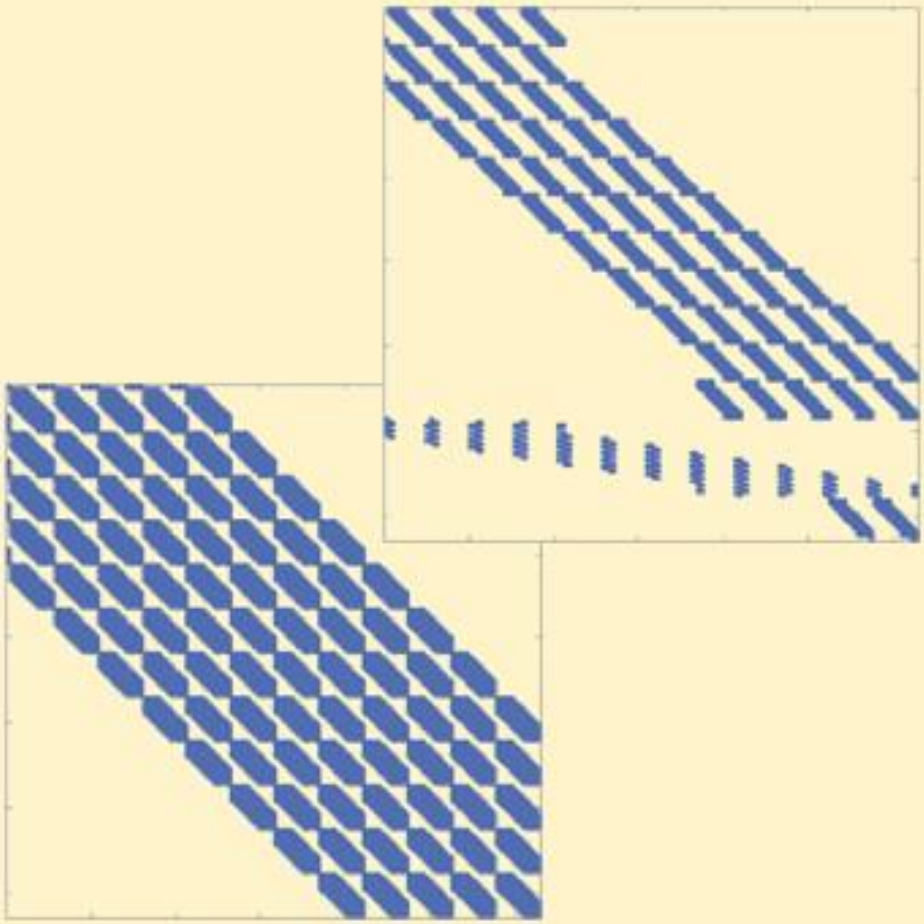
Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Πολιτικών Μηχανικών
Τομέας Δομοστατικής
Εργαστήριο Στατικής και Αντισεισμικών Ερευνών

Αξιολόγηση των Μεθοδολογιών της Ισογεωμετρικής Ανάλυσης



Σοφία Κορλού

Επιβλέποντες

Παπαδρακάκης Μανόλης Καθηγητής Ε.Μ.Π.

Καρακίτσιος Παναγιώτης Υποψήφιος Διδάκτωρ Ε.Μ.Π

Αθήνα, Οκτώβριος 2015

Αξιολόγηση των Μεθοδολογιών της Ισογεωμετρικής Ανάλυσης

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον καθηγητή κ. Παπαδρακάκη Μανόλη για την ευκαιρία που μου έδωσε να έρθω σε επαφή με τον κόσμο της έρευνας και να διεξάγω τη διπλωματική μου εργασία στα πλαίσια ενός αντικειμένου που έχει συγκεντρώσει το ενδιαφέρον της παγκόσμιας ερευνητικής κοινότητας, αυτού της ισογεωμετρικής ανάλυσης. Η ακούραστη συμπαράσταση και η συνεχής καθοδήγησή του καθόλη τη διάρκεια της εκπόνησης της παρούσας μελέτης σε συνδυασμό με την προσήνεια που τον χαρακτηρίζει ως άνθρωπο διαμόρφωσαν τις πιο ευνοϊκές συνθήκες για τη διεξαγωγή γνώσης και έρευνας.

Θα ήθελα ακόμη να ευχαριστήσω τον υποψήφιο διδάκτορα κ. Καρακίτσιο Παναγιώτη για την άψογη συνεργασία που είχαμε τα δύο τελευταία χρόνια στα πλαίσια της ερευνητικής ομάδας GIGA Team. Ο αστείρευτος ενθουσιασμός του για τη μάθηση αποτέλεσε για εμένα πηγή έμπνευσης, ενώ η διδασκαλία και ο συμβουλευτικός του ρόλος με εφοδίασαν με το απαραίτητο γνωστικό υπόβαθρο για την ενασχόληση με το συγκεκριμένο επιστημονικό πεδίο.

Ευχαριστώ θερμά τους φίλους των παιδικών και φοιτητικών μου χρόνων για την κατανόηση, την υποστήριξη και την ουσιαστική επικοινωνία, που αποτέλεσαν για μένα αρωγούς στην προσπάθεια για την ολοκλήρωση της διπλωματικής μου εργασίας.

Τέλος, ένα μεγάλο ευχαριστώ στα μέλη της οικογένειας μου για την άμετρη συμπαράσταση, την ενθάρρυνση και την ηθική στήριξη που μου προσέφεραν όλα αυτά τα χρόνια. Ιδιαίτερα στους γονείς μου οφείλω όλη την πορεία των σπουδών μου μέχρι σήμερα.

Σοφία Κορλού
Οκτώβριος 2015

Abstract

The scope of this thesis is the numerical evaluation of isogeometric analysis methodologies. Isogeometric analysis, introduced by J. Austin Cottrell, Thomas J.R. Hughes and Yuri Bazilevs, evolves from the finite element method and implements the revolutionary idea of complete CAD-CAE (Computer Aided Design-Computer Aided Engineering) integration. Properties of Non-Uniform Rational B-splines are examined thoroughly, in order to establish the theoretical framework for geometric representation. As far as analysis is concerned, the current study focuses on two formulations, Galerkin and collocation. Stiffness matrix and external load vector are formed in both methods. The comparison between the corresponding procedures points out the advantages and weaknesses of each scheme. Issues of convergence and computational cost are approached by linear static 2D and 3D applications, where the total number of degrees of freedom is the measure of cost. The code used for analysis purposes was developed in MATLAB, a programming environment which enables easy matrix manipulation and interactive simulation.

Σύνοψη

Ο σκοπός της παρούσας διπλωματικής εργασίας είναι η αξιολόγηση των μεθοδολογιών της ισογεωμετρικής ανάλυσης. Η ισογεωμετρική ανάλυση, η οποία προτάθηκε από τους J. Austin Cottrell, Thomas J.R. Hughes και Yuri Bazilevs, αποτελεί εξέλιξη της μεθόδου των πεπερασμένων στοιχείων και υλοποιεί την επαναστατική ιδέα της ενοποίησης των τεχνολογιών CAD-CAE (Computer Aided Design-Computer Aided Engineering). Οι ιδιότητες του σχεδιαστικού εργαλείου Non-Uniform Rational B-splines μελετώνται λεπτομερώς, έτσι ώστε να θεμελιωθεί το θεωρητικό υπόβαθρο για τη γεωμετρική αναπαράσταση. Όσον αφορά την ανάλυση, η συγκεκριμένη εργασία εστιάζει σε δύο μεθόδους, τη μέθοδο Galerkin και τη μέθοδο collocation. Το μητρώο στιβαρότητας και το διάνυσμα των εξωτερικών φορτίων μορφώνονται και με τους δύο σχηματισμούς. Από τη σύγκριση των αντίστοιχων διαδικασιών καταδεικνύονται τα πλεονεκτήματα και οι αδυναμίες της κάθε μεθόδου. Ζητήματα όπως αυτό της ακρίβειας και του υπολογιστικού κόστους προσεγγίζονται από δισδιάστατες και τρισδιάστατες εφαρμογές γραμμικής στατικής ανάλυσης, όπου η αξιολόγηση γίνεται με βάση τον συνολικό αριθμό βαθμών ελευθερίας. Ο κώδικας που χρησιμοποιήθηκε έχει αναπτυχθεί σε MATLAB, ένα διαδραστικό προγραμματιστικό περιβάλλον που διευκολύνει τις πράξεις με μητρώα.

Table of Contents

1 The concept of Isogeometric Analysis

1.1 Introduction

Isogeometric Analysis (IGA) was first introduced in 2003 by Thomas J.R. Hughes, Professor of Aerospace Engineering and Engineering Mechanics of the University of Texas at Austin. The proposed method became immediately the focus of attention of the whole academic and research community globally and a wide range of relevant publications have already managed to establish isogeometric analysis as a breakthrough in computational mechanics. It constitutes an innovative evolution of the standard Finite Element Method (FEM), considering that it has borrowed FEM's abstract framework and it adopts the same steps in the analysis process. The main difference between the two methods lies at the mesh generation. While the usual technique was so far to build an approximate mesh to the physical model for analysis purposes, IGA optimizes all the available information behind the geometrical representation and uses the design mesh as an analysis model. This means IGA works with the exact geometrical mesh, thus the corresponding error is eliminated.

Fig. 1.1. Thomas Joseph Robert Hughes

Through the revolutionary concept of isogeometric analysis, a complete integration of CAD (Computer Aided Design) and CAE (Computer Aided Engineering) industries is achieved. Despite the fact that computational engineering is strongly dependent on CAD, an immediate interconnection between these two fields had not ever been accomplished before IGA. A wide range of commercial software has been developed in order to serve the needs of each industry, representing economies of billion dollars. It is noteworthy that CAD market is expected to grow to $8.7 billion in 2016, while CAE will reach $3.4 billion in the same year.

CAE packages are devoted to the analysis of FEM models. They were first destined for aerospace engineering, however they soon spread to other engineering and scientific disciplines. NASTRAN, a widely used FEM platform, was originally developed by NASA in the 1960s and the release of similar packages such as Abaqus, ADINA and FEMAP followed afterwards. These programs can deal with both linear and non-linear problems, while they have applications in static and dynamic analysis, heat transfer, compressible and incompressible flows. A special category of analysis platforms is exclusively adapted to the requirements of civil engineering. ATENA, for instance, standing for Advanced Tool for Engineering Non-Linear Analysis, is specialized in reinforced concrete structures, while SOFiSTIK, first used in 1987, is directed towards bridge linear and non-linear analysis.

CAD software appeared later, in the 1970s and 1980s, when the technique for geometric representations in finite element analysis had already been established. This is probably one of the reasons why the mesh generation in CAE programs is completely separate from the model design of CAD programs. Some of the most prevalent products of CAD industry are AutoCAD, 3DS Max, Maya and Rhinoceros. These programs are able to efficiently represent any complicated geometry for both 2D and 3D drafting. They communicate with the CAE software, so as to provide information concerning the geometry, however they are not immediately connected to the analysis process. In particular, the engineering design is built from the beginning, attempting to approach the CAD representation. This may result in FEM performing badly, especially when complex structures are concerned.

It becomes clear that bridging the gap between CAE and CAD technologies was of great importance for the efficiency of FEA and troubled for a long time the research community. It was an especially challenging task, as it finally took the analysis to be reconstituted within the geometric framework of CAD tools. This venture was encouraged by the newer technologies emanating from the computational geometry research literature, which permit to exploit the functions used for the exact geometrical representation in order to describe the solution field.

The first candidate was NURBS, which had already started to spread across CAD industry. The pioneers of this technology are Pierre Bézier, a Renault engineer and Paul de Casteljau, a Citroen’s physicist and mathematician, who introduced the so-called

Bézier curves. This tool was widely used to model smooth curves, however they were not convenient enough to manipulate. Soon, the research directed to new alternatives and finally initiated B-splines. These curves were also defined by a set of points, called the “control points”, but their number was independent of the polynomial order of the curve. Additionally, they enabled a richer behavior of the model, since an alteration to a control point would only affect a specific part of the curve and not the whole entity. Nevertheless, B-splines were still not a remedy for all geometric representations, provided that they do not correspond well to the modeling of conic sections. At this point NURBS were developed in order to precisely produce shapes such as circles and ellipses. Ken Versprille was the first to work with NURBS on his dissertation in 1975. Boeing, in its ambitious project to create a single curve representation that included Bezier curves and conic sections, became the first to industrialize NURBS. After that they dominated the CAD applications and they successfully claimed a large market share of the corresponding industry.

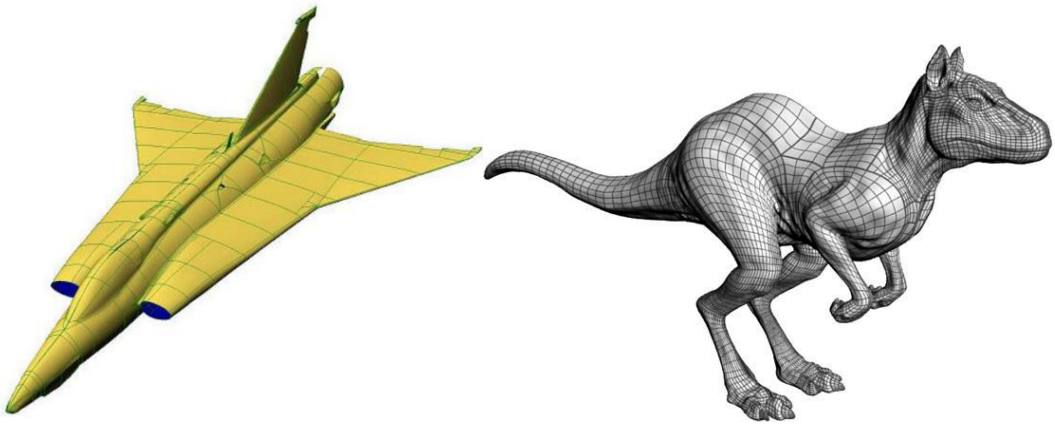
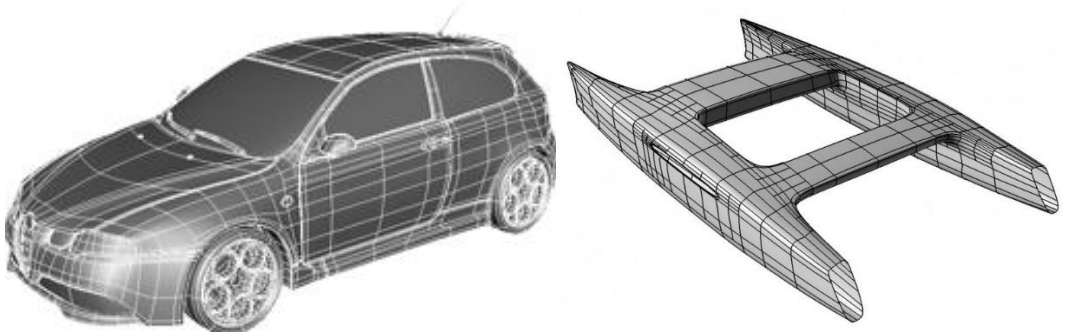
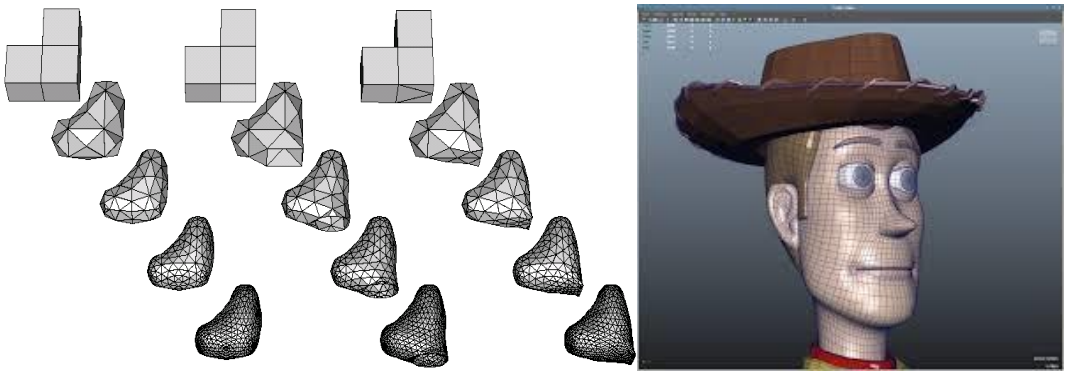


Fig. 1.2. Objects created with NURBS

After NURBS, even more evolved technologies emerged that appeal as well to the concept of isogeometric analysis. T-splines, polycube splines, subdivision surfaces are some of the representation tools proposing themselves to the new analysis method. They are all numerically stable mathematical procedures, they enable easy adjustments and modifications and they ensure that the analysis is performed on the exact geometric model. With T-splines in particular the parameterization of the whole structure is downsized to few control point coordinates, hence the problem can be solved at the lowest possible computational cost. Subdivision surfaces define the smooth curve or surface via successive refinements applied to an initial coarser piecewise linear polygon mesh. They are widely used in animation.



(a)



(b)

Fig. 1.3. Objects created with **(a)** T-splines, **(b)** subdivision surfaces

1.2 Finite element analysis

1.2.1 Evolution

The method of finite element analysis has its origins in the need for solving partial differential equations in the framework of complex elasticity and structural analysis problems. FEM achieves an approximation to the exact solution and in that sense it adopts its principles from the Galerkin method. The concept of the mesh discretization, which implies that the continuum domain is divided into a set of discrete sub-domains in order to define the displacement, stress and strain fields, was conceived by Hrennikoff and Courant. In particular the first one introduced the so called framework method, while the German mathematician Courant dealt with the problem of torsion initiating the triangular elements, in 1943.

The pioneers of the method represented mainly the aerospace industry. This is not coincidental, provided that the relevant ideas and applications were only supported by digital computation. Of course only large industrial companies were able to afford

mainframe computers during the 1950s. M. J. Turner, who worked at Boeing over the period 1950-1962, generalized and perfected the Direct Stiffness Method. B.M Irons invented the isoparametric models and the frontal solvers, an approach to solving sparse linear systems. R. J. Melosh recognized the Rayleigh-Ritz link and systematized the variational derivation of stiffness elements.

In 1943, the Royal Aeronautical Society of London met the problem of simulating the aircraft's swept-back wings. It was a really challenging issue, since none of the known methods could ensure the accurate representation of the inclined geometry. The task was assigned to John Argyris, the Greek civil engineer who became one of the creators of the finite element method. Argyris came up with the use of triangular elements that would exactly fit the desired geometry and held its first implementation in the electro-mechanical computing device the Society had just acquired and could solve equations with up to 64 unknowns. Analysis results were very close to the experimental results, with a deviation of approximately 8%. This breakthrough is considered as the birth of the finite analysis method.

Fig. 1.4. John Argyris

The "Matrix Force and Displacement Method", later established as "Finite Element Method", was promptly transferred from the aerospace industry to a wider field of engineering applications during the 1950s and 1960s, mostly by J. H. Argyris, R. W. Clough, H.C. Martin and O. C. Zienkiewicz. Clough and Martin published in 1953 a paper which is widely considered as the start of the present FEM. Clough baptized the method in 1960 and formed a research group at the University of California Berkeley, in order to adjust the idea in civil engineering problems. In 1967, Olenk Zienkiewicz wrote the first book on the subject and headed another important civil engineering research group in the University of Wales at Swansea. In the years that followed, a large amount of publications and books overwhelmed the research community, proposing techniques to evolve and enhance the method. Nowadays FEM is the most famous computational tool for performing engineering analysis and its implementations pertain

to linear or non-linear response, solid or fluid structure systems, static or transient dynamic loading.

1.2.2 Basic idea

FEM relies on the discrete representation of a physical continuum. Responses are sought only at chosen and countable points which are known as nodes. The unknown displacements correspond to degrees of freedom attributed to these nodes. The first step in the analysis procedure is to define the geometry of the problem. The mesh is created with respect to the CAD representation by computer algorithms called mesh generators. What follows is to define material properties, such as modulus of elasticity, Poisson's ratio and density (only if dynamics problems are concerned). Then, boundary and continuity conditions as well as applied loads are determined. It should be noted that both distributed and body forces are converted to nodal forces in a FE software.

In order to ensure reliable analysis results, special attention should be paid to the number and geometry of elements. As the complexity of the geometry increases, the discretization should become denser with the aim to achieve the desired convergence. The highly skewed elements should be avoided, as they lead to considerable inaccuracies. In cases that their use is obligatory, they should at least be surrounded by healthy elements. This way the unfavorable consequences will be limited. Undoubtedly, the stress results in the vicinity of the defective element will still be inaccurate, however they will be improved significantly in a small distance. Another factor that plays an important role in the efficiency of the method is the aspect ratio, which is the proportional relationship between element's width and height. The ideal aspect ratio is set to 1, while for increased values it reduces the quality of the simulation and therefore the quality of the results.

