized, we expect future work to include expansion of the interface to include simulation acceleration routines as advocated through the SOLAR consortium.

Figure 29: Discussions for advancing ray tracing use at the inaugural SOLAR consortium meeting in Santa Fe, NM, May 2019, co-hosted by LANL and TACC.

Figure 30: Presenters, attendees, and post-event discussion for the SOLAR Consortium “Birds of a Feather” (BOF) held at SC19 in Denver, CO. The presenters included Estelle Dirand (TOTAL), Jeff Amstutz (Intel), Pascal Grosset (LANL), Paul Navrátil (TACC) and Peter Messmer (NVIDIA).

4 Artifacts

The *Stardust* project team has created a wide range of publications, open source software releases and web content to provide access to the novel capabilities and ideas developed during this process.

4.1 Publications

- Greg Abram, Paul Navrátil, David Rogers and James Ahrens. Distributed Multi-tenant In Situ Analysis using Galaxy. in In Situ Visualization for Computational Science. Hank Childs, Janine Bennett, Christoph Garth, eds., 2020
- Francesca Samsel, Trinity Overmyer, and Paul A. Navrátil. Highlight Insert Colormaps: Luminance for Focused Data Analysis. In Jimmy Johansson, Filip Sadlo, and G. Elisabeta Marai, editors, *EuroVis 2019 - Short Papers*. The Eurographics Association, 2019
- Francesca Samsel, Phillip Wolfram, Annie Bares, Terece Turton, and Roxana Bujack. Colormapping resources and strategies for organized intuitive environmental visualization. *Environmental Earth Sciences*, 78, 04 2019
- Francesca Samsel and Lyn Bartram. Art, affect and color: Creating engaging expressive scientific visualization. IEEE Vis, VISAP, Conference Berlin, 10 2018
- Colin Ware, Turton Terry, Roxana Bujack, Francesca Samsel, Piyush Shrivastava, and David H. Rogers. Measuring and modeling the feature detection threshold functions of colormaps. IEEE Vis Conference Vancouver
- Andrew H Stevens, Colin Ware, Thomas Butkiewicz, David Rogers, and Greg Abram. Hairy slices ii: Depth cues for visualizing 3d streamlines through cutting planes. In *Computer Graphics Forum*, volume 39, pages 25–35. Wiley Online Library, 2020
- Colin Ware, Francesca Samsel, David H Rogers, Paul Navratil, and Ayat Mohammed. Designing pairs of colormaps for visualizing bivariate scalar fields. 2020
- Colin Ware, Terece L Turton, Roxana Bujack, Francesca Samsel, Piyush Shrivastava, and David H Rogers. Measuring and modeling the feature detection threshold functions of colormaps. *IEEE transactions on visualization and computer graphics*, 25(9):2777–2790, 2018
- P. Nardini, M. Chen, F. Samsel, R. Bujack, M. Bottinger, and G. Scheuermann. The making of continuous colormaps. TVCG, 2019
- Francesca Samsel. Optimizing color's potential: A hands-on tutorial on color tools and strategies enabling effective exploration, knowledge extraction and communicate your data and science, December 2020

- G. Abram, P. Navrátil, P. Grossett, D. Rogers, and J. Ahrens. Galaxy: Asynchronous ray tracing for large high-fidelity visualization. In *2018 IEEE 8th Symposium on Large Data Analysis and Visualization (LDAV)*, pages 72–76, 2018

- Stephanie Zeller and David Rogers. Visualizing science: How color determines what we see

- Stephanie Zeller, Francesca Samsel, and Paul Navártil. Environmental visualization: Moving beyond the rainbows. In *Practice and Experience in Advanced Research Computing*, pages 321–326. 2020

- Francesca Samsel, John M Patchett, David Honegger Rogers, and Karen Tsai. Employing color theory to visualize volume-rendered multivariate ensembles of asteroid impact simulations. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 1126–1134, 2017

