Challenges in visualizing results from more modern finite element formulations: high-order, DG, H(div), H(curl)...

Défis dans la visualisation des résultats issus de formulations éléments finis modernes : ordre-élevé, DG, H(div), H(curl)...

Abstract—As numerical methods for modeling and simulating physical phenomena evolve and improve they also shift programming paradigms and data structures accordingly. In more modern approaches of the Finite Element Method (FEM), research into using higher order basis functions, discontinuous functional spaces and even topological concepts such as homology and co-homology has been gaining traction and the formulations it provides are becoming increasingly mainstream in both academia and industry. As both adaptation of existing simulation codes and genesis of new software for taking advantage of these advanced methodologies multiply in number, so does the zoology of different types of numerical results and the formats used to describe those results. Post-processing and visualization tools, such as ParaView, face challenges on multiple levels when it comes to supporting datasets generated by these advanced methods. As such, many barriers to visualizing these results correctly using the rendering tools currently available are being encountered by the scientific community. This work is dedicated to formalizing and cataloging the different types of obstacles faced in the post-processing of these increasingly common types of data sets as well as proposing some pathways forward into reducing the feature gap and incrementally constructing support for evolving FEM research.

Thanks to its solid mathematical foundation in functional theory [5], the Finite Element (FE) method stands out amongst many numerical methods as a flexible framework for research and continuous improvement. Of the many formulations that exist [1], [8], this work is concerned, rather pragmatically, with specific FE approaches that are becoming more popular thanks to both their desirable numerical properties and advancements (and limitations) in computational hardware. More precisely, this formalizing effort is dedicated to three commonalities that most of the newest FE approaches share or combine in some shape or form: high-order element-wise basis functions [15] formulations integrating discontinuities between elements (so-called Discontinuous Galerkin or DG methods) [7], [9], [14] and certain conforming elements baking in constraints on the solution, such as divergence (H(div)) or curl (H(curl)) L^2 integrability, into the functional spaces themselves [2], [3]. While the research into these kinds of approaches is ongoing, their use cases are proliferating thanks to either desirable numerical properties (such as stability and/or accuracy) or gains in computational efficiency or both [10].

However, the improvements generated by these approaches usually incur a cost in the form of added complexity. Indeed, common conventional expectations of scientific data such that data be localized at points or highest dimensional cells, that degrees of freedom necessarily have geometric supports or even that basis functions are scalar by nature are being questioned by these modern formulations. This poses serious challenges to tools which are dedicated to post-processing and visualizing scientific data since conventional data formats, data structures and even certain data processing pipelines are no longer valid in these generalized settings [6]. Beyond a few preliminary publications around this issue [11], [12], no current post-processing solution is equipped to furnish the entirety of support needed when it comes to these novel structures. While these changes to the status-quo of data coming out of FE simulation pose a certain number of challenges to their integration and visualization in classical processing pipelines, they also present opportunities to improve and evolve approaches to post-processing scientific data in more powerful and generic directions.

Challenge Overview

The first obstacle to natively treating these results is in the complexity of interpolating values in more exotic cell types. While interpolating linear values in most common geometries is a common occurrence that can be performed with little cost, interpolation primitives in more exotic cells can present more of

- *Julien Fausty: Kitware*
  E-mail: julien.fausty@kitware.com.
- *Charles Gueunet: Kitware*
- *Mathieu Westphal: Kitware*
- *Fran cois Mazen: Kitware*

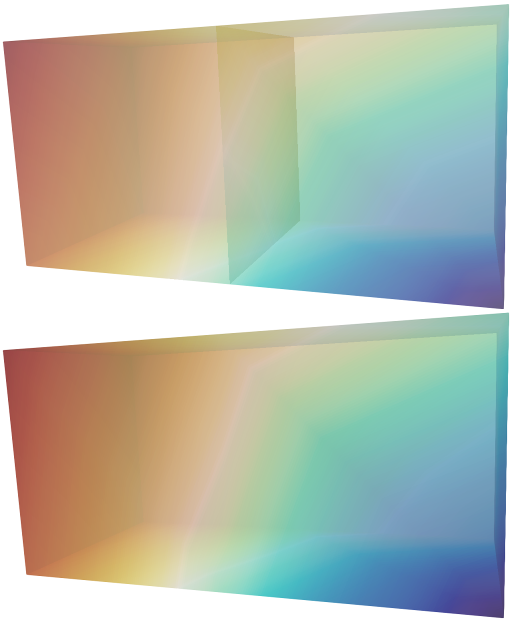

Fig. 1. Visualization of an artificial internal face rendered due to the discontinuous nature of the field (above) and similar continuous data (below) (rendered with ParaView)