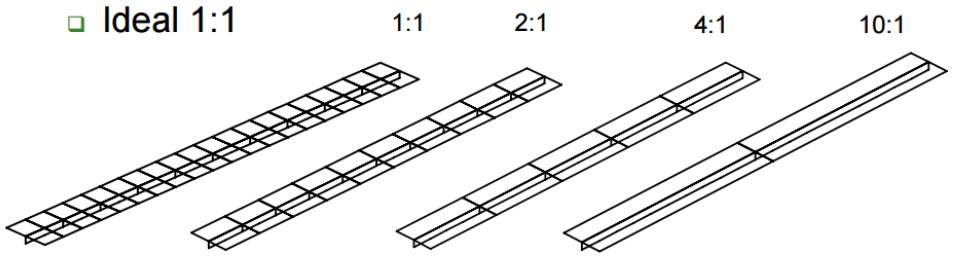


Fig. 1.5. Aspect ratio of finite elements

The approximation of the solution field is realized via piecewise polynomial functions, called the shape functions N. The displacement value $\{d\}$ at any internal point (x,y,z) of the element is calculated by interpolation of the nodal displacements $\{D\}$:

$$\underset{(3\times1)}{\{D(x,y,z)\}} = \underset{(3\times3n_e)}{[N(x,y,z)]}\ \underset{(3n_e\times1)}{\{d\}}$$

where n_e is the number of nodes per element.

Then, stress and strain vectors, which are connected through Hooke's constitutive law are defined.

$$\{\sigma\} = \left[\sigma_x\ \sigma_y\ \sigma_z\ \sigma_{xy}\ \sigma_{yz}\ \sigma_{zx}\right]^T$$

$$\{\varepsilon\} = \left[\varepsilon_x\ \varepsilon_y\ \varepsilon_z\ \varepsilon_{xy}\ \varepsilon_{yz}\ \varepsilon_{zx}\right]^T$$

$$\{\sigma\} = [E]\{\varepsilon\}$$

Deformation matrix [B] evaluates strains anywhere in the model from nodal displacements.

$$\underset{(6\times1)}{\{\varepsilon(x,y,z)\}} = \underset{(6\times3n_e)}{[B(x,y,z)]}\ \underset{(3n_e\times1)}{\{D\}}$$

The stiffness matrix [*k*] is evaluated on an element level as follows:

$$\underset{(3n_e\times3n_e)}{[k]} = \int_V \underset{(3n_e\times6)}{[B]^T}\ \underset{(6\times6)}{[E]}\ \underset{(6\times3n_e)}{[B]}\ dV$$

Distributed loads are transformed into equivalent nodal loads:

$$\underset{(3n_e\times1)}{\{r\}} = \int_V \underset{(3n_e\times3)}{[N]^T} \cdot \underset{(3\times1)}{\{f\}}\, dV$$

The final step of the procedure is to build the global stiffness matrix [*K*] and force vector {*R*} adding each element's contribution. The displacement vector {*D*} can now be calculated from:

$$\underset{(3n\times1)}{\{R\}} = \underset{(3n\times3n)}{[K]} \cdot \underset{(3n\times1)}{\{D\}}$$

where *n* is the total number of nodes.

The above relation is formulated with respect to Cartesian coordinates. However, the calculations in the natural space may prove to be extremely demanding, when complex geometries are concerned (nonrectangular elements with curved sides). FEM manages to overcome this difficulty incorporating the isoparametric method. The term isoparametric is derived from the use of the same shape functions (or interpolating

functions) *N* to represent the element's shape and to define the displacements within the element. Each element is projected on another space, called parameter space, where it has a regular shape (e.g. a cube, square or an equilateral triangle). The local coordinates of the isoparametric element range from -1 to 1 and in these limits, shape functions values and their derivatives are calculated. This variation range was chosen to facilitate use of standard Gauss integration formulas.

Actually, products of deformation *B* and elasticity *E* matrices need to be integrated over the element area or volume. The integrals are evaluated by numerical quadrature rules. Gauss quadrature is the most popular in FEM applications, since it suits perfectly to the method's features. A minimum number of required Gauss points are defined over each element and the value of the quantity to integrate is calculated at their parametric coordinates. The final system matrix is the sum of all Gauss points' contributions. The technique will be described thoroughly later.

1.2.3 Drawbacks

Despite the fact that a tremendous evolution has taken place in FEM applications and the method is now able to correspond to very demanding problems, there are still some crucial weaknesses to be solved. Without a doubt the major issue is the fact that FEM works with an approximate geometry mesh and not the actual geometry. The initial model is just used as a pattern for the mesh generation and afterwards it does not participate at all in the analysis process. Isoparametric elements take on the task of the representation, however what they offer is only an approach to the accurate figure. This means that the FE procedure is charged by birth with an error, the geometric one. After the calculations are completed, the analysis error has to be counted too. Analysis error is unavoidable considering that the analytical solution to the differential equations is not possible to be defined and only approximations can be achieved. Therefore, the total error may reach to not acceptable values when a high level of convergence is required.

As the complexity of the geometry increases, the results tend to be less accurate and thus refinement techniques have to be marshaled. The approximate geometry may conceal some of the shape's details and only a denser mesh will restore the distortion. This would not be the case if the calculations were held with respect to the exact geometry. Focusing on the refinement, it should be mentioned that it is a laborious procedure, since the fine mesh cannot be directly produced from the coarse mesh. Refinement algorithms will return to the initial geometry and produce a different approximation. Functions already completed will have to repeat from the beginning, instead of adjusting to the new conditions. This approach is time-consuming, while it involves the danger of design errors. This problem gets even more challenging when complex and innovative structures are concerned, as their behavior has not been

adequately studied yet and engineers cannot base on expected results in order to check the analysis performance.

Fig. 1.6 depicts a wine glass. In order to create this object, a designer has to define the following variables:

- degree of shape functions at each parametric axis,
- knot value vector at each parametric axis,
- control points (Cartesian coordinates and weights).

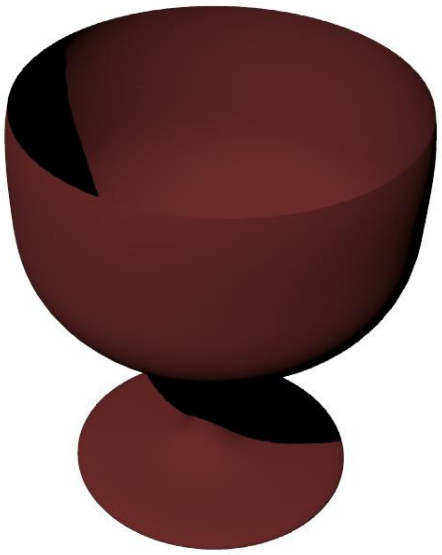


Fig. 1.6. Wine glass: geometry design

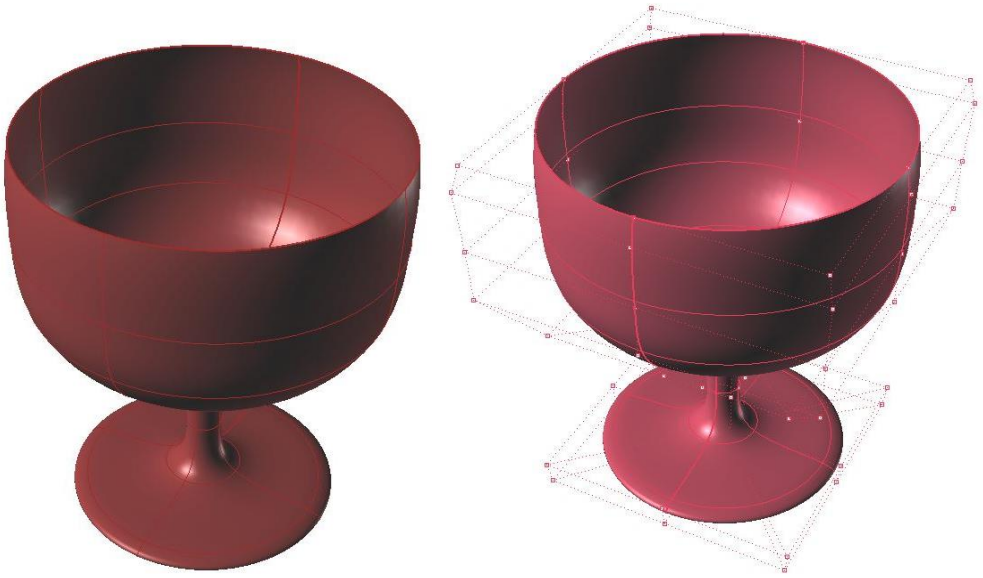


Fig. 1.7. Wine glass: initial geometry mesh

In FEM the smooth geometry of the cup is introduced as input into the FEA software and thereafter it is represented by quad finite elements. In IGA, on the contrary, the geometry remains intact and the mesh is the exact geometrical model. Fig. 1.8 shows a finer mesh of the wine glass for both cases.

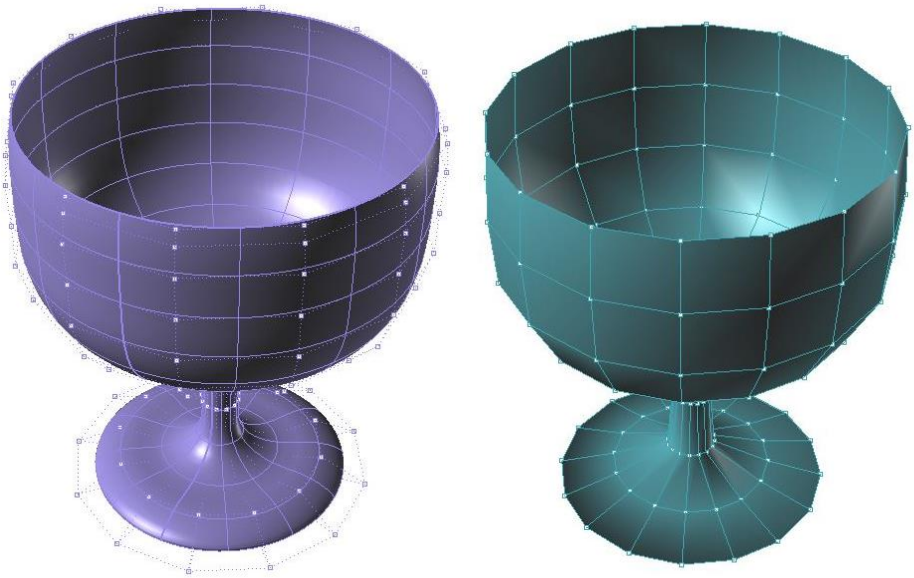


Fig. 1.8. Wine glass: fine FEA mesh (left), fine IGA mesh (right)

1.3 Isogeometric analysis

The compatibility between present CAD and FEM formulas is unreachable, since the two technologies evolved in different ways. A model is created for CAD representation and then a different one is built for the FEM solution. After the mesh is generated and the analyst obtains the first results, he informs the designer of the appropriate changes in the geometry. The designer updates the CAD model and gives it back to the engineer, who has to regenerate the FEM model and a new mesh all over again. This constant interaction between designer and engineer can increase dramatically the whole time of the procedure. In particular, when complex designs are concerned, where each design constitutes of numerous CAD entities combined together, the integration process is estimated to take up at around 80% of the whole analysis time.

Of course, the automatic CAD-CAE communication troubled the research community for a very long time. However, Thomas J.R. Hughes managed to approach the issue from a different perspective. He realized that instead of trying to connect current CAD and CAE formulas, they should be reinvented in ways that enable the integration. The basic idea is to use the same smooth and higher order basis functions both for the representation of the exact CAD geometry and for the approximation of the FEA

solution fields. Isogeometric analysis extends, in essence, isoparametric elements, but the process of altering geometry for the sake of the solution approximation is reversed.

IGA exhibits increased accuracy and robustness on a per degree of freedom basis in comparison to standard finite element methods. In FEM, classical and hierarchical Lagrange basis functions are usually met. Every function can be represented as a linear combination of the standard Lagrange basis as well as by a set of hierarchical basis functions, where the higher order basis contain all lower order functions. FEM's functions are defined exclusively in the interior of the element and achieve C^0 continuity at the inter-element boundary. Within the isoparametric context C^0 applies to both the geometry and the unknown displacement field. IGA invests on new technologies, such as NURBS or T-splines. The corresponding basis are defined globally on a patch and each function has support on more than one elements, enabling higher continuity and interconnectivity. Thus, a much richer behavior is obtained, which represents better the natural response of the object.

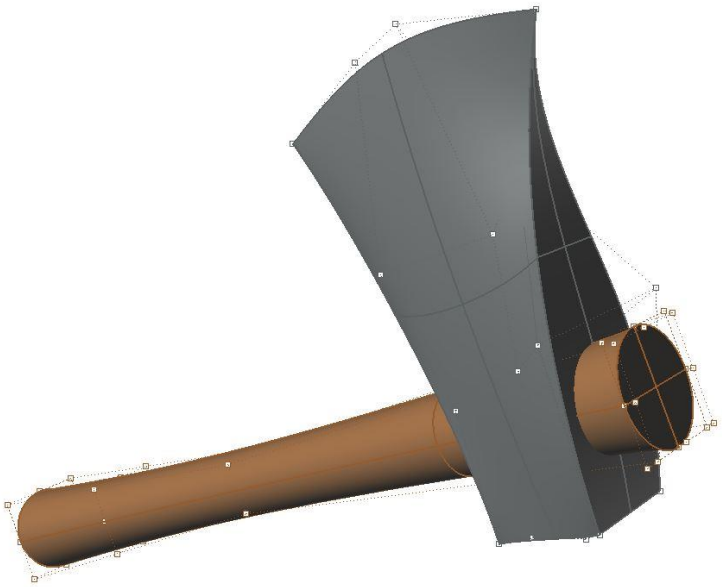


Fig. 1.9. NURBS object consisting of two patches for the two different materials (wood and steel)

IGA occupies the exact geometry model, hence it ensures accurate results even when coarse meshes are used. As far as mesh refinement is concerned, it simplifies the procedure because the geometry is fixed at the coarsest level of refinement and is unchanged throughout the process. This eliminates geometrical errors and the necessity of linking the refinement procedure to a CAD representation of the geometry, as in classical FEA. Both order elevation and knot insertion or even a combination of these techniques are supported by IGA. Isogeometric method also appeals to hierarchical structures, since they can easily be developed straight from the geometrical model.

Without a doubt IGA outperforms FEM in terms of accuracy. However, the computational cost of the new method is markedly increased. The first suspect to blame for this reduced efficiency is numerical quadrature. IGA inherited all the techniques developed within the FEA framework, however not all of them suit to the needs of the new method. In particular, FEA adopts the Galerkin method in order to solve the partial differential equations. Galerkin formulation leads to integrals, which are evaluated by numerical quadrature rules. Gauss rules are optimal for one-dimensional polynomials and have been used extensively for quadrilateral and hexahedral elements. As a result, the application of Gauss quadrature to the elements of isogeometric analysis was the prevailing option. Nevertheless, it turned out that the higher order NURBS and B-splines with the smooth properties across element boundaries and the overlapping of the corresponding basis functions increase by orders of magnitude the analysis cost.

It is clear that alternative formulations which will fit more appropriately to the concept of IGA have to be called on during the analysis process. In this thesis the method of collocation is examined and compared to the Galerkin performance. Collocation seems to be really promising as far as computational cost is concerned, however the choice between the two methods is not obvious, considering that collocation deals with some accuracy and application issues.

2 NURBS-Based Isogeometric Analysis

2.1 Introduction

The basic concept of isogeometric analysis is to extract the mesh for analysis purposes straight from the model's design. Classical FEA and the previous methods suggest that an approximate mesh is created instead of taking advantage of the existing accurate one. In cases that an error has to be corrected or a modification to take place, the intervention to the initial mesh is not an option. The standard procedure implies that a new mesh should be formed over again each time that an alteration is to be imposed. This way, the analysis process becomes time-consuming, while it also deprives of a high level of flexibility and accuracy, considering that the additional geometry error has to be counted too. On the contrary, IGA always works with the accurate geometry grid, thus the geometry error is eliminated. This observation seems now very obvious, but it took years of research until 2003, when Thomas J.R. Hughes and his research team succeeded to cut the gordian knot of CAD–CAE integration.

The node mesh is divided into two separate meshes, which derive directly from the geometrical representation, that are the control point mesh and the knot mesh.

- The control point mesh defines the geometry, it does not conform though to the structure. It consists of multilinear elements depending on the dimensions of the problem. The degrees of freedom, called as control variables, lie at the control points and they are the unknowns of the final linear system of equations. The control point mesh can "protect" and preserve the physical geometry, even when it is highly distorted, unlike a typical finite element mesh. Besides, continuity is determined by the basis functions, hence control points can be modified without influencing the curve's continuity.

- The knot mesh provides the discretization of the model, considering that knots constitute the boundaries of knot spans, or else elements. Knots are also the points where the piecewise polynomial basis functions change formula. Thus, numerical integration is conveniently accomplished at a knot span level. Apart

from that, the knot mesh is additionally used to define the support of each shape function.

The abstract framework and the numerical applications which are developed in this thesis, are based exclusively on Non-Uniform Rational B-splines (NURBS), as they are so far the most commonly used design tool. They are quite simple in their definition and they are able to represent with accuracy smooth curves and all conic sections. Of course more advanced technologies have emerged in the field of computational geometry in the last years, such as T-splines and subdivision surfaces. These models reduce the required number of control points and as a result they respond better to complicated geometries, while they accommodate local refinement techniques. Nevertheless, the efficiency of NURBS is undoubted and considering that they have been adopted in a wide range of applications, they still have a significant lead.

2.2 Index, parameter and physical space

Physical space hosts accurate geometrical representations of the natural model, designed in the familiar Cartesian system. However, in order to accurately simulate complex models with curved boundaries, the isoparametric concept, which is adopted in both FEA and IGA, introduces an imaginary, basic space, where all geometries can be represented as lines, rectangles and cuboids. This is parameter space. Finally, there exist index space too, which may only play an auxiliary role for NURBS-based isogeometric analysis, it is essential though for other kinds of splines.

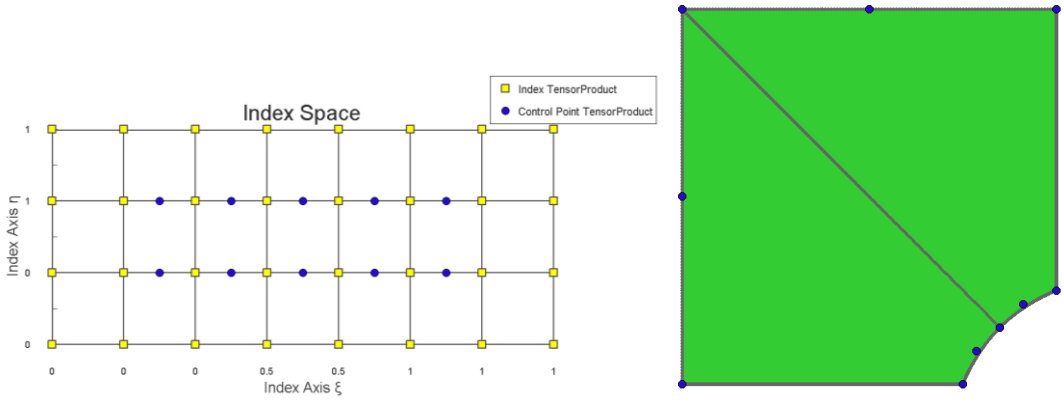


Fig. 2.1. B-spline solid: **(a)** index space, **(b)** physical space

2.2.1 Index space

In index space each knot is uniquely identified, even if it has a multiplicity greater than one. Knot values are located at equally spaced positions, so that axis calibration refers to their sequence rather their numerical content. Index space is a key to understanding NURBS, since it helps to identify which knot value spans are influenced by each basis function and thus apprehend support and overlapping issues. Control points are also depicted in index space at the center of the support of basis functions.

Expansion to 2D or 3D leads to the creation of rectangles or cuboids respectively. Due to tensor product properties, everything mentioned about 1D extends and applies to both 2D and 3D. Therefore, index space provides information that can contribute to the handling of a complex representation.