- John M Patchett, Boonthanome Nouanesengsy, James Paul Ahrens, Michael Kenneth Lang, David Honegger Rogers, Jennifer Kathleen Green, Francesca Samsel, Giovanni Antonio Cone, and Hans-Jurgen Hagen. Delivery of in situ capability to end users

- Francesca Samsel Kristin Hoch Greg Abram Sam Jones Alex Gaglianon Joseph Smidt, Brandon Wiggins. The first water in the universe

- Colin Ware, Francesca Samsel, David H Rogers, Paul Navratil, and Ayat Mohammed. Designing pairs of colormaps for visualizing bivariate scalar fields. 2020

- Francesca Samsel, John M Patchett, David Honegger Rogers, and Karen Tsai. Employing color theory to visualize volume-rendered multivariate ensembles of asteroid impact simulations. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 1126–1134, 2017

- Roxana Bujack, Terece L Turton, Francesca Samsel, Colin Ware, David H Rogers, and James Ahrens. The good, the bad, and the ugly: A theoretical framework for the assessment of continuous colormaps. *IEEE transactions on visualization and computer graphics*, 24(1):923–933, 2017

4.2 Invited Talks

- Optimizing Color Application for Clear, Complex, Communicative Visualization, Francesca Samsel, Lawrence Livermore National Laboratory

- Color Research and Resources for Scientific Visualization, Francesca Samsel, DOE Computer Graphics Forum 2020

- Perspectives in Color Research for Scientific Visualization: Understanding the Lenses and Languages, Organizer and Lead Panelist, Francesca Samsel, University of Texas at Austin, Panelists: Roxana Bujack, Los Alamos National Laboratory; Karen Schloss, University of Wisconsin; Lyn Bartram, Simon Fraser University; Andrew Stockman, UK, IEEE Vis 2018, Berlin, Germany

- Storyboards for Science: Combining the Visual and Verbal to Create Engaging Communication, Workshop, David Rogers, LANL, Francesca Samsel, UT Austin, Benjamin Bach, University of Edinburgh, IEEE Vis 2018, Berlin, Germany

- Colormapping Strategies for Large Multivariate Scientific Visualization, Francesca Samsel, DOE, IDEAS-ECP Productivity Webinar Series, http://ideas-productivity.org/wordpress/wp-content/uploads/2020/08/webinar043-sciviscolor.pdf

- Scientific Visualization: Employing Art, Design, and Color for Clarity and Impact, Francesca Samsel, TACC Summer Institute on Scientific Visualization 2020

- Color Tools and Strategies for Effective Engaging Visualizations, Francesca Samsel, Scientific Tutorial, AGU Ocean Sciences 2020, San Diego, CA

- Francesca Samsel, Danielle Szafir, Karen Schloss, Theory and Application of Visualization Color Tools and Strategies, IEEE Vis 2020 tutorial.

- Francesca Samsel, Co-convener, Color in Visualization: Strategies and Resources for Effective Exploration and Effective Communication of Climate Science, AGU Fall Meeting 2019, San Francisco, CA.

- Storyboards for Science: Combining the Visual and Verbal to Create Engaging Communication Workshop, David Rogers, LANL, Francesca Samsel, UT Austin, Matteo Farinella, Columbia University, Celia Gurney, Climate Nexus, AGU Fall Meeting 2018, Washington, D.C.

- Melding Computer Science and Artistic Design Theory, F. Samsel, A. Bares, Grace Hopper Conference 2019.

- Francesca Samsel, Employing Color to Create Effective Visualizations for Affective Communication, AGU Fall Meeting 2019, San Francisco, CA.

- Stephanie Zeller, Francesca Samsel, Improving the Science Communication Pipeline: A Proposed Framework for Increasing Accessibility and Engagement for Large, Multivariate and Dynamic Data thru Intuitive Visualization Methods and Color Theory Application, AGU Fall Meeting 2019, San Francisco, CA.

