Preprint, 14 July 2015

Geometry + Simulation Modules: Implementing Isogeometric Analysis

Bert Jüttler[1,3], **Ulrich Langer**[2,3], **Angelos Mantzaflaris**[3*], **Stephen E. Moore**[3], and **Walter Zulehner**[2]

[1] Institute of Applied Geometry, Johannes Kepler University, Linz, Austria
[2] Institute of Computational Mathematics, Johannes Kepler University, Linz, Austria
[3] Johann Radon Institute for Computational and Applied Mathematics (RICAM), Austrian Academy of Sciences

Isogeometric analysis (IGA) is a recently developed simulation method that allows integration of finite element analysis (FEA) with conventional computer-aided design (CAD) software [1,3]. This goal requires new software design strategies, in order to enable the use of CAD data in the analysis pipeline. To this end, we have initiated G+Smo (Geometry+Simulation Modules), an open-source, C++ library for IGA. G+Smo is an object-oriented, template library, that implements a generic concept for IGA, based on abstract classes for discretization basis, geometry map, assembler, solver and so on. It makes use of object polymorphism and inheritance techniques to provide a common framework for IGA, for a variety of different basis-types which are available. A highlight of our design is the dimension independent code, realized by means of template meta-programming. Some of the features already available include computing with B-spline, Bernstein, NURBS bases, as well as hierarchical and truncated hierarchical bases of arbitrary polynomial order. These basis functions are used in continuous and discontinuous Galerkin approximation of PDEs over (non-)conforming multi-patch computational (physical) domains.

1 Isogeometric Analysis

Isogeometric analysis, introduced by Hughes *et al.* in 2005 [1], proposes the integration of design and analysis, i.e. a bi-directional communication between Computer-aided Design (CAD) and simulation software. The last years this topic has received increased interest from the communities of engineering, geometric design and of numerical analysis, due to its promising potential.

IGA uses the same class of basis functions for both representing the geometry of the computational domain and approximating the solution of problems modeled by PDEs. This approach allows to perform simulations directly on CAD models, avoiding the expensive intermediate meshing step. Some of its advantages include the property that the exact CAD geometry is preserved and the fact that considerably less degrees of freedom are required when comparing with high-order finite element discretizations. The discretization spaces used in IGA are collections of Non-Uniform Rational B-Splines (NURBS) basis functions, which are widespread in CAD for representing surfaces (Fig. 1), as well as an industrial standard. They are known to have good approximation properties which make them suitable for analysis. Another advantage is the increased smoothness across element edges or faces.

The B-spline basis functions of a given degree are defined in a set of successive intervals (1D elements), whose endpoints are called knots. Their evaluation is done by the Cox-de Boor recursion formula applied to the knot vector. In 2D or 3D, the respective B-spline basis functions are derived as tensor products of one dimensional B-Spline basis functions, therefore defined on a rectangular domain (Fig. 2). For a more detailed description we refer the reader to [3].

Several challenges arise when applying the IGA paradigm. One important problem comes from the fact that the refinement propagates along knot-lines of tensor-product bases, causing the number of control variables to explode. Also, even though meshing is not needed, the conversion of CAD solids (B-reps) to analysis-suitable models is a non-trivial task. CAD models consist of many patches, each one parameterized by B-spline basis functions. These play the role of sub-domains, and coupling

* Corresponding author: e-mail Angelos.Mantzaflaris@oeaw.ac.at, phone +0043 732 2468 5279

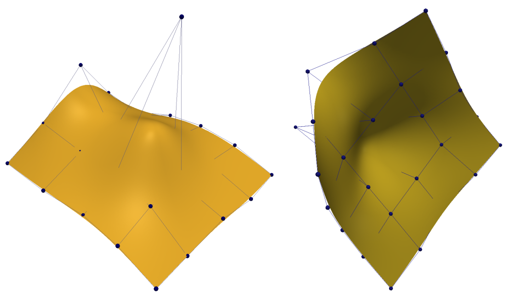

Fig. 1: A surface patch defined by B-splines. Black dots denote the control points.

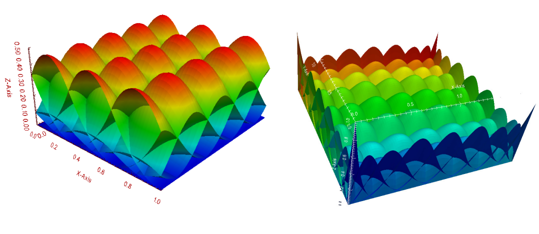

Fig. 2: Bi-variate tensor-product B-splines defined on a square domain. End conditions may be un-clamped (left) or clamped (right).