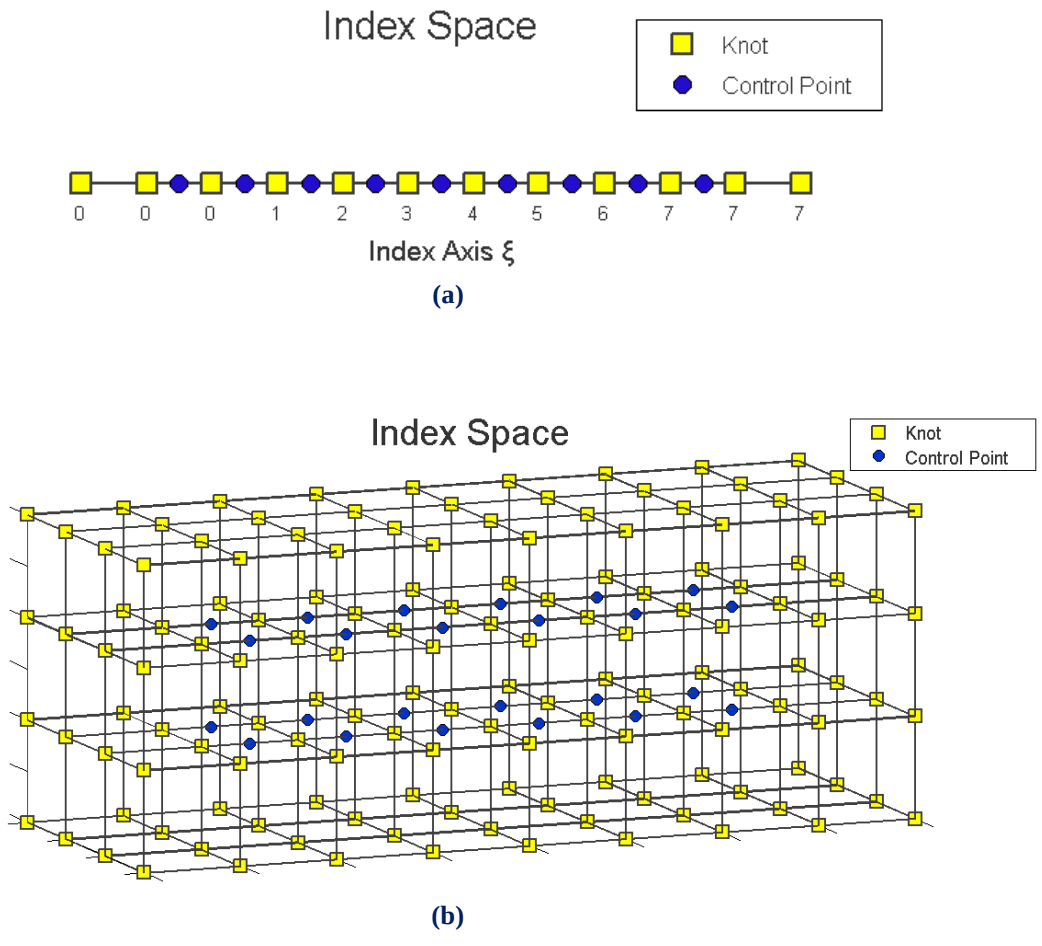
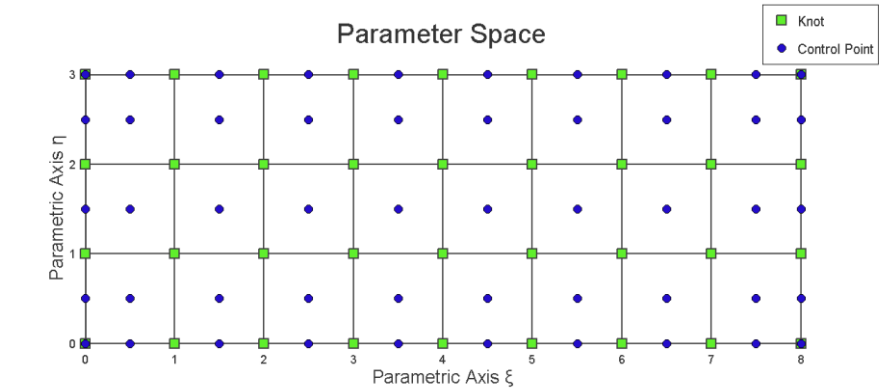


Fig. 2.2. (a) Curve and **(b)** solid represented in index space

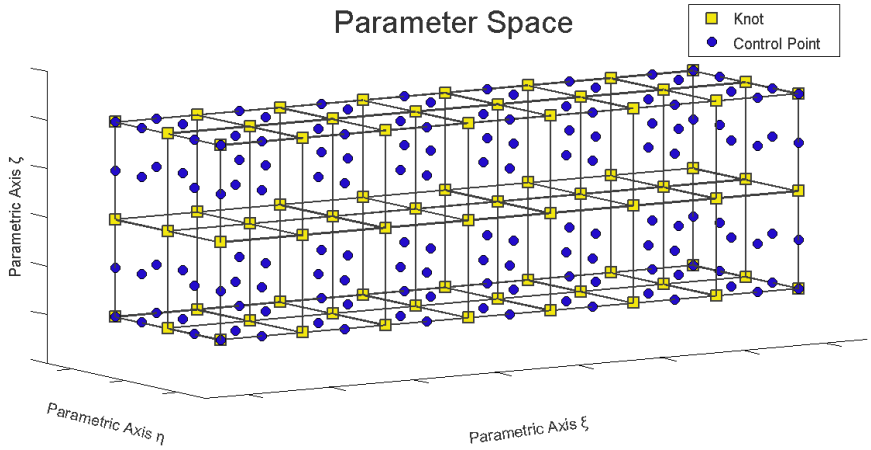
2.2.2 Parameter space

Parameter space is a representation of the model with respects to knots, which takes place at the parametric coordinate system (ξ,η,ζ). The computation of the entries of the stiffness matrix requires the mapping of the Cartesian system (x,y,z) to the parametric system. The transfer between the two spaces is achieved through the Jacobian matrix and its inverse. In parameter space, all spline entities, regardless the complexity of their pattern, are represented as orthogonal shapes.

The illustration of B-spline basis functions in parameter space reveals that they are non-zero only in specific knot spans, despite the fact they are defined throughout the space. These knot spans constitute the domain of each basis function. Each knot span belongs to the domain of more than one functions, provided that each knot denotes the start of one domain and the end of another. The highly localized phenomenon of overlapping points out which functions control a common part of the structure in physical space, while it is reflected at the density of the stiffness matrix.



(a)



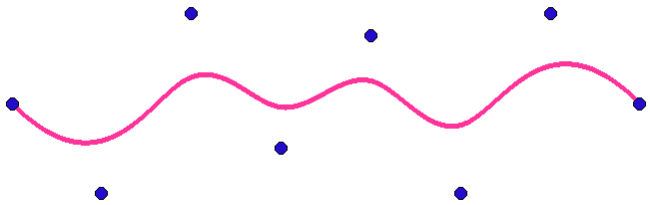
(b)

Fig. 2.3. (a) Surface and **(b)** solid represented in parameter space

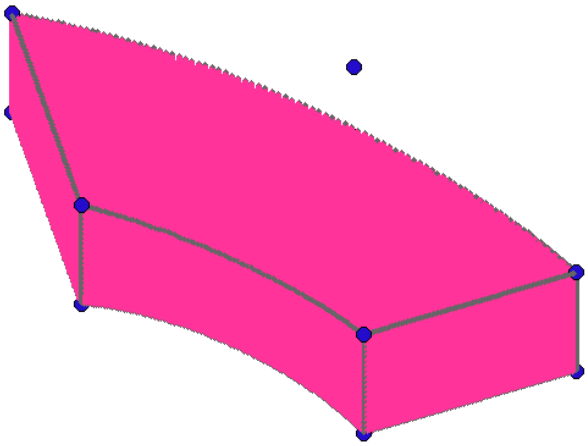
2.2.3 Physical space

Physical space is where the actual geometry is represented by a linear combination of the basis functions and the control points. The fact that the control points do not belong to the model in Cartesian space, but they suitably surround it, is the reason why NURBS and spline entities achieve accurate representations. Of course the contribution of B-spline basis functions is crucial too. For a given set of control points, the final geometry is determined by the choice of the set of basis functions. In contrast to finite element functions, in NURBS the basis functions are usually not interpolating the control points.

In physical mesh the actual geometry is decomposed into elements either with the approach of a patch or with the approach of a knot span. The predominant strategy is to consider as patch a subdomain that consists of many knot spans. Indeed, it is more convenient to attribute the role of element to knot spans, since the functions are C^{∞} within knot spans and their continuity is reduced only along knot lines where different "pieces" join. In addition, many complicated domains can be exactly represented by a single patch, as well as all the geometries which will be considered in this thesis. This aspect cannot conform to our familiar perception of an element.



(a)



(b)

Fig. 2.4. (a) Curve and **(b)** solid represented in physical space

2.3 B-splines

2.3.1 Knot vector

A B-spline consists of *n* polynomial basis functions of degree *p*. In order to acquire these functions, the corresponding knot vector should first be defined. A knot vector is defined in bibliography as a non-decreasing set of parametric coordinates $\Xi = \left\{\xi_1, \xi_2, ..., \xi_{n+p+1}\right\}$, where $\xi_i \in \mathbb{R}$ is the i^{th} knot value. It can contain the same number multiple times and generates the basis in a unique way. To prevent any potential confusion, let us assume that the "knot value vector" comprises the whole set of non-decreasing values, while the "knot vector" includes the set of unique coordinates. For example, a knot value vector could be {0 1 1 1 2 2 3 3} where 0, 1, 1, 1, 2, 2, 3, 3, are the separate knot values. The corresponding knot vector is {0 1 2 3} where 0, 1, 2, 3 are the separate knots.

If the knot values are equally spaced in parameter space, the knot value vector is considered uniform, or else it is non-uniform. If the first and the last knot are repeated *p*+1 times, the knot value vector is said to be open, as it forces the functions to have C^{-1} continuity at the edges, creating this way an open curve that is interpolatory at the corresponding knots. In general, the basis functions are not interpolating the interior knots.

A knot value vector may contain integers or decimals. In fact, the actual numerical content of knot values is of no importance. What matters is the relative distance between them. This means a knot value vector can be multiplied by any number, or have a number added to every knot value and the resulting basis would still be the same.

As it has already been mentioned, control points are located at the center of the support in index space. Provided that for the i^{th} basis function of order *p* the support refers to the interval $\left[\xi_i, \xi_{i+p+1}\right)$, it is obvious that the support of each basis function includes *p*+1 knot value spans, therefore *p*+2 knot values. For even degrees the center of the support lies between the sequential knot values $i + \frac{p}{2}$ and $i + \frac{p}{2} + 1$, while for odd degrees it is coincident with the knot value $i + \frac{p+1}{2}$. Consequently, control points in index space are located directly at a knot for odd *p* and either at a knot or in the middle of a knot span for even *p*.

2.3.2 Basis functions

Given a knot value vector $\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$, the B-spline basis functions can be evaluated at $\xi \in [\xi_1, \xi_{n+p+1}]$ using the Cox-de Boor recursive formula as described in [3]. The piecewise constants are defined for p=0 by

$$N_{i,0}(\xi) = \begin{cases} 1, & \text{if } \xi_i \le \xi < \xi_{i+1} \\ 0, & \text{otherwise} \end{cases}$$

The piecewise constant does not include the right edge ξ_{i+1} in order to ensure partition of unity, as the next basis function begins from the specific edge. The last function, however, includes both left and right edge, so as to be defined across the whole knot span.

For p=1, 2, 3,… functions are defined by

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi)$$

If the denominators of the above factors are equal to zero, the whole factor is set to zero.

B-spline basis functions comply with the nature of full tensor product, in order to form multidimensional B-spline shape functions. Therefore, the evaluations of basis functions in each parametric direction are associated through a linear combination producing curves, surfaces and solids in $\mathbb{R}^d$ space. The properties of the final geometries follow directly from the properties of their basis functions.

2D B-spline shape functions can be evaluated as tensor product of basis functions $N_{i,p}(\xi)$ and $M_{j,q}(\eta)$:

$$R_{i,j}^{p,q}(\xi, \eta) = N_{i,p}(\xi) M_{j,q}(\eta)$$

3D B-spline shape functions are a tensor product of basis functions on three directions, $N_{i,p}(\xi)$, $M_{j,q}(\eta)$ and $L_{k,r}(\zeta)$:

$$R_{i,j,k}^{p,q,r}(\xi, \eta, \zeta) = N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta)$$

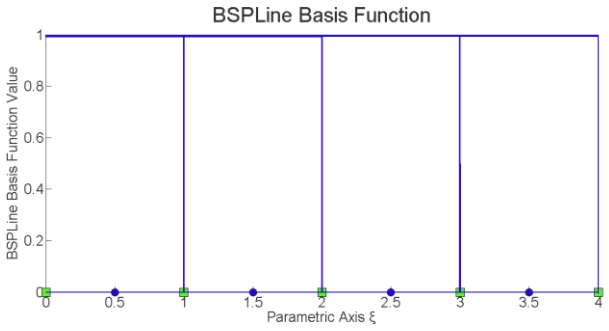
2.3.3 Properties

B-spline basis functions possess some important properties, as they are described in [3]. Most of them follow by induction from the recursive relation and they constitute the theoretical background of NURBS-based isogeometric analysis.

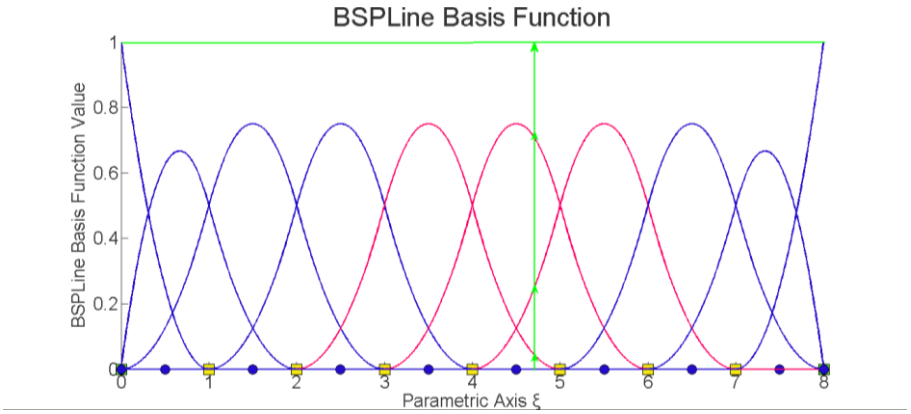
1. Partition of unity

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1 \quad \forall \xi, p \,.$$

It is obvious in following figures that the basis forms a partition of unity for every point ξ and every polynomial degree p. In other words the sum of B-spline values at any parametric coordinate is equal to 1 regardless the discretization order. It should be noted that in Fig. 2.5a, where p=0, the value at internal knots is equal to zero when the left knot span is concerned, and equal to one when the right knot span is concerned. This is imposed by the open interval $\xi \in \left[\xi_i, \xi_{i+1}\right)$, as it was underlined in the previous section.



(a)



(b)

Fig. 2.5. Sum of B-spline basis function value for every ξ

(a) $\Xi = \{0\ 1\ 2\ 3\ 4\}$

(b) $\Xi = \{0\ 0\ 0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 8\ 8\}$

2. Local support

Each B-spline basis function has a local support of p+1 knot value spans. This property arises from the recursive character of the Cox-de Boor formula. Functions are built gradually and hierarchically beginning from p=0 to the desired polynomial degree. Each box function has a support of one knot value span. Therefore it is convenient to establish the number of box functions required as a measure of the local support.

In order to form a function of order p, two consecutive functions of order p-1 are required. Each one of them uses two consecutive functions of order p-2. However, the two functions of p-1 share one of these subordinate functions, making a total of three functions of order equal to p-2. This procedure involves p steps until the zero degree is reached. At each step, where the degree is gradually reduced, the number of functions is increased by one. Apparently, the computations lead to p+1 box functions used to form the final function of degree p. This thought process is illustrated in the figure below.

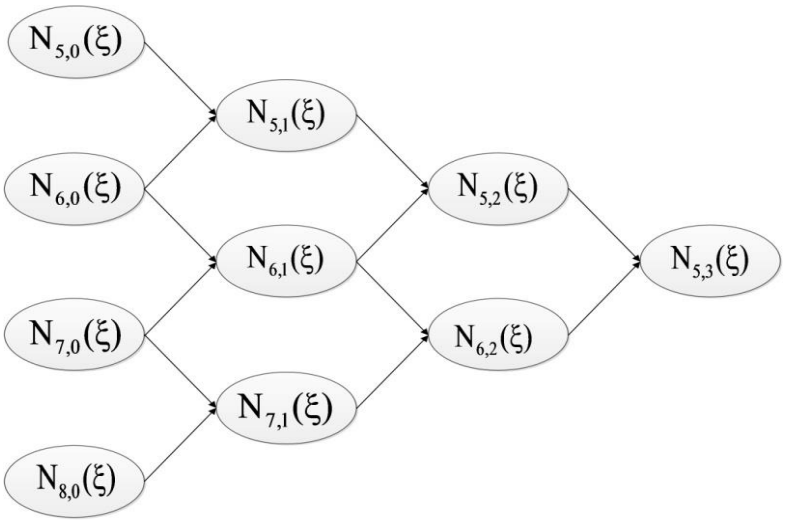


Fig. 2.6. Lower-order basis functions required for the creation of $N_{5,3}(\xi)$

In Fig. 2.7 the box functions, drawn in red, are required in order to build the linear basis functions, drawn in green. That means that $N_{4,1}$ and $N_{5,1}$ are evaluated from $N_{4,0}$, $N_{5,0}$ and $N_{5,0}$, $N_{6,0}$ respectively. Then the linear functions $N_{4,1}$ and $N_{5,1}$ are combined for the evaluation of the quadratic $N_{4,2}$. The support of $N_{4,2}$ is extended to the interval $[1,4)$, including the p+1=3 knot spans where the involved box functions are non-zero.

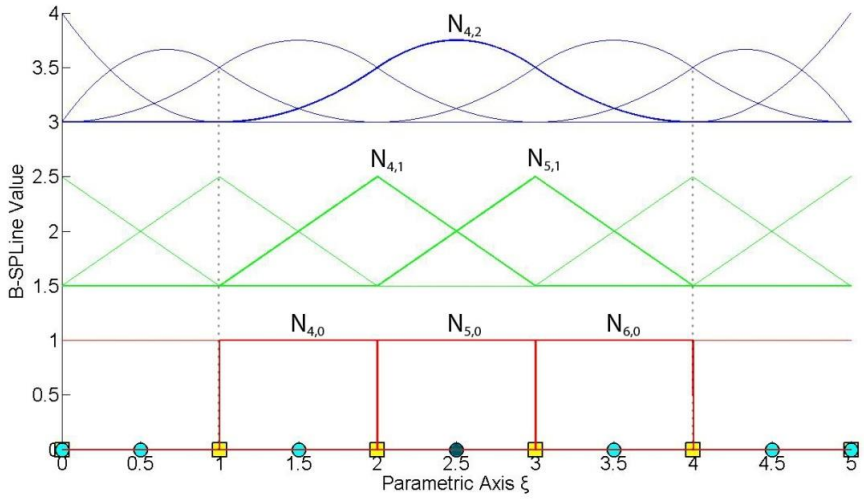


Fig. 2.7. B-spline recursive character:
Basis functions required for the evaluation of a quadratic B-spline function

It should be noted that trivial spans, which are formed when the knot values are repeated, are also taken into consideration. Fig. 2.8 presents the support of three consecutive basis functions emanating from the knot value vector $\Xi = \{0\ 0\ 0\ 1\ 2\ 3\ 4\ 5\ 5\ 5\}$. All the three of them have a support of p+1=2+1=3 knot value spans. However, only the third is fully developed, since the first two are related to the span $[0,0)$.

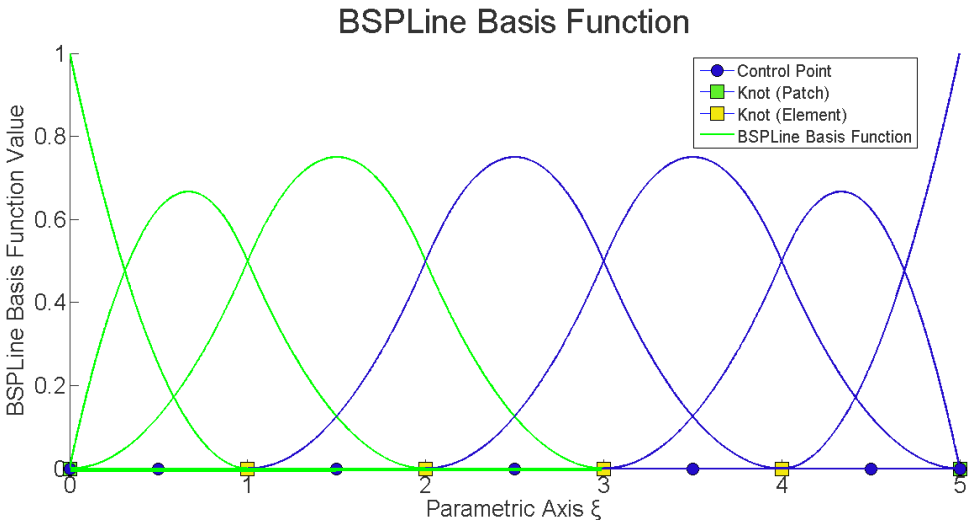


Fig. 2.8. Support of trivial knot value spans

Support of first basis function: $(0,0),\ (0,0),\ (0,1)$

Support of second basis function: $(0,0),\ (0,1),\ (1,2)$

Support of third basis function: $(0,1),\ (1,2),\ (2,3)$

The full tensor product property sets the local support of a two-dimensional shape function to $(p+1)(q+1)$ knot value spans, where *p* and *q* are the discretization orders in the parametric axes ksi and heta respectively. In Fig. 2.9 the 2D shape function $R^{2,2}_{5,5}(\xi,\eta)$ is formed by the evaluations of $N_{5,2}(\xi)$ and $M_{5,2}(\eta)$. The value of the bidirectional B-spline is represented in the third axis of the graph and by projection of the contour in the 2D plane. Both one-dimensional functions are quadratic, hence their support is extended to *p*+1=2+1=3 knot value spans. It was thus expected that the corresponding shape function would affect the domain defined by the combination of these knot spans lying across each direction. The support occupies a total area of $3\times3=9$ knot value rectangles.