a challenge with moderate sized linear systems that need to be inverted and degree of freedom mapping to complicated basis functions. This observation is valid not only for CPU based interpolation but also for rendering operations on the GPU making their implementation all the more challenging.

General reduction operations, such as slices and clips, also need to be redefined in the context of more generic element types. While relatively straightforward to construct a new linear cell from an arbitrary shape, higher-order or purely modal elements most likely will not support arbitrary element geometries. These kinds of cell-level decomposition or reduction operations will have to be accompanied by an interpolation-remeshing phase leading to added complexity as well as special care to design topology conserving operations on the discretized meshes.

Inter-cell discontinuities present in the more modern FE results also pose interesting problems in both post-processing and rendering. Operations like isocontouring or streamline tracing do not have native definitions in the context of discontinuous solutions. Aprubt jumps in solution values can also lead to rendering artifacts, illustrated in Figure 1 using current visualization frameworks.

Beyond theoretical and algorithmic considerations, pragmatic challenges in terms of software infrastructure are often underestimated in both academic and industrial projects. Indeed, considerable thought and energy has been devoted to the current structures and APIs in visualization software and libraries (such as the Visualization ToolKit (VTK) [13] and ParaView [4]). However, these modern FE solutions are currently pushing the limits on natively describing data with high fidelity in these frameworks. Efforts dedicated to improving these tools can effectively be divided into two equally important aspects: redesigning scientific data structures to support describing these modern FE solutions and refactoring existing Application Programmer Interfaces (APIs) to support operations on these enhanced formulations. In this context, the need for a dedicated road map resulting in a modular system where structures and interfaces are reusable in multiple settings is pressing.

Potential Solutions

The approach with the fastest "time to solution" is to bridge the gap between the modern FE result and current visualization capabilities. This strategy entails projecting the enriched solution on a simulation mesh onto a more refined visualization mesh using more basic element-wise solution spaces [12]. This approach thus strikes a compromise between the fidelity of the visualized solution and computational cost (both in memory and compute cycles) a user is willing to expend to render their data. As one increases the accuracy of the visualization one must use more computational resources to both project the original data onto the visualization mesh as well as render the refined linear dataset representation. Figure 2 illustrates iteratively refining the subdivisions of non-linear surface cells increasing the fidelity of the geometry representation. However, the wastefulness of this approach is that it requires more memory and computational resources to treat a FE solution than if a visualization tool had native support for describing the intricacies of the solution space at every step of the process.

Ideally, support for visualizing the different types of FE formulations would be directly implemented in the post-processing pipelines. Given the diversity of different types of formulations, this approach would incur a phenomenal amount of effort sustained in time. However, if one could devise sufficiently generic cell and dataset interfaces so that users could provide their own implementations of computational primitives (interpolation, integration, intersection, localization, etc.), then they would be able to use the implementations straight from the simulation code to also visualize the results. As such, users could provide sets of computational primitives with a clear, minimal and generic API and unlock additional forms of post-processing as they provide more and more information about their datasets. While demanding more work from the end-user, this approach to clearly defining the computational primitives and then providing a generic API has the potential to be successful in a domain that is still undergoing active research. A caveat that would need to be addressed is the performance or even ability of user defined and compiled