- Francesca Samsel,Creating Intuitive, Engaging Visualization: Color Tools and Strategies for Quickly and Easily Creating Visualizations Revealing Data and Communicating Science, AGU Ocean Sciences 2019, San Diego, CA.

- Discovery Jam II: Science, Design, Discovery Workshop, D. Rogers, F. Samsel, D. Keefe, IEEE Vis 2017, Phoenix, AZ

- Greg Abram, Galaxy-An Update. DOE CGF 2020

- Paul Navrátil. TACC site update. DOE Computer Graphics Forum. April 28, 2020

4.3 Software Releases

- Galaxy asynchronous engine:
  `https://github.com/stardustscience/Galaxy.git`
  (mirrored at `https://github.com/TACC/Galaxy.git`)
- Online color editor `http://vislab-ccom.unh.edu/colorEditCIElab`
- ParaViewCustomModules. A set of ParaView filtesr that extend ParaView to support prototyping of visualizations for the Stardust project. Details and code at:
  `https://github.com/stardustscience/ParaviewCustomModules`

4.4 Web pages

- Project web page at `https://stardustscience.github.io`
- StardustScience group at `https://github.com/stardustscience`
- Scientific Color portal at `https://sciviscolor.org`

References

[1] Visualizing science: How color determines what we see. *EOS*, May 2020.

[2] G. Abram, P. Navrátil, P. Grossett, D. Rogers, and J. Ahrens. Galaxy: Asynchronous ray tracing for large high-fidelity visualization. In *2018 IEEE 8th Symposium on Large Data Analysis and Visualization (LDAV)*, pages 72–76, 2018.

[3] Ann S. Almgren, John B. Bell, Mike J. Lijewski, Zarija Lukic, and Ethan Van Andel. Nyx: A massively parallel amr code for computational cosmology. *Astrophysical Journal*, 765(1), 3 2013.

[4] Roxana Bujack, Terece L Turton, Francesca Samsel, Colin Ware, David H Rogers, and James Ahrens. The good, the bad, and the ugly: A theoretical framework for the assessment of continuous colormaps. *IEEE transactions on visualization and computer graphics*, 24(1):923–933, 2017.

[5] Y. L. Chang and L. Y. Oey. Analysis of stcc eddies using the okubo–weiss parameter on model and satellite data. *Ocean Dynamics*, 64(2):259–271, Feb 2014.

[6] Khronos Gruop. Anari overview. `https://www.khronos.org/anari`, September 2020.

[7] Francesca Samsel Kristin Hoch Greg Abram Sam Jones Alex Gaglianon Joseph Smidt, Brandon Wiggins. The first water in the universe.

[8] D. Livescu, C. Canada, K. Kanov, R. Burns, IDIES, and J. Pulido. Homogeneous buoyancy driven turbulence data set, 2014. http://turbulence.pha.jhu.edu/docs/README-HBDT.pdf.

[9] P. Nardini, M. Chen, F. Samsel, R. Bujack, M. Bottinger, and G. Scheuermann. The making of continuous colormaps. TVCG, 2019.

[10] Kaitlin A. Naughten, Katrin J. Meissner, Benjamin K. Galton-Fenzi, Matthew H. England, Ralph Timmermann, and Hartmut H. Hellmer. Future Projections of Antarctic Ice Shelf Melting Based on CMIP5 Scenarios. *Journal of Climate*, 31(13):5243–5261, 06 2018.

[11] Paul Navratil. Accelerated ray tracing for scientific simulations. `https://solarrt.org`, Nov 2019.

[12] Paul Navratil. Ixpug in situ analysis hackathon. `https://www.ixpug.org/events/in-situ-hackathon`, May 2019.

[13] John M Patchett, Boonthanome Nouanesengsy, James Paul Ahrens, Michael Kenneth Lang, David Honegger Rogers, Jennifer Kathleen Green, Francesca Samsel, Giovanni Antonio Cone, and Hans-Jurgen Hagen. Delivery of in situ capability to end users.