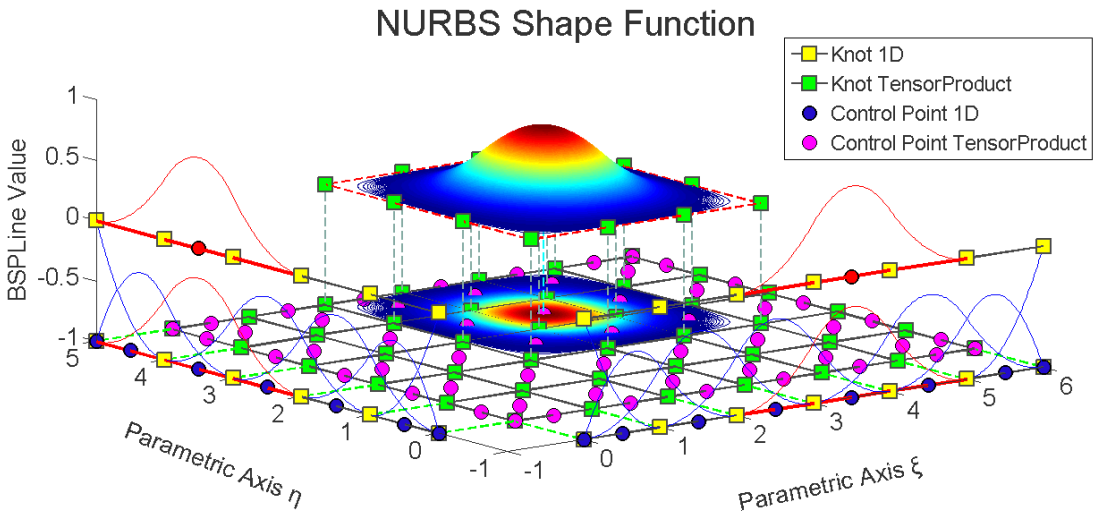


Fig. 2.9. Shape function $R^{p,q}_{i,j} = R^{2,2}_{5,5}(\xi,\eta)$ as a tensor product of $N_{5,2}(\xi)$ and $M_{5,2}(\eta)$
Ξ={0 0 0 1 2 3 4 5 6 6 6}
H={0 0 0 1 2 3 4 5 5 5}

3D B-spline shape functions can be represented as in Fig. 2.10. Shape functions for ξ are drawn in the ξ-η plane, functions for η in the η-ζ plane and functions for ζ in the ζ-ξ plane. Their tensor product value has been calculated as a function of two parametric directions at the control point coordinate of the remaining direction. The resulting contour is projected on a plane that is parallel to the two directions and intersects with the tensor product control point. This process is repeated for all three possible combinations, thus creating contours at all three planes.

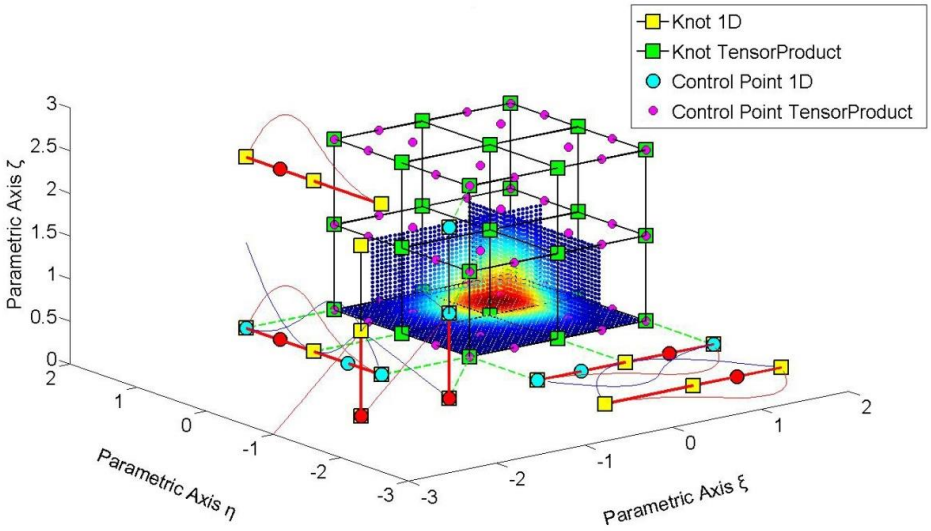


Fig. 2.10. Shape function $R^{p,q,r}_{i,j,k} = R^{2,2,1}_{3,3,1}(\xi,\eta,\zeta)$ as tensor product of $N_{3,2}(\xi)$, $M_{3,2}(\eta)$, $L_{1,1}(\zeta)$

Ξ={0 0 0 1 2 2 2}

H={0 0 0 1 2 2 2}

Z={0 0 1 2 2}

3. Non-zero functions per knot span

The maximum number of basis functions that are non-zero across a knot span is p+1. This outcome flows from the Cox-de Boor recursive formula as well and addresses the "decomposition" process of basis functions of order p. In particular, one box function that is non-zero over a specific knot span participates to the evaluation of two consecutive functions of order p=1. These linear functions contribute to the creation of three consecutive basis functions of order p=2. In general, one box function is involved in the formation of p+1 functions of order p.

At this point it should be underlined that over a knot span only one box function has non-zero values. Therefore, the p^{th} degree basis functions that are non-zero at a specific knot span are exclusively the p+1 functions which are built based on the corresponding box function. Specifically, at a non-trivial knot value span $[\xi_i, \xi_{i+1})$ only the basis functions $N_{i-p,p}(\xi)$, $N_{i-p+1,p}(\xi)$, ..., $N_{i,p}(\xi)$ are non-zero. This property is used efficiently in stiffness matrix formulation, in order to reduce the computational cost.

The maximum number of shape functions that are non-zero across a two-dimensional or three-dimensional knot span is $(p+1)(q+1)$ and $(p+1)(q+1)(r+1)$ respectively.

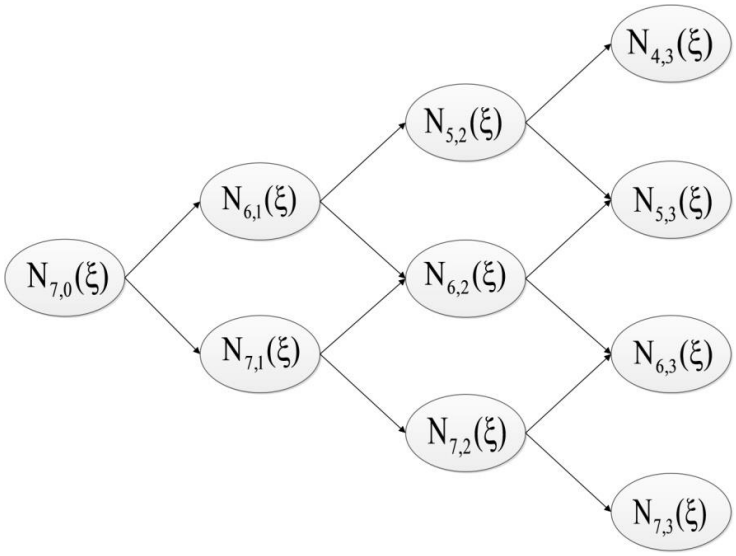


Fig. 2.11. Contribution of one box function
to the creation of higher-order B-spline basis functions

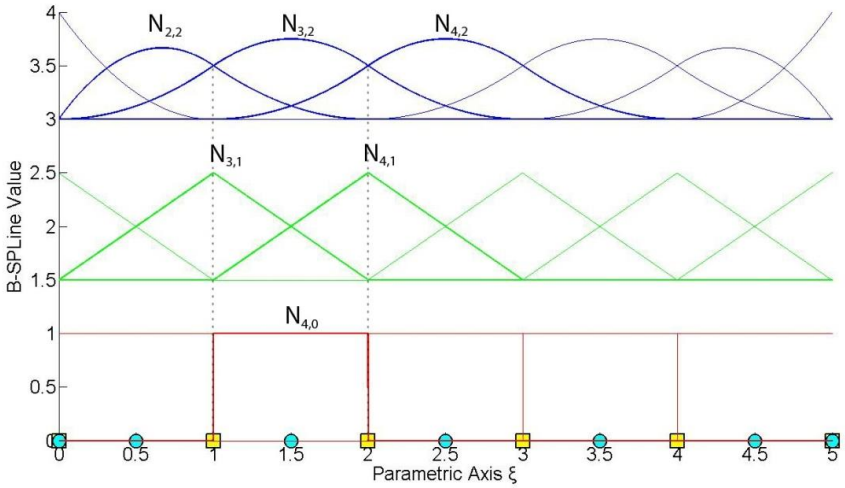


Fig. 2.12. Contribution of one box function to non-zero higher-order
B-spline basis functions across knot span [1,2)

4. Non-negativity

$$N_{i,p}(\xi) \geq 0 \quad \forall\, \xi, i, p$$

In contrast to finite element basis functions where negative values appear, B-spline basis functions are always positive or equal to zero for every ξ, p. This enables the computations to be less complicated and more numerically stable.

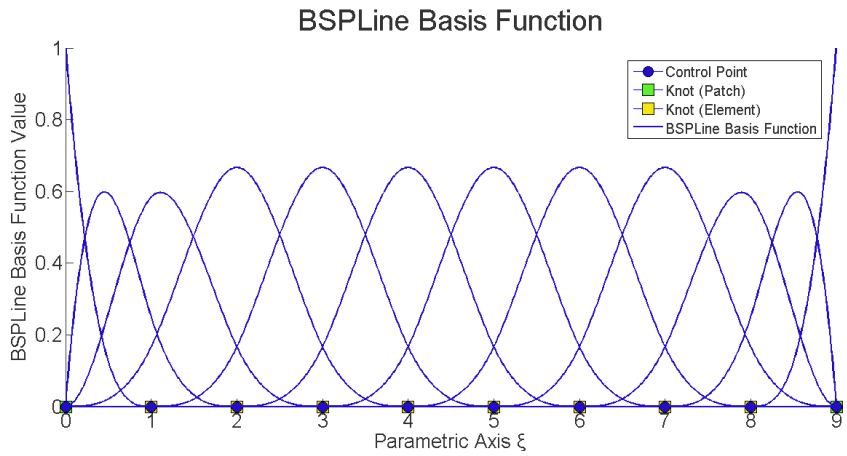


Fig. 2.13. B-spline basis functions for knot value vector
Ξ={0 0 0 0 1 2 3 4 5 6 7 8 9 9 9 9}

5. Shared support

Each B-spline basis function shares support with 2*p* functions, which is *p* on each side. This leads to a bandwidth equal to 2*p*+1 (including the function in question). Surprisingly, the increasing support of B-splines compared to standard finite element basis functions does not result in increased bandwidths in a numerical method. The shared support of 2*p*+1 corresponds to both methods as well. The difference between IGA and FEA lies at the density of the stiffness matrix. The overlapping in FEA is caused by corner or edge nodes, which are involved in more than one elements. Internal nodes affect exclusively their own elements. In IGA on the contrary, each control point associates with *p*+1 knot spans (elements), thus more positions in the stiffness matrix are occupied and the density increases markedly with respect to *p*, especially in multi-dimensional cases.

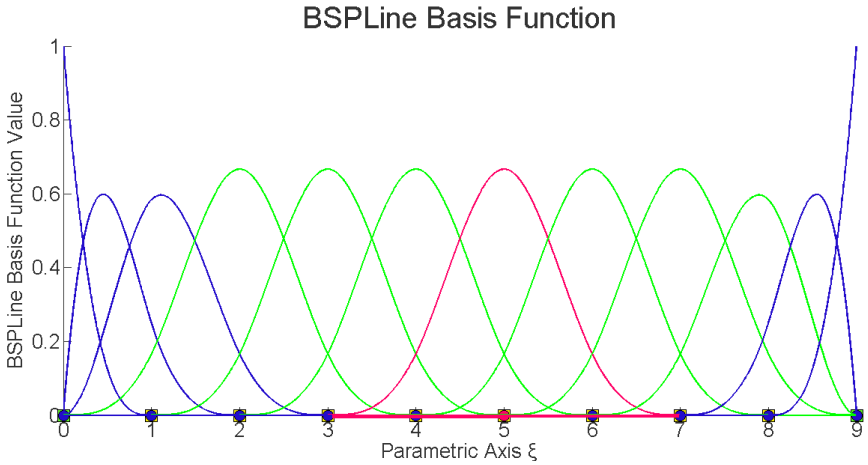


Fig. 2.14. Shared support for $N_{7,3}(\xi)$
Ξ={0 0 0 0 1 2 3 4 5 6 7 8 9 9 9 9}

$N_{7,3}(\xi)$ interacts with p=3 basis functions on each side, $N_{4,3}(\xi)$, $N_{5,3}(\xi)$ and $N_{6,3}(\xi)$ on the left, $N_{8,3}(\xi)$, $N_{9,3}(\xi)$, $N_{10,3}(\xi)$ on the right. As a result, the respective positions in the stiffness matrix will be non-zero.

6. C^{p-m} Continuity

B-spline basis functions have C^{p-m} continuity across a knot with multiplicity m. That means that p-m continuous derivatives can be defined over the knot in question. In the interior of knot spans, where the basis functions are smooth polynomials, the continuity is always C^{∞}. As the level of C decreases, B-splines tend to be steeper. Indeed, the reduced continuity indicates multiple repetitions of knots, therefore the corresponding functions cannot be fully developed, since their domain includes trivial spans. When the multiplicity of a knot value is exactly p, the basis is interpolatory at that knot. Knots with multiplicity p+1 are responsible for discontinuities and they take the role of patch boundaries. In the patch interior, continuity less than C^0 is not acceptable, so the internal knots can be repeated up to p times.

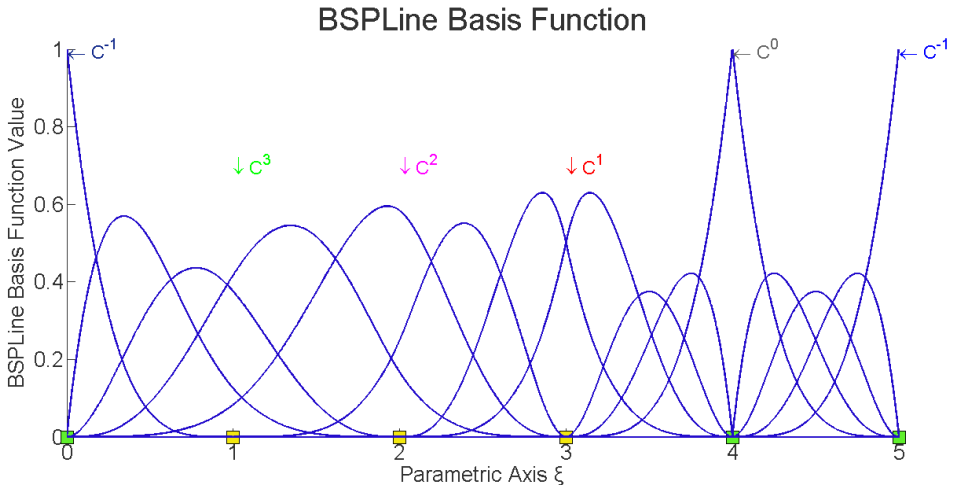
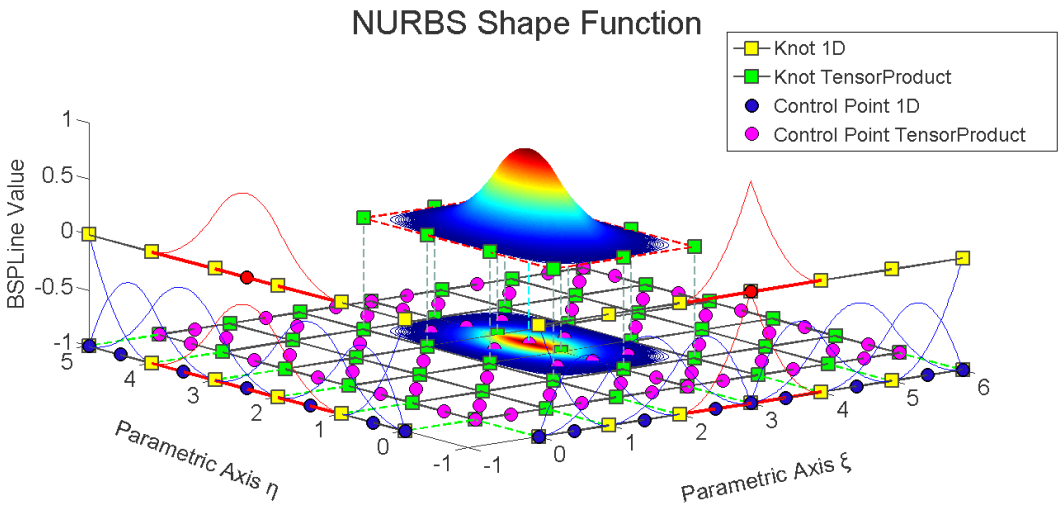


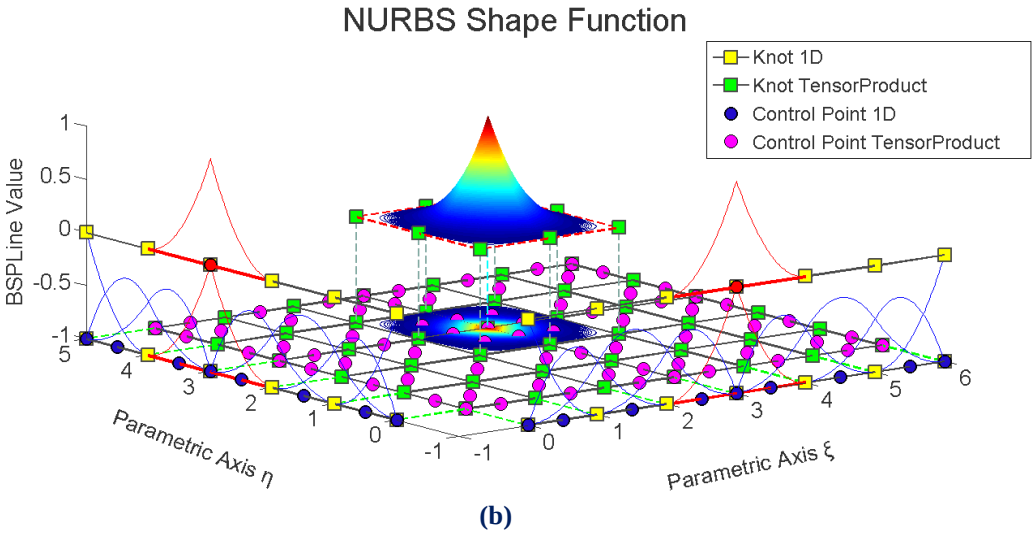
Fig. 2.15. Continuity across interior element boundaries
Ξ={0 0 0 0 0 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5}

In 2D and 3D cases, continuity at each direction is obtained straight from the continuity of the corresponding one-directional B-spline basis functions. In Fig. 2.16a, $R^{2,2}_{5,4}(\xi,\eta)$ is tensor product of $N_{5,2}(\xi)$ and $M_{4,2}(\eta)$. $N_{5,2}(\xi)$ has $C^{p-m}=C^{2-2}=C^0$ continuity across ξ=3 and $M_{4,2}(\eta)$ has $C^{p-m}=C^{2-1}=C^1$ continuity across η=2. Therefore, $R^{2,2}_{5,4}(\xi,\eta)$ has C^0 continuity with respect to ξ and C^1 continuity with respect to η across $(\xi,\eta)=(3,2)$.

In Fig. 2.16b, $R^{2,2}_{5,5}(\xi,\eta)$ is tensor product of $N_{5,2}(\xi)$ and $M_{5,2}(\eta)$. Both B-spline basis functions have C^0 continuity across ξ=3, η=3 respectively. As a result, $R^{2,2}_{5,5}(\xi,\eta)$ has C^0 continuity across $(\xi,\eta)=(3,3)$. In general, in multi-dimensional functions C^0, continuity across a knot requires the basis functions per all directions to be C^0 continuous at that point.



