12th International Conference on IsoGeometric Analysis

October 27-30, 2024, Casa Monica Resort & Spa, St. Augustine, Florida, United States

TREATMENT OF LOCKING IN IGA

Hugo Casquero[1] and Manfred Bischoff[2]*

[1]*University of Michigan - Dearborn*
[2]*Universität Stuttgart*

MINISYMPOSIUM

When applied to structural theories that take into account transverse shear deformation (e.g., Timoshenko rods and Reissner-Mindlin shells), NURBS-based discretizations of the Galerkin method suffer from the same types of locking as conventional FEA discretizations based on Lagrange polynomials. When applied to structural theories that neglect transverse shear deformation (e.g., Kirchhoff rods and Kirchhoff-Love shells), NURBS-based discretizations of the Galerkin method still suffer from membrane locking. When the material considered is either nearly-incompressible or plastic, NURBS-based discretizations suffer from volumetric locking as it is the case for conventional FEA discretizations based on Lagrange polynomials. Locking results not only in smaller displacements and rotations than expected, but also large-amplitude spurious oscillations of the stresses. The large-amplitude spurious oscillations of the stresses can persist for meshes for which the displacements and rotations are already accurate. Therefore, in order to reliably use coarse NURBS-based discretizations in engineering applications, locking treatments are needed. A direct deployment of the numerical schemes used to overcome locking when using Lagrange polynomials is not an effective strategy to vanquish locking when using NURBS since the levels of continuity across element boundaries of Lagrange polynomials and NURBS are different. Thus, the higher inter-element continuity of NURBS requires the development of new numerical schemes to overcome locking. This minisymposium will feature a broad range of locking treatments for either NURBS or other types of smooth splines. Emphasis will be made on locking treatments that not only effectively overcome locking, but do so without significantly increasing the computational cost.