[14] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. 21(4):25–34, August 1987.

[15] Todd Ringler, Mark Petersen, Robert L. Higdon, Doug Jacobsen, Philip W. Jones, and Mathew Maltrud. A multi-resolution approach to global ocean modeling. *Ocean Modelling*, 69:211 – 232, 2013.

[16] F. Samsel, S. Klaassen, and D. H. Rogers. Colormoves: Real-time interactive colormap construction for scientific visualization. 38:20–29.

[17] Francesca Samsel. Colormapping strategies for large multivariate data in scientific applications. `http://ideas-productivity.org/events/hpc-best-practices-webinars/`, August 2020.

[18] Francesca Samsel. Optimizing color's potential: A hands-on tutorial on color tools and strategies enabling effective exploration, knowledge extraction and communicate your data and science, December 2020.

[19] Francesca Samsel and Lyn Bartram. Art, affect and color: Creating engaging expressive scientific visualization. IEEE Vis, VISAP, Conference Berlin, 10 2018.

[20] Francesca Samsel, Trinity Overmyer, and Paul A. Navrátil. Highlight Insert Colormaps: Luminance for Focused Data Analysis. In Jimmy Johansson, Filip Sadlo, and G. Elisabeta Marai, editors, *EuroVis 2019 - Short Papers*. The Eurographics Association, 2019.

[21] Francesca Samsel, John M Patchett, David Honegger Rogers, and Karen Tsai. Employing color theory to visualize volume-rendered multivariate ensembles of asteroid impact simulations. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 1126–1134, 2017.

[22] Francesca Samsel, Phillip Wolfram, Annie Bares, Terece Turton, and Roxana Bujack. Colormapping resources and strategies for organized intuitive environmental visualization. *Environmental Earth Sciences*, 78, 04 2019.

[23] B. K. Shivamoggi, G. J. F. Heijst, and L. P. J. Kamp. The okubo-weiss criteria in two-dimensional hydrodynamic and magnetohydrodynamic flows, 2011.

[24] Andrew H Stevens, Colin Ware, Thomas Butkiewicz, David Rogers, and Greg Abram. Hairy slices ii: Depth cues for visualizing 3d streamlines through cutting planes. In *Computer Graphics Forum*, volume 39, pages 25–35. Wiley Online Library, 2020.

[25] Colin Ware, Daniel Bolan, Ricky Miller, David H Rogers, and James P Ahrens. Animated versus static views of steady flow patterns. In *Proceedings of the ACM Symposium on Applied Perception*, pages 77–84, 2016.

[26] Colin Ware, Francesca Samsel, David H Rogers, Paul Navratil, and Ayat Mohammed. Designing pairs of colormaps for visualizing bivariate scalar fields. 2020.

[27] Colin Ware, Turton Terry, Roxana Bujack, Francesca Samsel, Piyush Shrivastava, and David H. Rogers. Measuring and modeling the feature detection threshold functions of colormaps. IEEE Vis Conference Vancouver.

[28] Colin Ware, Terece L Turton, Roxana Bujack, Francesca Samsel, Piyush Shrivastava, and David H Rogers. Measuring and modeling the feature detection threshold functions of colormaps. *IEEE transactions on visualization and computer graphics*, 25(9):2777–2790, 2018.

[29] Stephanie Zeller and David Rogers. Visualizing science: How color determines what we see.

[30] Stephanie Zeller, Francesca Samsel, and Paul Navártil. Environmental visualization: Moving beyond the rainbows. PEARC '20, page 321–326, New York, NY, USA, 2020. Association for Computing Machinery.

[31] Stephanie Zeller, Francesca Samsel, and Paul Navártil. Environmental visualization: Moving beyond the rainbows. In *Practice and Experience in Advanced Research Computing*, pages 321–326. 2020.