(a)



(b)

Fig. 2.16. NURBS shape functions for directions ξ, η

(a) Shape Function $R^{p,q}_{i,j} = R^{2,2}_{5,4}(\xi,\eta)$ as tensor product of $N_{5,2}(\xi)$ and $M_{4,2}(\eta)$

Ξ={0 0 0 1 2 3 3 4 5 6 6 6}

H={0 0 0 1 2 3 4 5 5 5}

(b) Shape Function $R^{p,q}_{i,j} = R^{2,2}_{5,5}(\xi,\eta)$ as tensor product of $N_{5,2}(\xi)$ and $M_{5,2}(\eta)$

Ξ={0 0 0 1 2 3 3 4 5 6 6 6}

H={0 0 0 1 2 3 3 4 5 5 5}

7. Number of knot values

The evaluation of n basis functions of polynomial order p requires $n+p+1$ knot values. According to the property pertaining to the local support, each control point has a support of $p+1$ knot value spans, which is $p+2$ knot values. n control points lead to the sum of $n(p+1)$ knot values. However one knot value belongs to the support of more than one control points. There are $p+1$ knot values repeated in $n-1$ control point interconnections. Therefore, the total number of knot values is:

$$n(p+2)-(n-1)(p+1)=np+2n-np-n+p+1=n+p+1$$

2.3.4 Basis function derivatives

The definition of B-spline basis functions derivatives is of great importance for isogeometric analysis. The values of derivatives calculated at Gauss points form the entries of the deformation and the stiffness matrix, thus they determine the displacement, stress and strain fields. Given a polynomial order p and a knot value vector $\Xi=\{\xi_1,\xi_2,...,\xi_{n+p+1}\}$, the derivative of the i^{th} basis function is represented as a linear combination of basis functions of previous polynomial order [1]:

$$\frac{d}{d\xi}N_{i,p}(\xi)=\frac{p}{\xi_{i+p}-\xi_i}N_{i,p-1}(\xi)-\frac{p}{\xi_{i+p+1}-\xi_{i+1}}N_{i+1,p-1}(\xi)$$

This equation can be generalized for the k^{th} derivative as follows:

$$\frac{d^k}{d\xi^k}N_{i,p}(\xi)=\frac{p!}{(p-k)!}\sum_{j=0}^{k}a_{k,j}N_{i+j,p-k}(\xi)$$

$$a_{0,0}=1$$

$$a_{k,0}=\frac{a_{k-1,0}}{\xi_{i+p-k+1}-\xi_i}$$

$$a_{k,j}=\frac{a_{k-1,j}-a_{k-1,j-1}}{\xi_{i+p+j-k+1}-\xi_{i+j}},\qquad j=1,\ldots,k-1,$$

$$a_{k,k}=\frac{-a_{k-1,k-1}}{\xi_{i+p+1}-\xi_{i+k}}$$

Two-dimensional shape functions give the following partial derivatives:

$$\frac{\partial}{\partial\xi}R_{i,j}^{p,q}(\xi,\eta)=\left(\frac{d}{d\xi}N_{i,p}(\xi)\right)M_{j,q}(\eta)$$

$$\frac{\partial}{\partial\eta}\mathrm{R}_{\mathrm{i,j}}^{\mathrm{p,q}}\left(\xi,\eta\right)=\mathrm{N}_{\mathrm{i,p}}(\xi)\left(\frac{\mathrm{d}}{\mathrm{d}\eta}\mathrm{M}_{\mathrm{j,q}}(\eta)\right)$$

3D shape function derivatives per direction can be obtained in like manner:

$$\frac{\partial}{\partial\xi}\mathrm{R}_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta)=\left(\frac{\mathrm{d}}{\mathrm{d}\xi}\mathrm{N}_{\mathrm{i,p}}(\xi)\right)\mathrm{M}_{\mathrm{j,q}}\left(\eta\right)\mathrm{L}_{\mathrm{k,r}}\left(\zeta\right)$$

$$\frac{\partial}{\partial\eta}\mathrm{R}_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta)=\mathrm{N}_{\mathrm{i,p}}(\xi)\left(\frac{\mathrm{d}}{\mathrm{d}\eta}\mathrm{M}_{\mathrm{j,q}}\left(\eta\right)\right)\mathrm{L}_{\mathrm{k,r}}\left(\zeta\right)$$

$$\frac{\partial}{\partial\zeta}\mathrm{R}_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta)=\mathrm{N}_{\mathrm{i,p}}(\xi)\mathrm{M}_{\mathrm{j,q}}\left(\eta\right)\left(\frac{\mathrm{d}}{\mathrm{d}\zeta}\mathrm{L}_{\mathrm{k,r}}\left(\zeta\right)\right)$$

2.3.5 B-spline curves

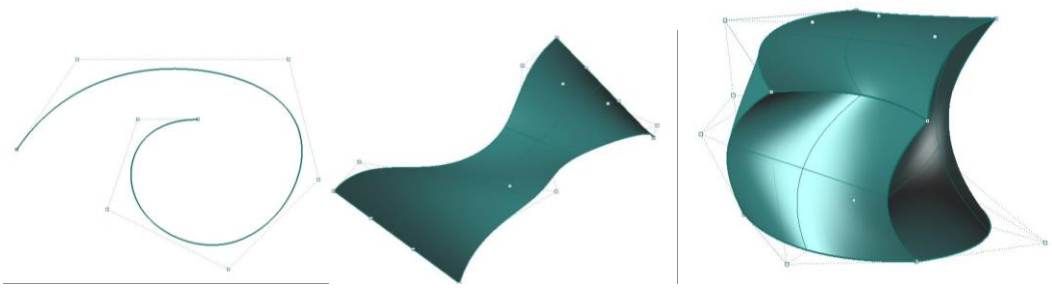
B-spline curves constitute a linear combination of B-spline basis functions and each one of them is related to a set of coordinates, the so-called control points $\mathbf{X_i}=\{\mathrm{X_i},\mathrm{Y_i},\mathrm{Z_i}\}$. Given *n* basis functions and the coefficients $\mathbf{X_i}$ of the corresponding control points, the B-spline curve is evaluated at every ξ as follows:

$$\mathrm{C}(\xi)=\sum_{\mathrm{i=1}}^{\mathrm{n}}\left\{\mathrm{N}_{\mathrm{i,p}}(\xi)\mathrm{X}_{\mathrm{i}}\right\}$$

As far as the two-dimensional B-spline surfaces and the three-dimensional B-spline solids are concerned, the procedure is analogous to the curve case respectively:

$$\mathrm{S}(\xi,\eta)=\sum_{\mathrm{i=1}}^{\mathrm{n}}\sum_{\mathrm{j=1}}^{\mathrm{m}}\left\{\mathrm{N}_{\mathrm{i,p}}(\xi)\mathrm{M}_{\mathrm{j,q}}(\eta)\mathrm{X}_{\mathrm{i,j}}\right\}=\sum_{\mathrm{i=1}}^{\mathrm{n}}\sum_{\mathrm{j=1}}^{\mathrm{m}}\left\{\mathrm{R}_{\mathrm{i,j}}^{\mathrm{p,q}}\mathrm{X}_{\mathrm{i,j}}\right\}$$

$$\mathrm{S}(\xi,\eta,\zeta)=\sum_{\mathrm{i=1}}^{\mathrm{n}}\sum_{\mathrm{j=1}}^{\mathrm{m}}\sum_{\mathrm{k=1}}^{\mathrm{l}}\left\{\mathrm{N}_{\mathrm{i,p}}(\xi)\mathrm{M}_{\mathrm{j,q}}(\eta)\mathrm{L}_{\mathrm{k,q}}(\zeta)\mathrm{X}_{\mathrm{i,j,k}}\right\}=\sum_{\mathrm{i=1}}^{\mathrm{n}}\sum_{\mathrm{j=1}}^{\mathrm{m}}\sum_{\mathrm{k=1}}^{\mathrm{l}}\left\{\mathrm{R}_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}\left(\xi,\eta,\zeta\right)\mathrm{X}_{\mathrm{i,j,k}}\right\}$$



(a) **(b)** **(c)**

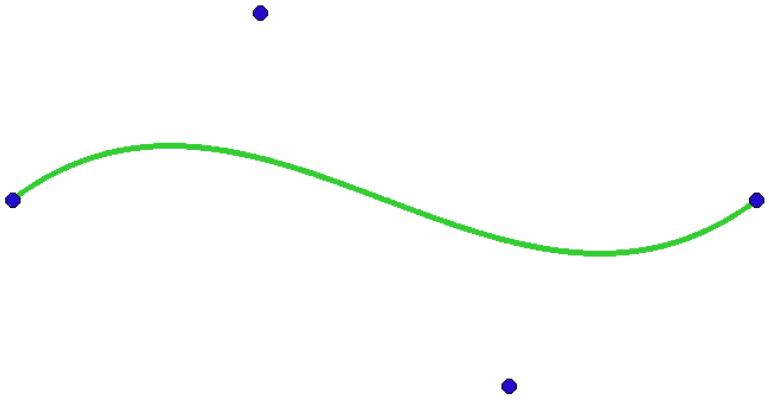
Fig. 2.17. B-spline objects: **(a)** curve, **(b)** surface, **(c)** solid

2.3.6 B-spline curve properties

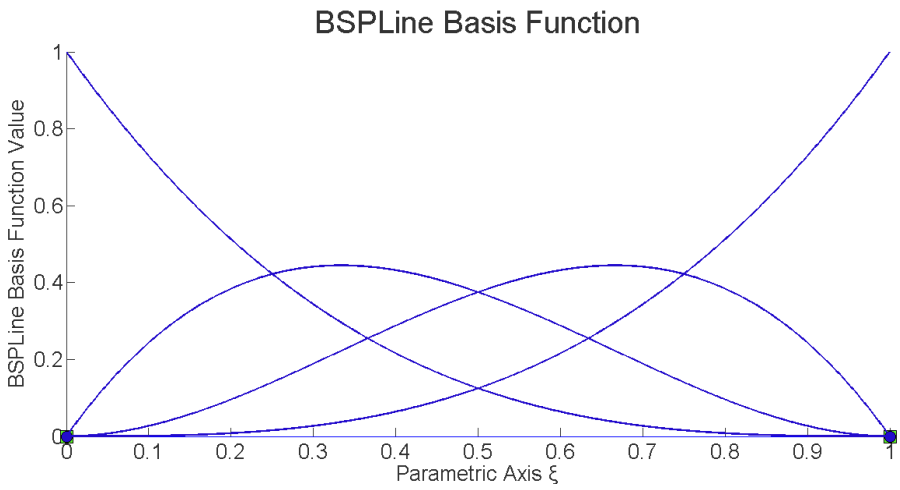
The properties of B-spline curves are described in [3]. Plenty of them are inherited by the basis functions participating to their evaluation.

1. Generalization of Bézier curves

Bézier curves were until recently widely used in computer graphics in order to represent smooth curves. Their difference with B-splines is that they are all defined across the same knot span and as a result they are non-zero across the entire domain. A possible alteration to a control point affects the whole model and not just a part of it. Fig. 2.18 presents the physical space and the basis functions of a Bézier curve based on the knot value vector {0 0 0 0 1 1 1 1}.



(a)



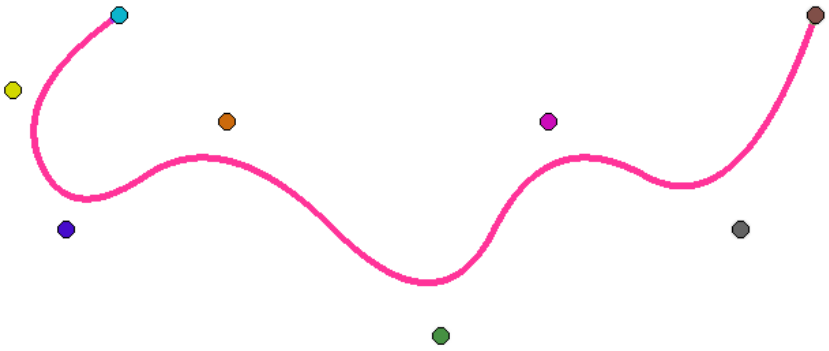
(b)

Fig. 2.18. B-spline as generalization of Bézier curves:
(a) physical space, **(b)** basis functions

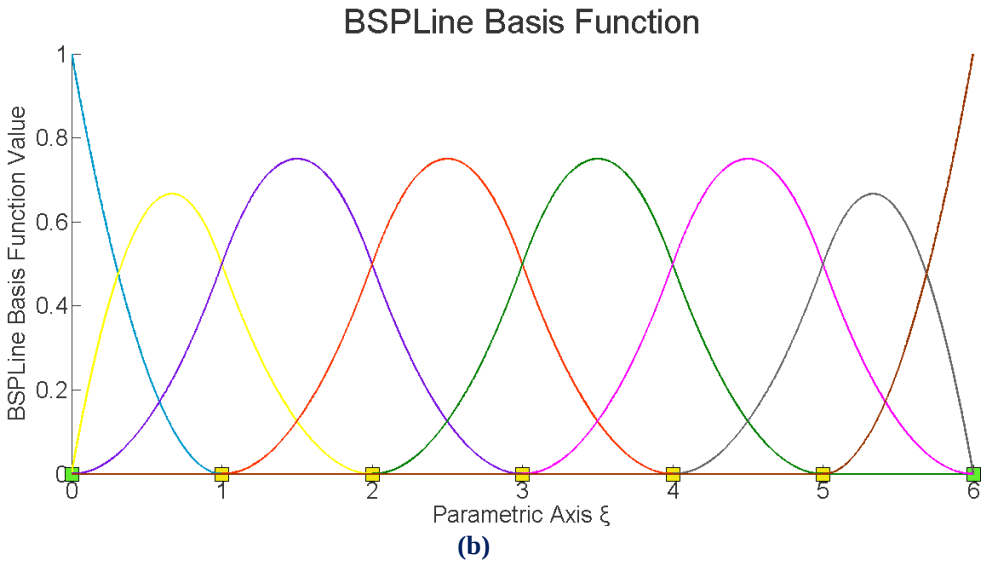
2. Control point – basis function correspondence

A B-spline curve is defined by n control points and n basis functions and the correspondence between them is unique. That means that each basis function is associated with a certain control point, despite the fact that other control points belong to its domain of influence too. This property is valid for surfaces and solids as well. In these occasions B-spline is a tensor product of basis functions across ξ, η and ζ. Each one of them is controlled by a specific one-dimensional point. The tensor product of these points constitutes the two-dimensional or the three-dimensional point which controls the B-spline surface or solid respectively.

In Fig. 2.19 the pairs of control points and their corresponding basis functions are presented in different colors.



(a)

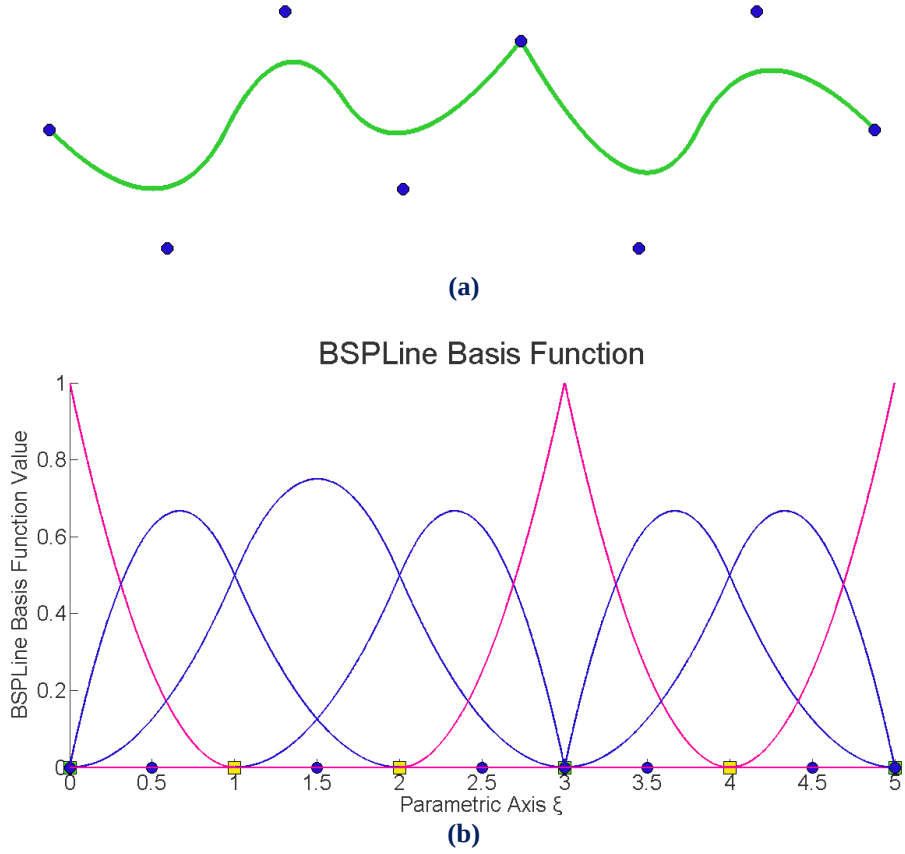


(b)

Fig. 2.19. Control point-basis function correspondence:
(a) physical space and **(b)** basis functions with the corresponding control points

3. Endpoint interpolation

The first and last control points are interpolatory to the curve. Any internal control point corresponding to C^0 continuous basis function is also interpolatory to the curve. In Fig. 2.20 the first and the last control points have C^{-1} continuity, while the internal control point lying at ξ=3 has C^0 continuity.



(a)

(b)

Fig. 2.20. Control point interpolation:
(a) B-spline curve, **(b)** the corresponding basis functions

4. Affine invariance

Any affine transformation destined for the B-spline curve, can be applied directly to the control points. By the term affine transformations, translations, rotations, scalings and uniform stretchings and shearings are meant. Affine invariance, or else affine covariance, arises from the partition of unity property of B-spline basis functions. The transformation is denoted by Φ: $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ and has the form

$$\Phi(x)=Ax+v$$

where A is a 3x3 matrix and v is a vector.

5. Control point local support

Each control point is associated with a certain B-spline basis function. This function is non-zero exclusively across its local support, which includes p+1 knot value spans. The corresponding control point shares the same local support with the function, and thus its domain of influence is defined. That is, moving a control point $\mathbf{X_i}$, only the interval $\left[\xi_i, \xi_{i+p+1}\right)$ will "perceive" the alteration.

In Fig. 2.21 the control point lying at ξ=1.5 is moved and the curve is partially affected. The support of the control point is shown in the chart with the B-spline values.

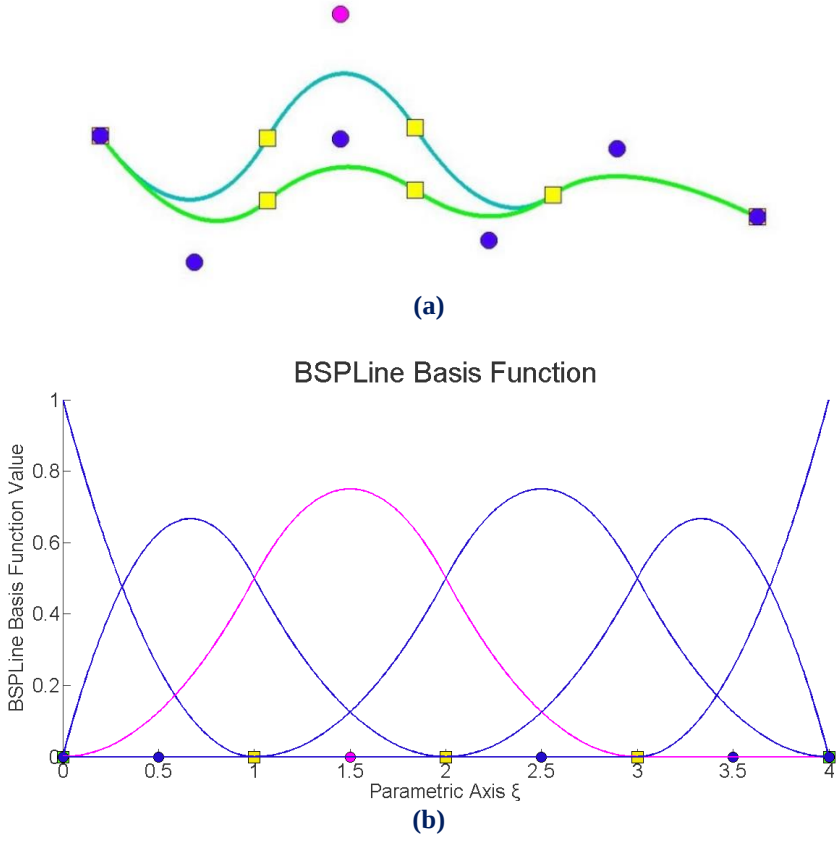


Fig. 2.21. Control point local support:
(a) physical space, **(b)** B-spline basis functions

6. Convex hull

A B-spline curve is completely contained in the convex hull defined by its control points. This follows from the non-negativity and partition of unity properties of the basis, as well as the support of p+1 spans for every function. Given a curve of degree p the convex hull is formed by merging the convex hulls created by p+1 consecutive control points.

In Fig. 2.22 the convex hull is built gradually for a curve of degree p=2, connecting each control point with its p=2 subsequent ones. It is apparent that the convex hull encloses the curve, while it describes its general form.

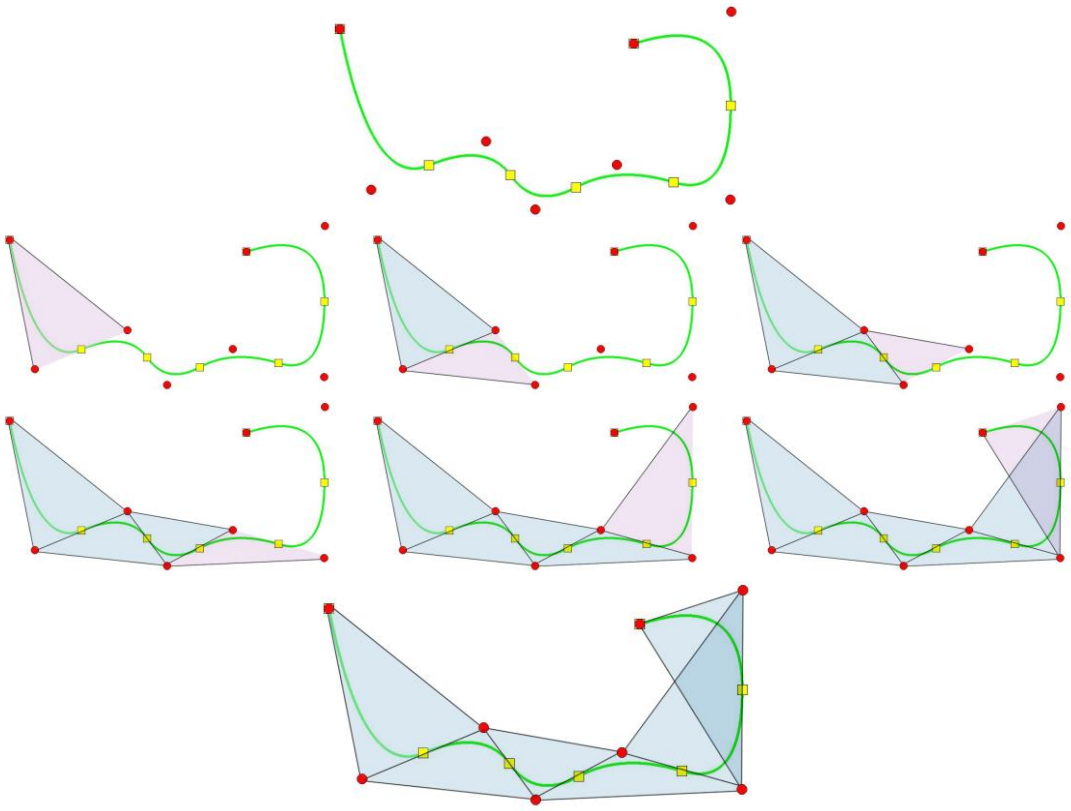


Fig. 2.22. Step-by-step convex hull creation for a B-spline curve

2.4 Non-Uniform Rational B-splines

2.4.1 The basic concept

Non uniform rational B-splines, NURBS, are piecewise rational polynomials built from B-splines. The term non uniform denotes the use of non-uniform knot value vectors. These vectors enable the engineer or the designer to intervene in the continuity of the function across knots by increasing their multiplicity, therefore a better approximation to the physical model is achieved. The term rational implies that two polynomials are involved in the formation of NURBS, which are produced by the basis functions and certain weighting functions.