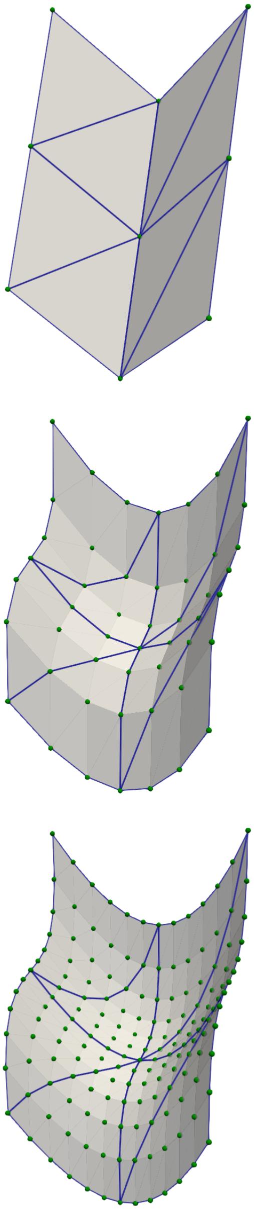

Fig. 2. Illustration of the successive subdivision of non-linear cells (from top to bottom: none, 1 point per edge, 2 points per edge) to improve the approximate representation of the underlying geometry of a saddle point (rendered with ParaView)

computational primitives on rendering acceleration hardware such as the GPU.

Outlook

As it is, this push for improving the capabilities and formulations of FE codes needs to be accompanied by evolutions in post-processing tools. In the face of currently unsatisfactory support for adequately treating the results of these more advanced FEMs, work on how to better support these data structures needs to accelerate now in order to catch up with advances in the methods themselves. Kitware, as the most active maintainer of both ParaView and VTK, is uniquely positionned at the interface between data visualization and scientific modeling and looking to improve both the methods and tools for treating these advanced scientific datasets.

References

[1] P. F. Antonietti, A. Cangiani, J. Collis, Z. Dong, E. H. Georgoulis, S. Giani, and P. Houston. Review of discontinuous galerkin finite element methods for partial differential equations on complicated domains. *Building bridges: connections and challenges in modern approaches to numerical partial differential equations*, pages 281–310, 2016.

[2] D. Arnold, R. Falk, and R. Winther. Finite element exterior calculus: from hodge theory to numerical stability. *Bulletin of the American mathematical society*, 47(2):281–354, 2010.

[3] D. N. Arnold. *Finite element exterior calculus*. SIAM, 2018.

[4] U. Ayachit. *The paraview guide: a parallel visualization application*. Kitware, Inc., 2015.

[5] S. C. Brenner, L. R. Scott, and L. R. Scott. *The mathematical theory of finite element methods*, volume 3. Springer, 2008.

[6] H. Childs, B. Geveci, W. Schroeder, J. Meredith, K. Moreland, C. Sewell, T. Kuhlen, and E. W. Bethel. Research challenges for visualization software. *Computer*, 46(5):34–42, 2013.

[7] B. Cockburn and C.-W. Shu. Runge–kutta discontinuous galerkin methods for convection-dominated problems. *Journal of scientific computing*, 16:173–261, 2001.

[8] T.-P. Fries and T. Belytschko. The extended/generalized finite element method: an overview of the method and its applications. *International journal for numerical methods in engineering*, 84(3):253–304, 2010.

[9] Q. Hong, F. Wang, S. Wu, and J. Xu. A unified study of continuous and discontinuous galerkin methods. *Science China Mathematics*, 62:1–32, 2019.

[10] R. M. Kirby, S. J. Sherwin, and B. Cockburn. To cg or to hdg: a comparative study. *Journal of Scientific Computing*, 51:183–212, 2012.

[11] B. Nelson, E. Liu, R. M. Kirby, and R. Haimes. Elvis: A system for the accurate and interactive visualization of high-order finite element solutions. *IEEE transactions on visualization and computer graphics*, 18(12):2325–2334, 2012.

[12] J.-F. Remacle, N. Chevaugeon, E. Marchandise, and C. Geuzaine. Efficient visualization of high-order finite elements. *International Journal for Numerical Methods in Engineering*, 69(4):750–771, 2007.

[13] W. Schroeder, K. M. Martin, and W. E. Lorensen. *The visualization toolkit an object-oriented approach to 3D graphics*. Prentice-Hall, Inc., 1998.

[14] C.-W. Shu. Discontinuous galerkin method for time-dependent problems: survey and recent developments. *Recent Developments in Discontinuous Galerkin Finite Element Methods for Partial Differential Equations: 2012 John H Barrett Memorial Lectures*, pages 25–62, 2014.

[15] P. Solin, K. Segeth, and I. Dolezel. *Higher-order finite element methods*. CRC press, 2003.