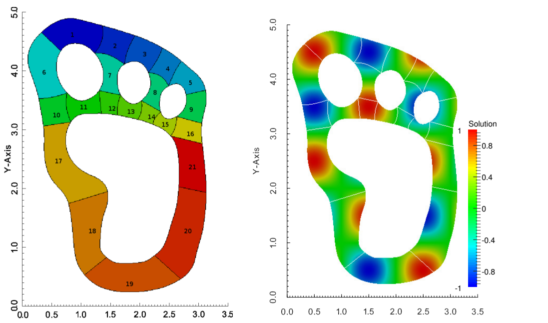

Fig. 3: Solving a 2D diffusion problem. defined on a domain partitioned in B-spline patches. Coupling of interfaces is using a IGA Discontinuous Galerkin method.

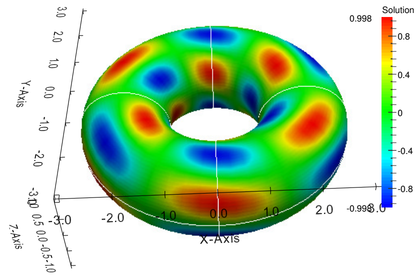

Fig. 4: A diffusion PDE on a torus surface, both the geometry and the solution are parameterized by NURBS basis functions.

the interfaces between them is a non-trivial task, especially when the parameterizations at the interface are non-matching or contains small gaps. Furthermore, this topology of patches allows extra-ordinary vertices and singularities, which require special treatment.

2 A C++ library for IGA

Isogeometric analysis requires seamless integration of FEA and CAD software. The existing software libraries, however, cannot be adapted easily to the arising new challenges, since they have been designed and developed for different purposes. In particular, FEA codes are traditionally implemented by means of functional programming, and are focused on treating nodal shape function spaces. In CAD packages, on the other hand, the central objects are free-form curves and surfaces, defined by control points, which are realized in an object-oriented programming environment.

G+SMO is an object-oriented, template C++ library, that implements a generic concept for IGA, based on abstract classes for geometry map, discretization basis, assemblers, solvers and so on. It makes use of object polymorphism and inheritance techniques in order to support a variety of different discretization bases, namely B-spline, Bernstein, NURBS bases, hierarchical and truncated hierarchical B-spline bases of arbitrary polynomial order [4], and so on.

Regarding architecture, the library is organized into three main levels: The iso-geometry level, which contains the abstract IGA, geometry and simulation concepts, the implementation level where a variety of different realizations of IGA bases, geometries, solvers is done, and the I/O level, that consists of utilities related to file input and output, visualization and plug-ins for third-party software.

Our design allows the treatment of geometric entities such as surfaces or volumes through dimension independent code, realized by means of template meta-programming. Available features include simulations using continuous and discontinuous Galerkin approximation of PDEs, over conforming and non-conforming multi-patch computational domains [5], see Fig. 3. PDEs on surfaces(Fig. 4) as well as integral equations arising from elliptic boundary value problems are also supported. Boundary conditions may be imposed both strongly and weakly. In addition to advanced discretization and generation techniques [6], efficient solvers like multi-grid iteration schemes are available [2]. Methods for solving non-linear problems are under development. Finally, we aim to employ existing high-end libraries for large-scale parallelization. For more details visit www.gs.jku.at/gismo.

ACKNOWLEDGEMENTS G+SMO is developed in the frame of the NFN *Geometry + Simulation* (S117), supported by the Austrian Science Fund. It is jointly developed by numerous contributors at the Institutes of Numerical Analysis and of Applied Geometry, Johannes Kepler University, and at the RICAM Institute of the AAS.

References

[1] J. A. Cottrell and T. J. R. Hughes and Y. Bazilevs, Isogeometric analysis: Toward Integration of CAD and FEA, (Wiley, 2009)
[2] C. Hofreither, and W. Zulehner, http://www.gs.jku.at/pubs/NFNreport13.pdf (2014)
[3] T. J. R. Hughes and J. A. Cottrell and Y. Bazilevs, Compt. Meth. in Appl. Mech. and Engr., 39–41, **194**, (2005) pp. 4135–4195.
[4] G. Kiss, C. Giannelli, and B. Jüttler, Math. Meth. for Curves and Surfaces **8177** (2014), pp. 304–323.
[5] U. Langer and St. E. Moore, http://www.gs.jku.at/pubs/NFNreport12.pdf (2014).
[6] A. Mantzaflaris, and B. Jüttler, Math. Meth. for Curves and Surfaces, **8177**, (2014), pp. 364–382.