NURBS can exactly represent a wide range of objects, especially conic sections such as circles and ellipses. Second-order Lagrange polynomials on quadrilateral elements

in standard FEA would require an increased number of nodes in order to mimic the curved behavior and eliminate the angles that successive elements form. NURBS instead use elements that fit exactly the curve's shape, thus they work with a minimum number of nodes. For instance, a circle can be exactly represented by only nine points. It is clear that NURBS suit perfectly to the concept of isogeometric analysis, that is to use the same mathematical model for both design and analysis, since they provide accuracy, robustness and flexibility.

The basic idea is the principles of B-splines to be adopted by NURBS, so that they can be handled in a similar way. Indeed NURBS curves, surfaces and solids are generated by a linear combination of basis functions and control points, and as a result all the properties of B-splines are efficiently transferred to NURBS products. The final NURBS object is produced from the actual section of a cone with a plane. As shown in Fig. 2.38, the projective B-spline curve $\mathbf{C}^{\mathrm{w}}(\xi)$ is created from the projective 3D control points $\mathbf{X}^{\mathrm{w}} = \left\{ \mathrm{X}^{\mathrm{w}} \quad \mathrm{Y}^{\mathrm{w}} \quad \mathrm{Z}^{\mathrm{w}} \right\}$.

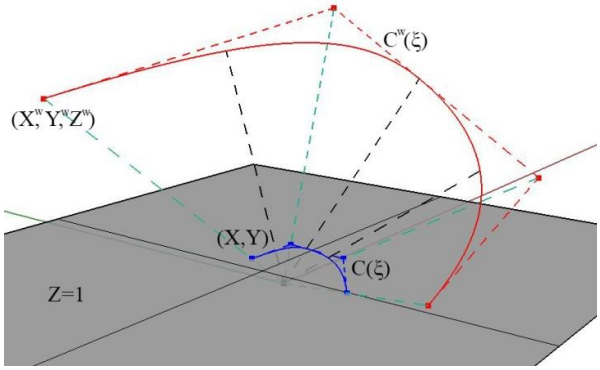


Fig. 2.23. B-spline curve and projective transformation to NURBS curve

Projection of the curve and control points on the plane z=1 produces the NURBS curve $\mathrm{C}(\xi)$ and the 2D control points:

$$\mathbf{X} = \left\{ \mathrm{X} \quad \mathrm{Y} \right\}$$

where,

$$\left\{ \mathrm{X_i} \quad \mathrm{Y_i} \right\} = \left\{ \frac{\mathrm{X_i^w}}{\mathrm{Z_i^w}} \quad \frac{\mathrm{Y_i^w}}{\mathrm{Z_i^w}} \right\}$$

The weights of the NURBS curve are defined as:

$$\mathrm{w} = \left\{ \mathrm{Z^w} \right\}$$

In general *n*-dimensional rational B-splines are projections of $(n+1)$-dimensional non rational B-splines.

2.4.2 Shape functions

In order to evaluate NURBS shape functions, the weighting function is defined:

$$W(\xi) = \sum_{i=1}^{n} \left\{ N_{i,p}(\xi) \cdot w_i \right\}$$

In most engineering applications, weights have positive values. Unless otherwise stated, they will be considered positive for the scope of this thesis. $W(\xi)$ is in fact the Z-coordinate of the projective B-spline curve. Projective transformation is applied by dividing the other two coordinates of the B-spline curve with the Z-coordinate. NURBS shape functions are calculated from

$$R_i^p(\xi) = \frac{N_{i,p}(\xi) \cdot w_i}{W(\xi)} = \frac{N_{i,p}(\xi) \cdot w_i}{\sum_{i'=1}^{n} \left\{ N_{i',p}(\xi) \cdot w_{i'} \right\}}$$

$R_i^p(\xi)$ are piecewise rational functions. The expression "the order of NURBS" refers to the order of the projective B-SPLine curve.

NURBS shape functions in multiple directions can be obtained as tensor products of one-directional basis functions. Shape functions for two directions:

$$R_{i,j}^{p,q}(\xi,\eta) = \frac{N_{i,p}(\xi) \cdot M_{j,q}(\eta) \cdot w_{ij}}{\sum_{i'=1}^{n} \sum_{j'=1}^{m} \left\{ N_{i',p}(\xi) \cdot M_{j',q}(\eta) \cdot w_{i'j'} \right\}}$$

$$W(\xi,\eta) = \sum_{i'=1}^{n} \sum_{j'=1}^{m} \left\{ N_{i',p}(\xi) \cdot M_{j',q}(\eta) \cdot w_{i'j'} \right\}$$

Similarly, the equations are extended in three directions:

$$R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) = \frac{N_{i,p}(\xi) \cdot M_{j,q}(\eta) \cdot L_{k,r}(\zeta) \cdot w_{ijk}}{\sum_{i'=1}^{n} \sum_{j'=1}^{m} \sum_{k'=1}^{l} \left\{ N_{i',p}(\xi) \cdot M_{j',q}(\eta) \cdot L_{k',r}(\zeta) \cdot w_{i'j'k'} \right\}}$$

The weighting function is now defined as:

$$W(\xi,\eta,\zeta) = \sum_{i'=1}^{n} \sum_{j'=1}^{m} \sum_{k'=1}^{l} \left\{ N_{i',p}(\xi) \cdot M_{j',q}(\eta) \cdot L_{k',r}(\zeta) \cdot w_{i'j'k'} \right\}$$

Observe that for $w_{ijk} = 1$, $\forall$ i, j, k , it applies that NURBS shape functions downgrade to B-spline basis functions. Actually, NURBS entities are a generalization of B-spline entities. All the B-spline properties examined in this thesis apply for NURBS as well.

2.4.3 Shape function derivatives

Simple application of the quotient rule yields the derivatives of NURBS shape functions for one and multiple directions.

$$\frac{d}{d\xi} R_i^p(\xi) = w_i \cdot \frac{\left(\frac{d}{d\xi} N_{i,p}(\xi)\right) \cdot W(\xi) - \left(\frac{d}{d\xi} W(\xi)\right) \cdot N_{i,p}(\xi)}{\left(W(\xi)\right)^2}$$

where

$$\frac{d}{d\xi} W(\xi) = \sum_{i=1}^{n} \left(\frac{d}{d\xi} N_{i,p}(\xi)\right) \cdot w_i$$

For bidirectional shape functions:

$$\frac{\partial}{\partial\xi} R_{i,j}^{p,q}(\xi,\eta) = w_{ij} \cdot \frac{\left(\frac{d}{d\xi} N_{i,p}(\xi)\right) \cdot M_{j,q}(\eta) \cdot W(\xi,\eta) - \left(\frac{\partial}{\partial\xi} W(\xi,\eta)\right) \cdot N_{i,p}(\xi) \cdot M_{j,q}(\eta)}{\left(W(\xi,\eta)\right)^2}$$

$$\frac{\partial}{\partial\eta} R_{i,j}^{p,q}(\xi,\eta) = w_{ij} \cdot \frac{N_{i,p}(\xi) \cdot \left(\frac{d}{d\eta} M_{j,q}(\eta)\right) \cdot W(\xi,\eta) - \left(\frac{\partial}{\partial\eta} W(\xi,\eta)\right) \cdot N_{i,p}(\xi) \cdot M_{j,q}(\eta)}{\left(W(\xi,\eta)\right)^2}$$

Derivatives of 3D shape functions per direction are evaluated as shown

$$\frac{\partial}{\partial\xi} R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) =$$

$$= w_{ijk} \cdot \frac{\left(\frac{d}{d\xi} N_{i,p}(\xi)\right) \cdot M_{j,q}(\eta) \cdot L_{k,r}(\zeta) \cdot W(\xi,\eta,\zeta) - \left(\frac{\partial}{\partial\xi} W(\xi,\eta,\zeta)\right) \cdot N_{i,p}(\xi) \cdot M_{j,q}(\eta) \cdot L_{k,r}(\zeta)}{\left(W(\xi,\eta,\zeta)\right)^2}$$

$$\frac{\partial}{\partial\eta} R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) =$$

$$= w_{ijk} \cdot \frac{N_{i,p}(\xi) \cdot \left(\frac{d}{d\eta} M_{j,q}(\eta)\right) \cdot L_{k,r}(\zeta) \cdot W(\xi,\eta,\zeta) - \left(\frac{\partial}{\partial\eta} W(\xi,\eta,\zeta)\right) \cdot N_{i,p}(\xi) \cdot M_{j,q}(\eta) \cdot L_{k,r}(\zeta)}{\left(W(\xi,\eta,\zeta)\right)^2}$$

$$\frac{\partial}{\partial\zeta} R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) =$$

$$= w_{ijk} \cdot \frac{N_{i,p}(\xi) \cdot M_{j,q}(\eta) \cdot \left(\frac{d}{d\zeta} L_{k,r}(\zeta)\right) \cdot W(\xi,\eta,\zeta) - \left(\frac{\partial}{\partial\zeta} W(\xi,\eta,\zeta)\right) \cdot N_{i,p}(\xi) \cdot M_{j,q}(\eta) \cdot L_{k,r}(\zeta)}{\left(W(\xi,\eta,\zeta)\right)^2}$$

2.4.4 Entities

NURBS entities are created as a linear combination of NURBS shape functions, exactly the same way as B-spline entities. The following is the equation for the creation of NURBS curves:

$$C(\xi) = \sum_{i=1}^{n} \left\{ R_i^p(\xi) \cdot X_i \right\}$$

surfaces:

$$S(\xi,\eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \left\{ R_{i,j}^{p,q}(\xi,\eta) \cdot X_{i,j} \right\}$$

and solids:

$$S(\xi,\eta,\zeta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{l} \left\{ R_{i,j,k}^{p,q,r}(\xi,\eta,\zeta) \cdot X_{i,j,k} \right\}$$

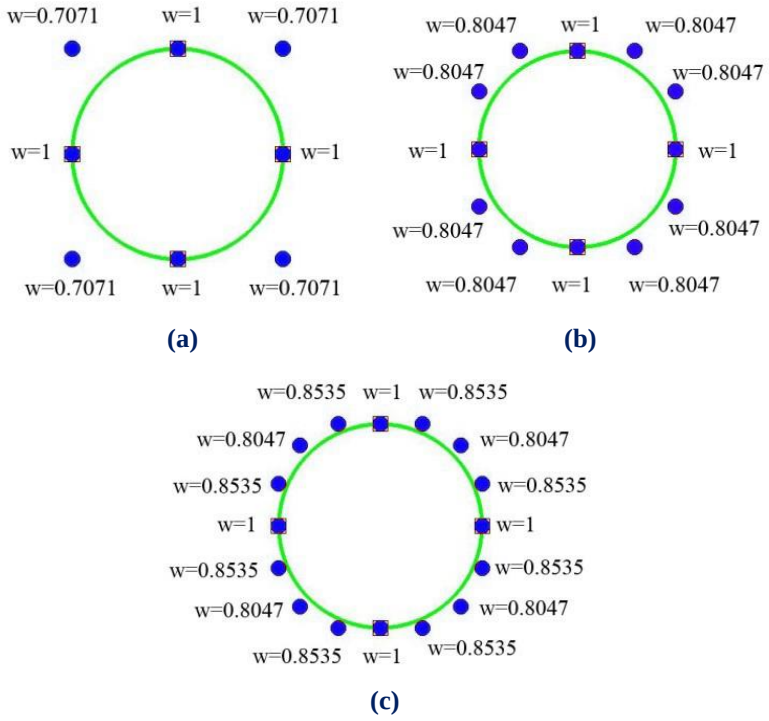


Fig. 2.24. NURBS circle of different polynomial degrees. Weight values for each control point.

(a) Quadratic basis functions

$$\Xi = \{0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 2 \quad 2 \quad 3 \quad 3 \quad 4 \quad 4 \quad 4\}$$

(b) Cubic basis functions

$$\Xi = \{0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 2 \quad 2 \quad 2 \quad 3 \quad 3 \quad 3 \quad 4 \quad 4 \quad 4 \quad 4\}$$

(c) Quadric basis functions

$$\Xi = \{0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 2 \quad 2 \quad 2 \quad 2 \quad 3 \quad 3 \quad 3 \quad 3 \quad 4 \quad 4 \quad 4 \quad 4 \quad 4\}$$

3 Numerical Integration Schemes

3.1 The method of weighted residuals

In order to solve differential equations and define a satisfying approach to the actual solution, the method of weighted residuals (MWR) is widely used. The basic concept is to approximate the actual solution by a linear combination of basis functions chosen from a linearly independent set. Suppose we have a linear differential operator D acting on a function u to produce function p.

$$D\left(u\left(x\right)\right) = p\left(x\right)$$

The exact solution u is approximated by a set of functions $\tilde{u}$ so that

$$u \cong \tilde{u} = \sum_{i=1}^{n} a_i \varphi_i$$

When substituted into the initial equation an error or residual will occur

$$E\left(x\right) = R\left(x\right) = D\left(\tilde{u}\left(x\right)\right) - p\left(x\right) \neq 0$$

The method of weighted residuals pursues to minimize that error and forces the residual to zero in some average sense over the domain through the equation:

$$\int_x R\left(x\right) W_i dx = 0 \quad i = 1, 2, \ldots, n$$

W_i are called weight functions and since their number is equal to the number of unknown constants α_i in $\tilde{u}$, the integral results in a system of n algebraic equations for α_i.

There exist variant techniques in order to define these test functions and each one of them constitutes a MWR sub-method. Collocation, sub-domain, least squares, Galerkin and method of moments are some of the most popular ones. For the scope of this thesis

collocation and Galerkin method are examined in particular and their efficiency for NURBS based isogeometric analysis is under investigation.

3.2 The Boundary Value Problem (BVP)

Suppose we have a boundary value problem defined in strong form as

$$L(u) = f \quad in\ \Omega$$
$$u = u_D \quad on\ \Gamma_D$$
$$\boldsymbol{n} \cdot D\nabla u = h \quad on\ \Gamma_N$$

where L denotes a linear operator, D is the diffusion coefficient, f is a source term and $\boldsymbol{n}$ represents the unit outward normal along Γ. The function u_D specifies the solution of u on the Dirichlet boundary Γ_D, while function h specifies the normal flux on the Neumann boundary Γ_N.

According to the method of weighted residuals the approximate solution u^* is of the form:

$$u^* = \tilde{u}_D(\boldsymbol{x}) + \sum_{i=1}^{n} N_i(\boldsymbol{x}) c_i$$

where N_i are the NURBS basis functions described in Chapter 2 and c_i are the unknown displacement control variables. The function $\tilde{u}_D$ is considered as an extension to the prescribed boundary condition, that is, it is defined on Ω and satisfies the Dirichlet boundary condition when evaluated on Γ_D.

In order to minimize the error that arises with the substitution of u^* into the equations that form the boundary value problem in Ω and on Γ_N, the unknown coefficients c_i are calculated with respect to the following equation:

$$\int_{\Omega} \left(L(u^*) - f \right) \omega_{\Omega} d\Omega + \int_{\Gamma_N} \left(\boldsymbol{n} \cdot D\nabla u^* - h \right) \omega_{\Gamma} d\Gamma = 0 \tag{1}$$

It is worth mentioning that Dirichlet boundary Γ_D is not taken into consideration, since the corresponding boundary conditions are exactly satisfied a priori. Functions ω_{Ω} and ω_{Γ} are test functions defined according to the selected numerical method (here Galerkin and collocation) over the domain Ω and the Neumann boundary Γ_N respectively.

3.3 Galerkin

3.3.1 Test functions

Galerkin is the most widely used numerical method in FEA, hence its implementation to the advanced method of isogeometric analysis constitutes an obvious alternative. Galerkin procedure adopts the same basis functions for the approximate solution and the weighting function approach. In other words, test functions ω_Ω and ω_Γ identify with the approximation u^* of the solution, while on Γ_D they are equal to zero.

$$\omega_\Omega = \omega_\Gamma = \sum_{i=1}^{n} N_i(\boldsymbol{x})\hat{c}_i$$

The mathematical formulation forces the weighted residual defined in Eq. (1) to be orthogonal to each basis function ω_i as follows

$$\int_\Omega \left(L\left[u^*(\boldsymbol{x})\right] - f(\boldsymbol{x})\right)\omega_i d\Omega + \int_{\Gamma_N} \left(\boldsymbol{n}\cdot D\nabla u^*(\boldsymbol{x}) - h(\boldsymbol{x})\right)\omega_i d\Gamma = 0$$

The above equation indicates that Galerkin supports the weak form of the boundary value problem. As it has already been proved in section 2.3.3, Galerkin test functions have a local support of p+1 knot value spans, which implies that integrals are non-zero in more than one point x_i and thus they cannot be eliminated from the discretized variational statement. Consequently, numerical quadrature rules of the form

$$\int_\Omega g(\boldsymbol{x})d\Omega = \sum_k g(\boldsymbol{x}_k)w_k$$

are required, so as to covert the continuous integral into a sum of several point evaluations multiplied by corresponding weights w_k in each element. This procedure concerns Gauss points, which are chosen for the domain of every piece of the polynomial basis function.

3.3.2 Gauss points

Despite the fact there are two ingredients of isogeometric analysis that could play the role of FEM element, the patch or the knot span, in this thesis the second alternative is adopted, since it serves better the needs of numerical integration. Knot span corresponds to the domain of every piece of the polynomial basis function, where Gauss points are defined, thus resembles the element of FEM.

Gauss points' parametric coordinates are obtained as the roots of the Legendre Polynomial on a reference element spanning [-1,1]. The next step is to transform the coordinates and weights from the reference knot span ξ^R to the desired knot span $\left[\xi_i, \xi_{i+1}\right)$.

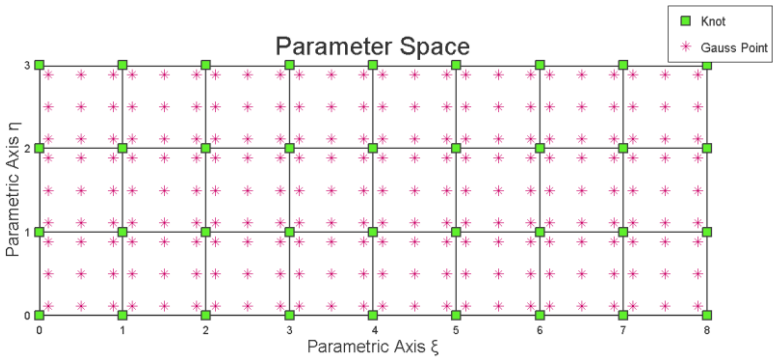
$$\xi = \frac{\left(\xi_{i+1} - \xi_i\right)\xi^R + \left(\xi_{i+1} + \xi_i\right)}{2}$$

$$w^{GP\xi} = \frac{\left(\xi_{i+1} - \xi_i\right)}{2} w_\xi^R$$

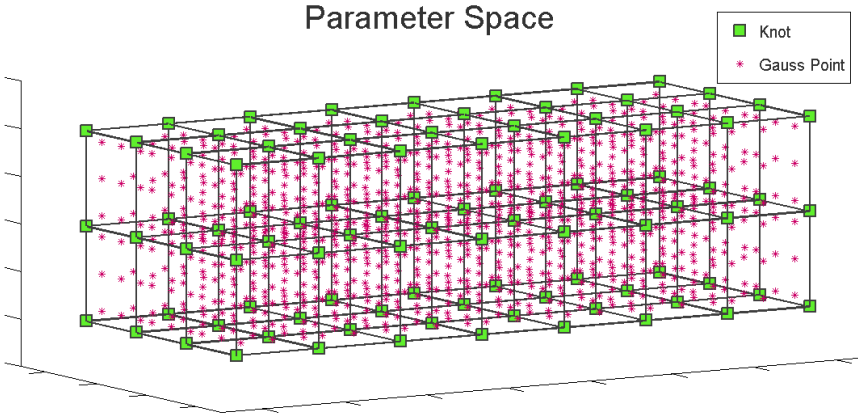
Full tensor product properties apply here as well, leading in similar equations for the other two parametric directions:

$$\eta = \frac{\left(\eta_{j+1} - \eta_j\right)\eta^R + \left(\eta_{j+1} + \eta_j\right)}{2}, \quad \zeta = \frac{\left(\zeta_{k+1} - \zeta_k\right)\zeta^R + \left(\zeta_{k+1} + \zeta_k\right)}{2}$$

$$w^{GP\eta} = \frac{\left(\eta_{j+1} - \eta_j\right)}{2} w_\eta^R, \quad w^{GP\zeta} = \frac{\left(\zeta_{k+1} - \zeta_k\right)}{2} w_\zeta^R$$



(a)



(b)

Fig. 3.1. Gauss points in parameter space:
(a) surface, **(b)** solid

In order to achieve exact numerical integration of a polynomial of degree *p*, $\frac{p+1}{2}$ or $\frac{p+2}{2}$ Gauss points are required on its domain, for *p* odd and even respectively. B-spline basis functions consist of various polynomials, thus the appropriate number of Gauss points at each knot span is determined by the degree of the amount to integrate. According to [4] numerical quadrature with respect to stiffness matrix has to calculate the integral

$$\int_{\hat{\Omega}^2} \nabla R_i(\xi) \nabla R_j(\xi) \phi(\xi) d\xi$$

where *i* and *j* are two-dimensional multi-indices, R_i and R_j are NURBS shape functions and ϕ is the space-dependent factor that is responsible for alterations in geometry. In other words, it represents the Jacobian of the geometry map and, possibly, the derivatives of its inverse.

In one-dimensional cases, this product yields to a polynomial of maximum degree $q=(p\text{-}1)+(p\text{-}1)=2p\text{-}2$, considering that the derivative of each basis function is of degree *p*-1. Therefore, the minimum number of Gauss points per knot span required for exact integration is:

$$\frac{q+2}{2}=\frac{2p-2+2}{2}=p$$

For 2D problems, the evaluation of the gradients forces the basis function of one direction (degree: *p*) to be multiplied with the derivative of the basis function in the other direction (degree: *p*-1). After this product is squared, the polynomial that arises is of maximum degree $q=p+p=2p$. In order to achieve exact integration, the minimum number of Gauss points per knot span is

$$\frac{q+2}{2}=\frac{2p+2}{2}=p+1$$

In a similar way it can be proved that 3D cases also require *p*+1 Gauss points per knot span for exact integration.

3.3.3 Computational efficiency

Classical FEM downsizes the natural problem of infinite unknowns to a finite number of unknowns, which are the degrees of freedom of the nodes. The position of the nodes depends on the element type. As a general rule, the nodes can be usually found in the

corners and middle of the sides of the elements. They are part of the element and therefore part of the physical model. Displacements in other areas of the model can be approximated by a linear combination of displacements on the degrees of freedom. Distribution in the model is evaluated via the corresponding shape functions. In isoparametric elements, shape functions and their respective nodes are also used to approximate the geometry, thus enabling relatively complex shapes to be approximated with Lagrange polynomials.

In isogeometric analysis, NURBS are often chosen as shape functions. The isoparametric concept is reversed, as geometrical mapping now defines the solution approximation. The geometrical representation is achieved through a combination of control points and their corresponding shape functions. Degrees of freedom at the control points are now the unknowns.

A major difference between the two methods is that in FEA basis functions are restricted to the interior of each element, while in IGA they are non-zero across p+1 knot spans (elements) and overlap with more basis functions. That implies that in the first case Gauss points are influenced only by the nodes of their own elements. In the second case instead each Gauss point is involved in computations with control points that are extended to a significant larger area than a knot span, that is $(p+1)^d$, where d is the dimension of the problem.

Fig. 3.2 depicts the areas influenced by a control point with respect to the polynomial degree p in 1D case. Fig 3.3 focuses on 2D cases and illustrates the difference with the classical FEM. A distinction between even and odd p is essential, since for even degrees control points are located in the middle of the knot spans in parameter space, while for odd degrees they coincide with knots.

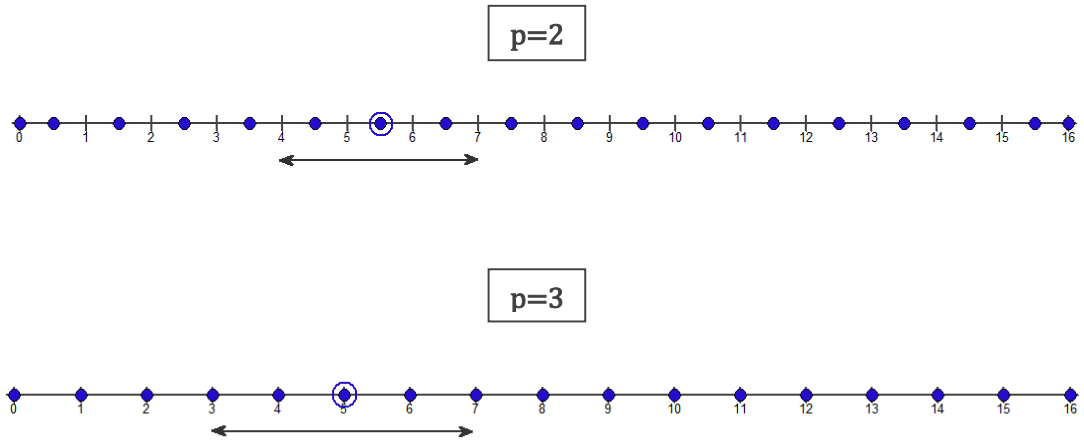
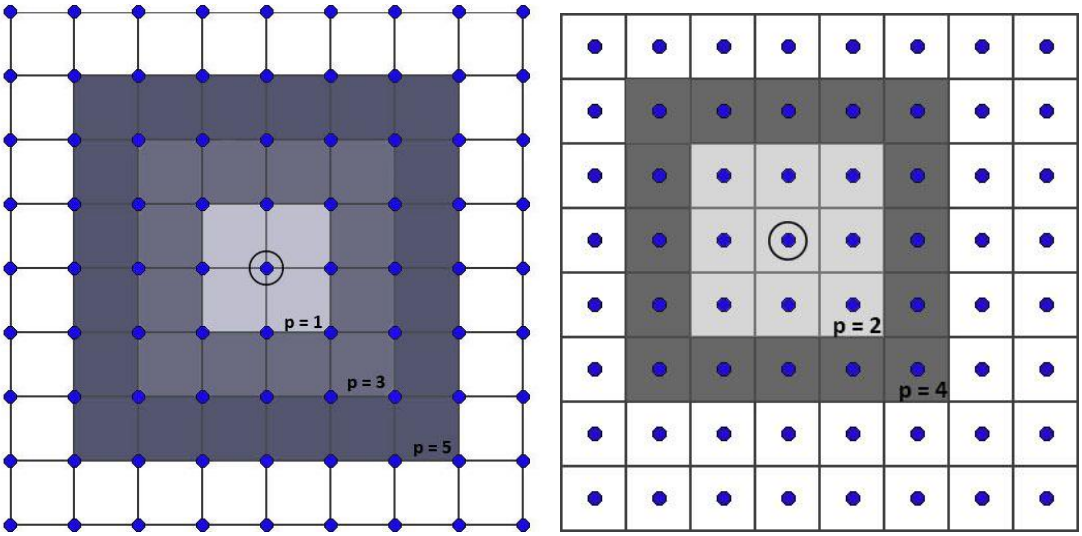
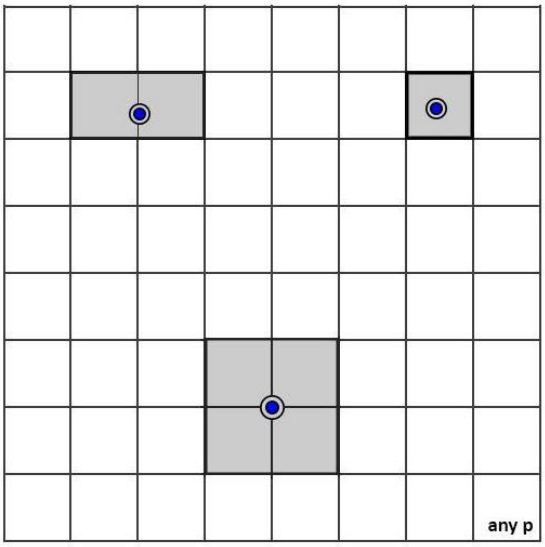


Fig. 3.2. IGA 1D domains of influence of control points



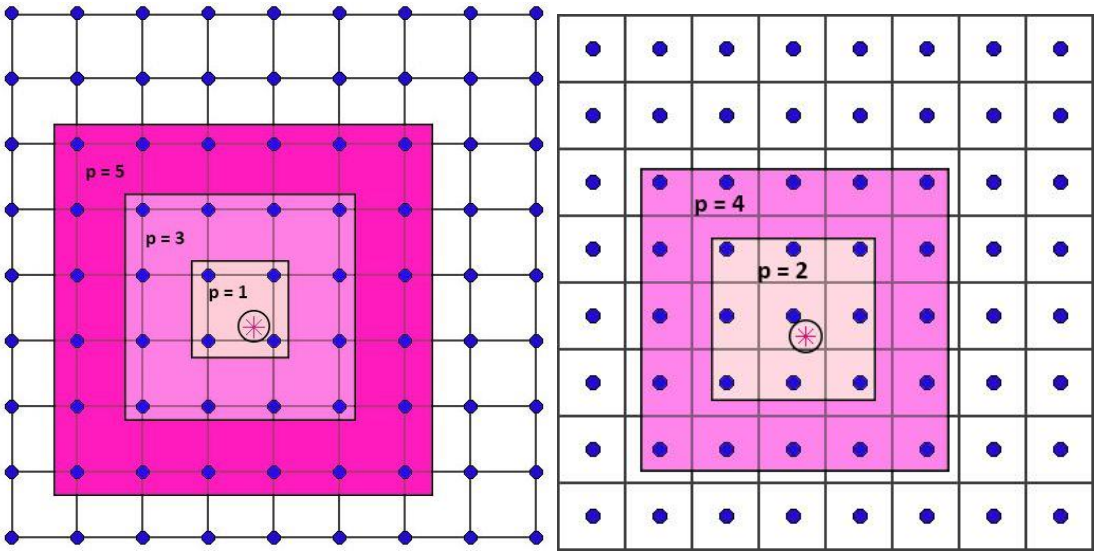
(a)



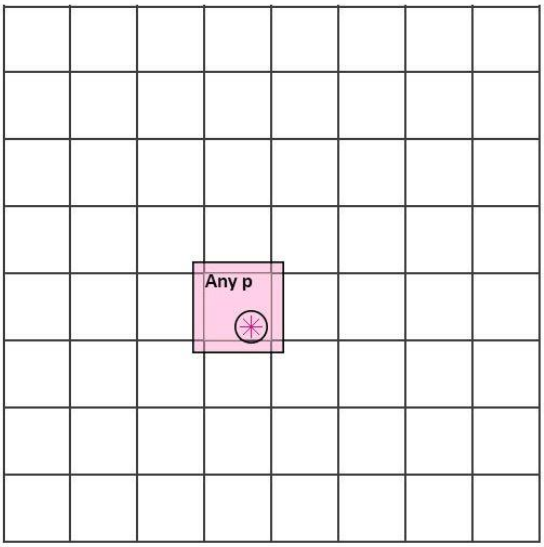
(b)

Fig. 3.3. 2D domains of influence of control points:
(a) IGA (*p* odd and even respectively), **(b)** FEA

The large amount of control point interactions is the reason why IGA has denser stiffness matrices than FEA, despite the fact that the bandwidth is the same between the two methods when meshes with the same degrees of freedom are concerned. It leads to highly localized overlapping of shape functions, therefore consecutive elements occupy same positions of global stiffness matrix. Apart from that, the areas influenced by a control point also reveal the amount of control point-Gauss point correlations, which is of great importance for the computational cost of each method. Apparently, this number is significantly increased in IGA. This conclusion is underlined in the following figure, depicting control points/nodes influenced by a Gauss point in IGA/FEA.



(a)



(b)

Fig. 3.4. Control points/nodes influenced by a Gauss point:
(a) IGA (*p* odd and even respectively), **(b)** FEA

Table 3.1 calculates the total number of elements and Gauss points in IGA and FEA for 121 control points/nodes in each axis and various values of order *p*. A *p*+1 integration rule is adopted in both methods. As a result the number of Gauss points per element is the same for the two formulations. The increasing ratio in the last column is due to the different set up of elements in each method according to the available knots/nodes.

	n=121		Total elements		Total Gauss points		Ratio
	p	GP per element	IGA	FEA	IGA	FEA	
2D	1	4	14.400	14.400	57.600	57.600	1
	2	9	14.161	3.600	127.449	32.400	3,9
	3	16	13.924	1.600	222.784	25.600	8,7
	4	25	13.689	900	342.225	22.500	15,2
	5	36	13.456	576	484.416	20.736	23,4
3D	1	8	1.728.000	1.728.000	13.824.000	13.824.000	1
	2	27	1.685.159	216.000	45.499.293	5.832.000	7,8
	3	64	1.643.032	64.000	105.154.048	4.096.000	25,7
	4	125	1.601.613	27.000	200.201.625	3.375.000	59,3
	5	216	1.560.896	13.824	337.153.536	2.985.984	112,9

Table 3.1. Total number of elements and Gauss points in IGA and FEA, for n=121 control points/nodes

For the same number of degrees of freedom, IGA occupies orders of magnitude more quadrature points than FEA. The quantitative comparison is illustrated in the following charts.

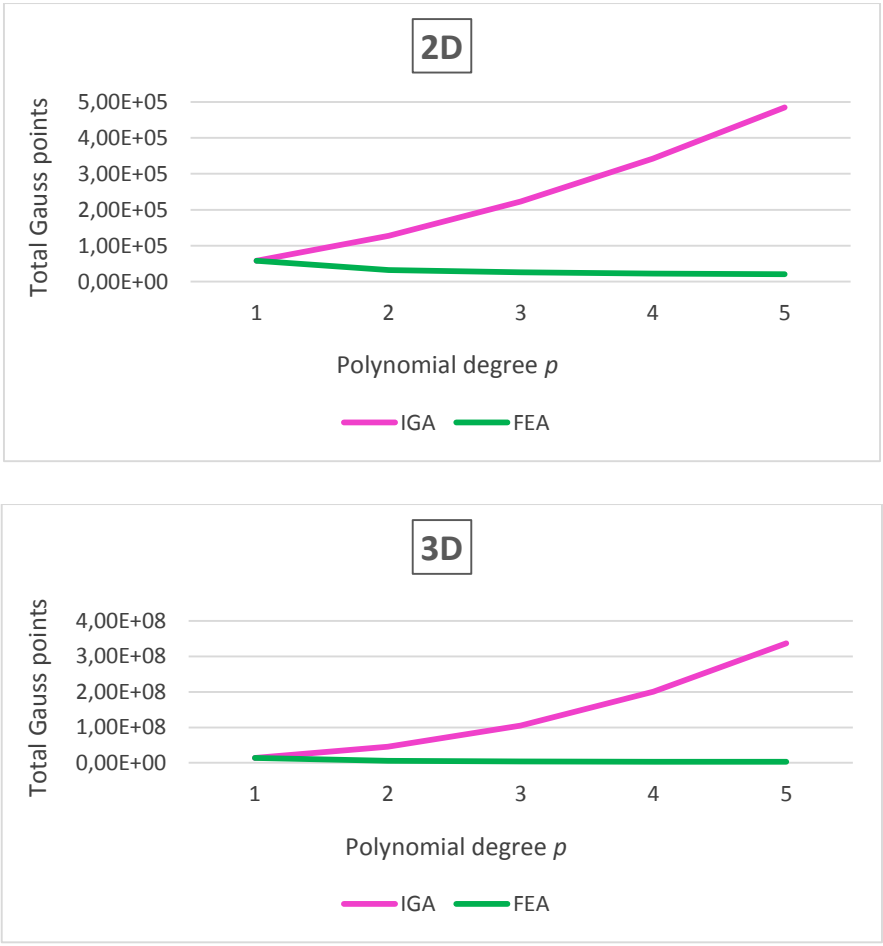


Fig. 3.5. Total number of Gauss points in IGA and FEA for different p

Let us thoroughly elaborate the calculations of Table 3.1.

• 2D, *p*=1

Gauss points per element: In two-dimensional cases the required number of quadrature points per element is determined by the rule $(p+1)^2$. For $p=1 \Rightarrow (p+1)^2 = (1+1)^2 = 4.$

Total elements in IGA: The number of knot values *k* is related to the number of control points *n* and the polynomial degree *p* via the equation $k = n + p + 1$, which is applied to each parametric direction. In the current example knot values across axis ξ and axis η are equal to *k*=121+1+1=123. Due to the open vector nature of the knot vector, the first and the last knot values are repeated *p*+1 times, that is 2. Considering that the internal knots have continuity C^1 (their multiplicity is equal to 1), it becomes clear that there are 123-2=121 knots and therefore 120 knot spans at each parametric axis. The adaptation to 2D parameter space yields $120^2 = 14400$ elements.

Total elements in FEA: Provided that *p*=1, one-dimensional elements are formed by 2 nodes. Axes ξ and η are divided in 120 spans, thus there $120^2 = 14400$ two-dimensional elements.

Total Gauss points in IGA: This size arises from the multiplication of the number of Gauss points per element with the total number of elements in IGA. Indeed, 4x14400=57600.

Total Gauss points in FEA: In a similar way, total Gauss points in FEA are equal to the number of Gauss points per element multiplied with the total number of elements in FEA. 4x14400=57600.

• 2D, *p*=2

Gauss points per element: The minimum number of quadrature points per element for efficient numerical integration is equal to $(p+1)^2 = (2+1)^2 = 9$.

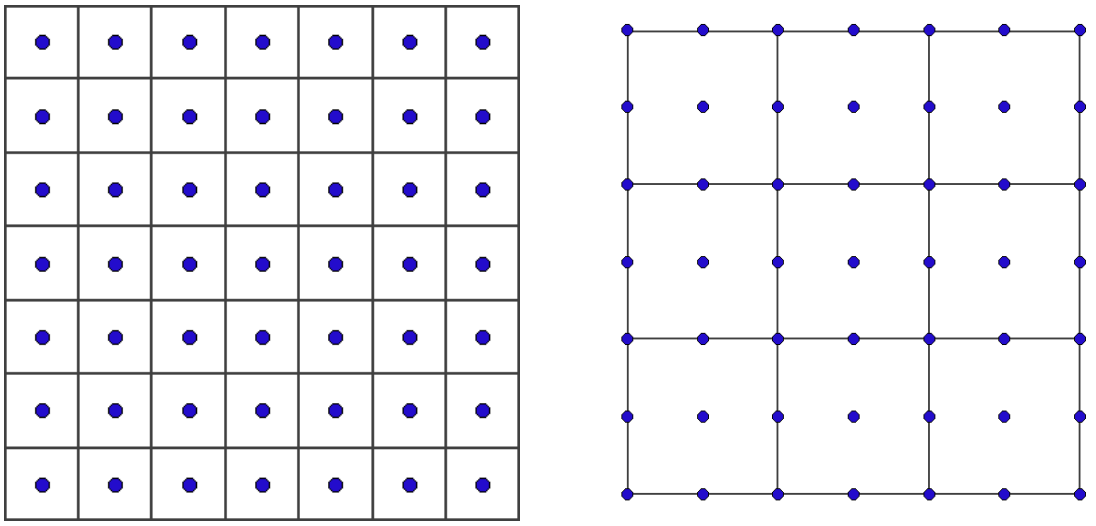
Total elements in IGA: *k*=*n*+*p*+1=121+2+1=124 knot values at each axis. The first and the last knot values appear *p*+1=2+1=3 times, thus there exist 124-2x2=120 knots across ξ and 120 knots across η, which means 119 knot spans at each direction. The total number of elements is equal to $119^2 = 14161$.

Total elements in FEA: In order to form a quadratic one-dimensional element in classical FEM, three nodes are required. Now that 121 nodes are available, the corresponding elements across each axis are 60. The total number of two-dimensional elements is equal to $60^2 = 3600$.

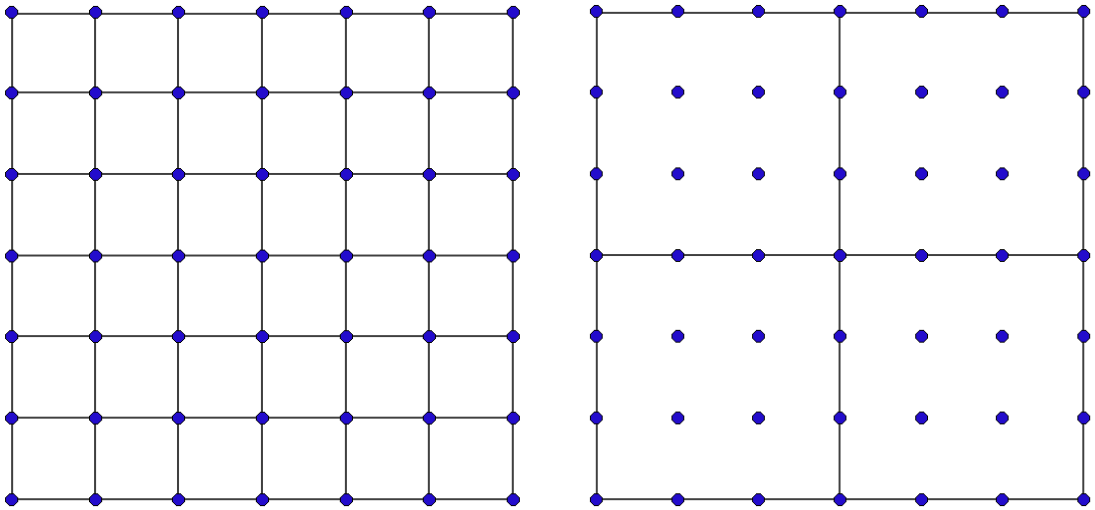
Total Gauss points in IGA: The multiplication of the number of Gauss points per element with the total number of elements in IGA yields 9x14161=127449.

Total Gauss points in FEA: The multiplication of the number of Gauss points per element with the total number of elements in FEA yields 9x3600=32400.

The computations of the following rows of Table 3.1 are continued in the same way. This table points out the difference between the elements of the two methods. In classical FEA each element of degree p occupies p+1 nodes. Thus, for a given number of nodes and an increasing p, the total number of elements tends to be reduced in a significantly high rate.



(a)



(b)

Fig. 3.6. Visual comparison between IGA (left column) and FEA (right column) for the same number of control points
(a) p=2, **(b)** p=3

In IGA, on the other hand, the one-dimensional element is always defined by two knots regardless the value of the discretization degree. The role of *p* is limited to determining the repetitions of the first and the last knot values, which affect up to a point the number of knots, the difference though between consecutive orders considering a standard amount of control points is negligible. As a result, the total number of elements remains at the same levels, despite the increase of *p*.

The overlapping of basis functions in IGA increases the number of elements and therefore, the number of Gauss points which influence a control point compared to FEM. Tables 3.2 and 3.3 calculate the corresponding correlations for FEA and IGA respectively.

	p	GP per element	Nodes influenced by GP/element	Elements influencing a node		Gauss points influencing a node	
				min	max	min	max
2D	1	4	4	4	4	16	16
	2	9	9	1	4	9	36
	3	16	16	1	4	16	64
	4	25	25	1	4	25	100
	5	36	36	1	4	36	144
3D	1	8	8	8	8	64	64
	2	27	27	1	8	27	216
	3	64	64	1	8	64	512
	4	125	125	1	8	125	1000
	5	216	216	1	8	216	1728

Table 3.2. Correlations of nodes with elements and Gauss points for FEA

Gauss points per element: This number is determined by the integration rule $(p+1)^d$, where *d* is the dimension of the problem.

Nodes influenced by GP/element: In classical FEM one element of degree *p* is formed by *p*+1 nodes in each parametric direction. For instance, a quadratic element in 2D simulation requires $(2+1)^2=3^2=9$ nodes.

Elements influencing a node: The number of elements that each node communicate with, depends on the position of the node. Corner nodes in 2D and 3D problems participate in 4 and 8 elements respectively, while internal nodes are only influenced by the element they belong to. Nodes lying at the edges/sides are in-between. Apparently, at elements with *p*=1, corner nodes are met exclusively, therefore a min and max tabulation is pointless.

Gauss points influencing a node: It is determined by the number of elements that influence the node and the number of Gauss points that influence each element. The classification to min/max is thus inherited by the previous column.

	p	GP per element	Control points influenced by GP/element	Elements influencing a control point	Gauss points influencing a control point
2D	1	4	4	4	16
	2	9	9	9	81
	3	16	16	16	256
	4	25	25	25	625
	5	36	36	36	1296
3D	1	8	8	8	64
	2	27	27	27	729
	3	64	64	64	4096
	4	125	125	125	15625
	5	216	216	216	46656

Table 3.3. Correlations of control points with elements and Gauss points for IGA

Gauss points per element: The integration rule $(p+1)^d$ is adopted here as well.

Control points influenced by GP/element: Each one-dimensional span is affected by p+1 basis functions. Therefore, one Gauss point that belongs to a two-dimensional or a three-dimensional element, influences $(p+1)^2$ or $(p+1)^3$ control points respectively.

Elements influencing a control point: Due to the overlapping of shape functions, each control point is influenced by $(p+1)^d$ elements.

Gauss points influencing a control point: Each control point intervenes in $(p+1)^d$ elements, while each one of them is integrated by $(p+1)^d$ quadrature points. Therefore, the number of Gauss points-control points correlations is equal to $(p+1)^{2d}$.

The number of control points affected by each Gauss point in IGA is the same as the number of influenced nodes per element in FEA. However, in FEA each Gauss point affects the nodes within its own element, while in IGA each Gauss point affects surroundings areas (range depending on p). Each control point is affected by more elements in IGA than in FEA and consequently by a lot more Gauss points. The following table shows the number of Gauss points influencing a control point/node with respect to p, demonstrating the growth rate.

Gauss points influencing a control point/node			
Problem type	IGA	FEA min	FEA max
1D	$(p+1)^2$	$(p+1)^1$	$2^1(p+1)^1$
2D	$(p+1)^4$	$(p+1)^2$	$2^2(p+1)^2$
3D	$(p+1)^6$	$(p+1)^3$	$2^3(p+1)^3$

Table 3.4. Number of Gauss points influencing a control point/node with respect to p

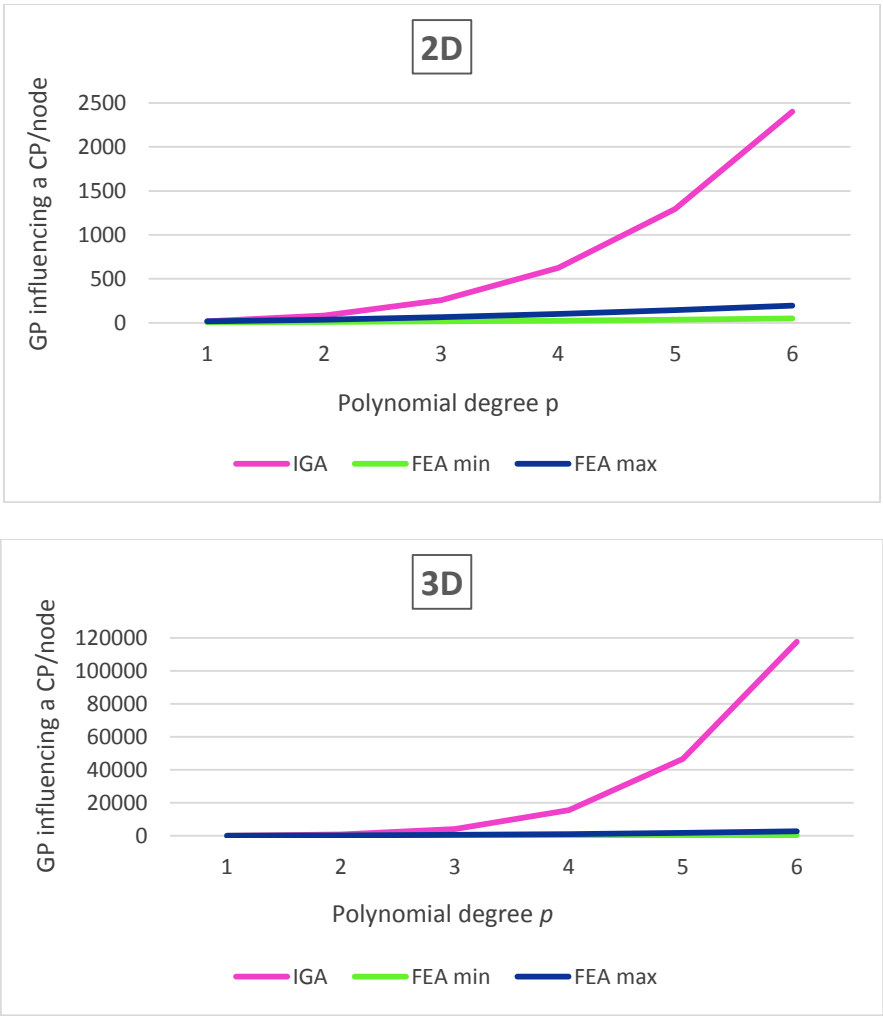


Fig. 3.7. Number of Gauss points influencing a control point/node with respect to *p*

3.3.4 Efficient quadrature

From the above section it can be concluded that the correlations between Gauss points and control points as well as the total number of Gauss points required is extremely increased in IGA compared to FEA. However Galerkin method neglects the important factor of continuity and that is why it is perfectly suitable for standard C^0-continuous finite elements.

In IGA basis functions are of continuity C^{p-m} across knots where p is the degree of the polynomial and m is the multiplicity of each knot. This property enables increased continuity unlike FEA C^0 and encourages smoothness across element boundaries, in case of knots with reduced multiplicity. Considering the already known equation that relates the number of knot values k with the number of control points n

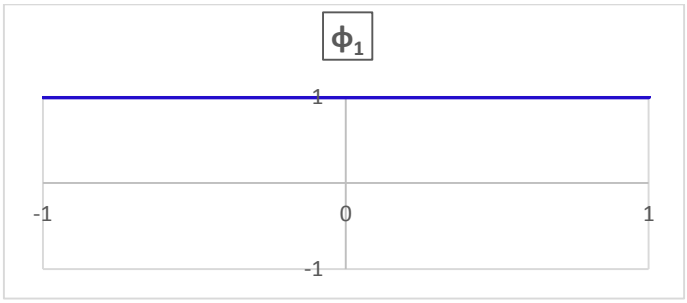
$$k = n + p + 1,$$

it can be shown that the higher continuity stands for less control points and therefore less basis functions.

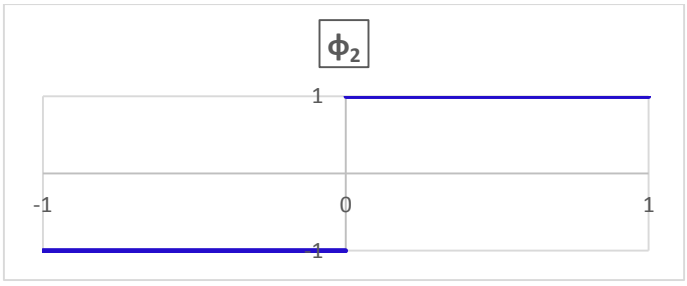
As it is suggested in [4] the number of quadrature points should depend mainly on number of degrees of freedom and less on polynomial degree *p*. A simple example is described, in order to illustrate that increased continuity results in improved quadrature rules.

Consider a biunit interval [-1,1] consisting of two unit subintervals ("elements") [-1,0] and [0,1]. A basis of piecewise quadratic polynomials defined on [-1,1], though with no continuity at $\{0\}$, could be composed from the following functions:

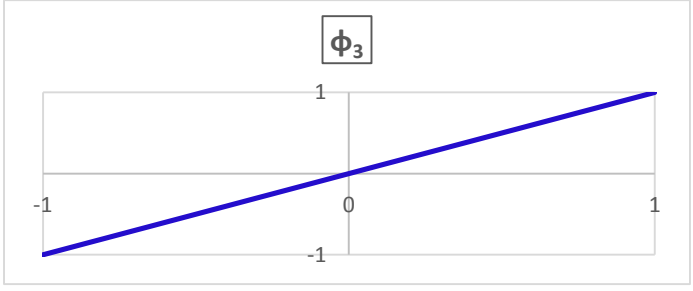
- $\varphi_1(\xi) = 1, \ \forall \xi \in [-1,1]$



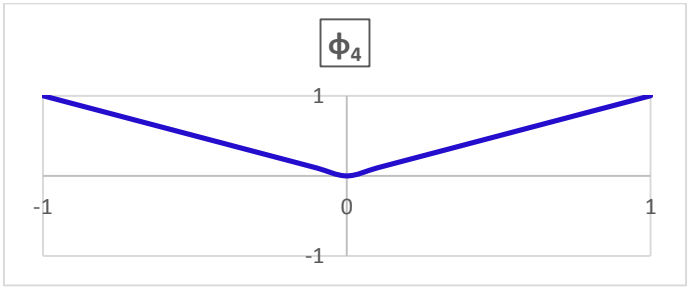
- $\varphi_2(\xi) = \begin{cases} -1, & \forall \xi \in [-1,0) \\ 1, & \forall \xi \in (0,1] \end{cases}$



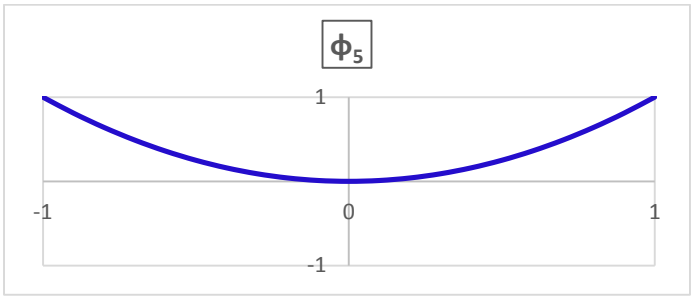
- $\varphi_3(\xi) = \xi, \ \forall \xi \in [-1,1]$



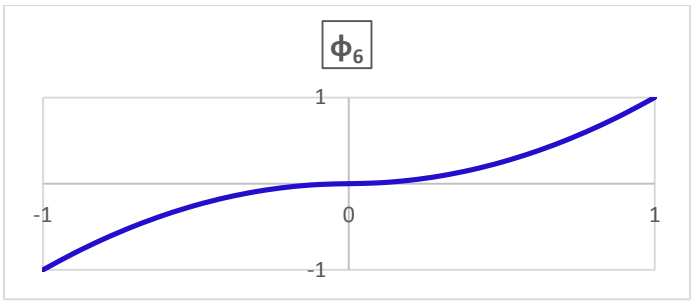
- $\varphi_4(\xi) = \begin{cases} -\xi, & \forall \xi \in [-1,0) \\ \xi, & \forall \xi \in (0,1] \end{cases}$



- $\varphi_5(\xi) = \xi^2, \quad \forall \xi \in [-1,1]$



- $\varphi_6(\xi) = \begin{cases} -\xi^2, & \forall \xi \in [-1,0) \\ \xi^2, & \forall \xi \in (0,1] \end{cases}$



φ_1, φ_4 and φ_5 are even functions, while φ_2, φ_3 and φ_6 are odd functions. The basis is discontinuous $\left(C^{-1}\right)$ at 0 because of φ_2. The number of required quadrature points is defined by applying the $\frac{p+2}{2}$ rule on each subinterval separately. Substitution of p with 2 yields two points on each subinterval, for the total of four quadrature points. With the removal of φ_2 C^0-continuity is achieved across 0 and now the piecewise quadratic basis has dimension five. According to the three point Gauss rule, in case of odd number of points as the case shown in Fig. 3.8, origin will automatically be one

Gauss point and the other Gauss points are symmetrically arranged with respect to the origin.

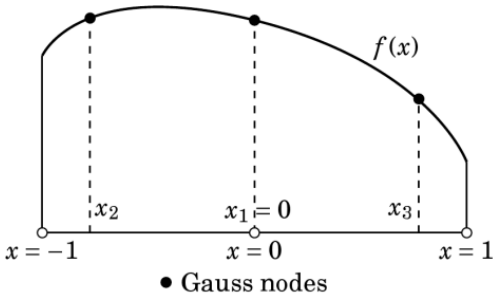


Fig. 3.8. Gauss quadrature with three points

For the continuity to be further increased at C^1, another basis function should be removed, specifically φ_4. Now the $\frac{p+2}{2}$ rule is directed at the whole interval [-1,1], considering that the remaining functions φ_1, φ_3 and φ_5 are continuous and not piecewise polynomials at their domain. As far as φ_6 is concerned, it should be noted that the integral of an odd function over a symmetric interval is equal to zero. Therefore, only two points can exactly integrate the basis of the four functions. This example demonstrates that increased continuity at 0 enables optimal quadrature rules with fewer quadrature points compared to a Gauss rule on each knot span.

Since the Galerkin method is the prevalent numerical procedure in FEA computations, its adjustment to IGA peculiarities was the first candidate to be studied and implemented. Even though it constitutes a reasonable choice, it does not mean that it is the most suitable one too. The fact that IGA ends up to denser system matrices and requires more computational effort should not be interpreted as a weakness of the new very promising method. It just indicates that Galerkin technique does not fit well at the increased overlapping and smoothness properties of IGA and direct the research to alternative numerical integration schemes that combine accuracy with computational efficiency.

3.4 Collocation

3.4.1 Test functions

As mentioned in [6] in the collocation method two sets of Dirac δ functions represent test functions $\omega_\Omega(x)$ and $\omega_\Gamma(x)$, which can be formally constructed as the limit of a sequence of smooth functions with compact support that converge to a distribution, satisfying the so-called sifting property

$$\int_{\Omega} g_{\Omega}(\boldsymbol{x})\,\delta_{\Omega}(\boldsymbol{x}-\boldsymbol{x}_i)\,d\Omega = g_{\Omega}(\boldsymbol{x}_i)$$
$$\int_{\Gamma} g_{\Gamma}(\boldsymbol{x})\,\delta_{\Gamma}(\boldsymbol{x}-\boldsymbol{x}_i)\,d\Gamma = g_{\Gamma}(\boldsymbol{x}_i)$$

provided that g_{Ω} is a continuous function about the point $\boldsymbol{x}_i \in \Omega$ and g_{Γ} is a continuous function on the boundary about the point $x_i \in \Gamma$.

The two sets of Dirac δ functions refer to interior points and points across boundary Γ_N separately.

$$\omega_{\Omega} = \sum_{i=1}^{k} \delta_{\Omega}(\boldsymbol{x}-\boldsymbol{x}_i)\hat{c}_i$$

are defined at *k* interior points in Ω with coordinates $\boldsymbol{x}_i,\ i=1,...,k$ while

$$\omega_{\Gamma} = \sum_{i=1}^{k} \delta_{\Gamma}(\boldsymbol{x}-\boldsymbol{x}_i)\hat{c}_i$$

are defined at *n-k* boundary points on Γ_N with coordinates $\boldsymbol{x}_i,\ i=k+1,...,n$, where *n* represents the total number of basis functions. Collocation points lying across Γ_D do not have to be taken into consideration since the Dirichlet boundary condition is enforced a priori. Substitution of the previous two equations into the weak form of Eq. (1) yields

$$\sum_{i=1}^{k}\hat{c}_i\left(L\left[\tilde{u}_D(\boldsymbol{x}_i)+\sum_{j=1}^{k}N_j(\boldsymbol{x}_i)c_j\right]-f(\boldsymbol{x}_i)\right)+\sum_{i=k+1}^{n}\hat{c}_i\left(\boldsymbol{n_i}\cdot D\sum_{j=k+1}^{n}\nabla N_j(\boldsymbol{x}_i)c_j-h(\boldsymbol{x}_i)\right)=0$$

Thanks to the sifting property of the Dirac δ test functions, the integrals are eliminated and thus collocation supports the discretization of the strong form of the governing partial differential equations. In other words collocation forces the strong form of the residual to be zero at specific points of the domain, which are collocation points.

3.4.2 Collocation points

The choice of suitable collocation points affects in a large scale the efficiency of the method. Literature suggests various options for NURBS-based collocation schemes such as orthogonal collocation on Gauss-type quadrature points, the maxima of spline basis functions, the Demko abscissae and the Greville abscissae. Demko points in particular are proved to be stable for any mesh and degree. They are the extrema of the Chebyshev splines and they are obtained by an iterative algorithm.

In the present study Greville abscissae is preferred, since the location of the points is defined in a simple way and it is very competitive from the aspect of accuracy. It is proved to be stable up to degree 3, while examples of instability appear for degrees higher than 19 concerning mainly non-uniform meshes.

Given a knot value vector $\Xi=\{\xi_1, \xi_2,\ldots,\xi_{n+p+1}\}$ Greville abscissae can be easily be computed as

$$\hat{\xi}_i = \frac{\xi_{i+1}+\ldots+\xi_{i+p}}{p}, \quad i=1,\ldots,n$$

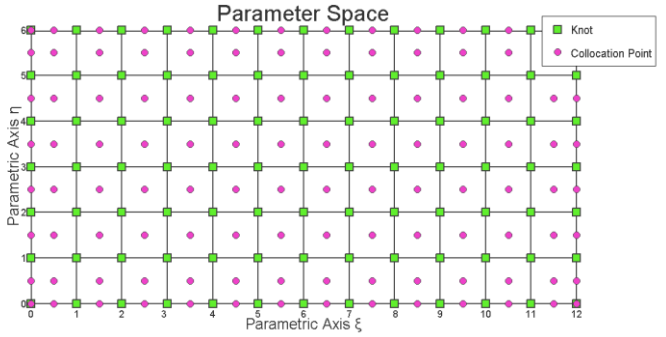
where n denotes the number of basis functions on axis ξ.

Analogously given a knot value vector $H=\{\eta_1, \eta_2,\ldots,\eta_{m+p+1}\}$ Greville points are defined as

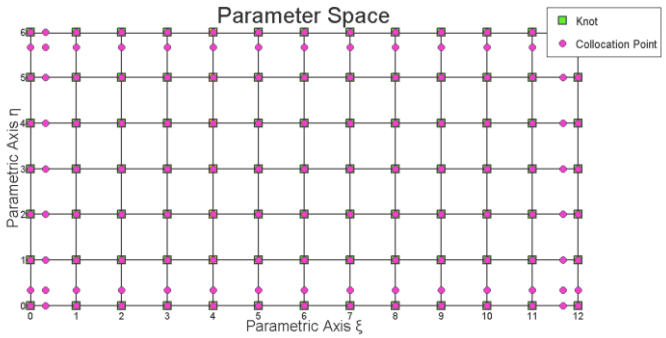
$$\hat{\eta}_j = \frac{\eta_{j+1}+\ldots+\eta_{j+p}}{p}, \quad j=1,\ldots,m$$

where m denotes the number of basis functions on axis η. Note that $\hat{\xi}_1 = \hat{\eta}_1 = 0$, $\hat{\xi}_m = \hat{\eta}_m = 1$, while all the remaining points belong to (0,1). The collocation points $\hat{\tau}_{i,j} \in \hat{\Omega}$ are defined by the tensor product structure

$$\hat{\tau}_{ij} = \left(\hat{\xi}_i, \hat{\eta}_j\right) \in \hat{\Omega}, \quad \tau_{ij} = F\left(\hat{\tau}_{ij}\right) \text{ for } i=1,\ldots,n \text{ and } j=1,\ldots,m.$$



(a)



(b)

Fig. 3.9. Collocation points in parameter space:
(a) p=2 (even), **(b)** p=3 (odd)

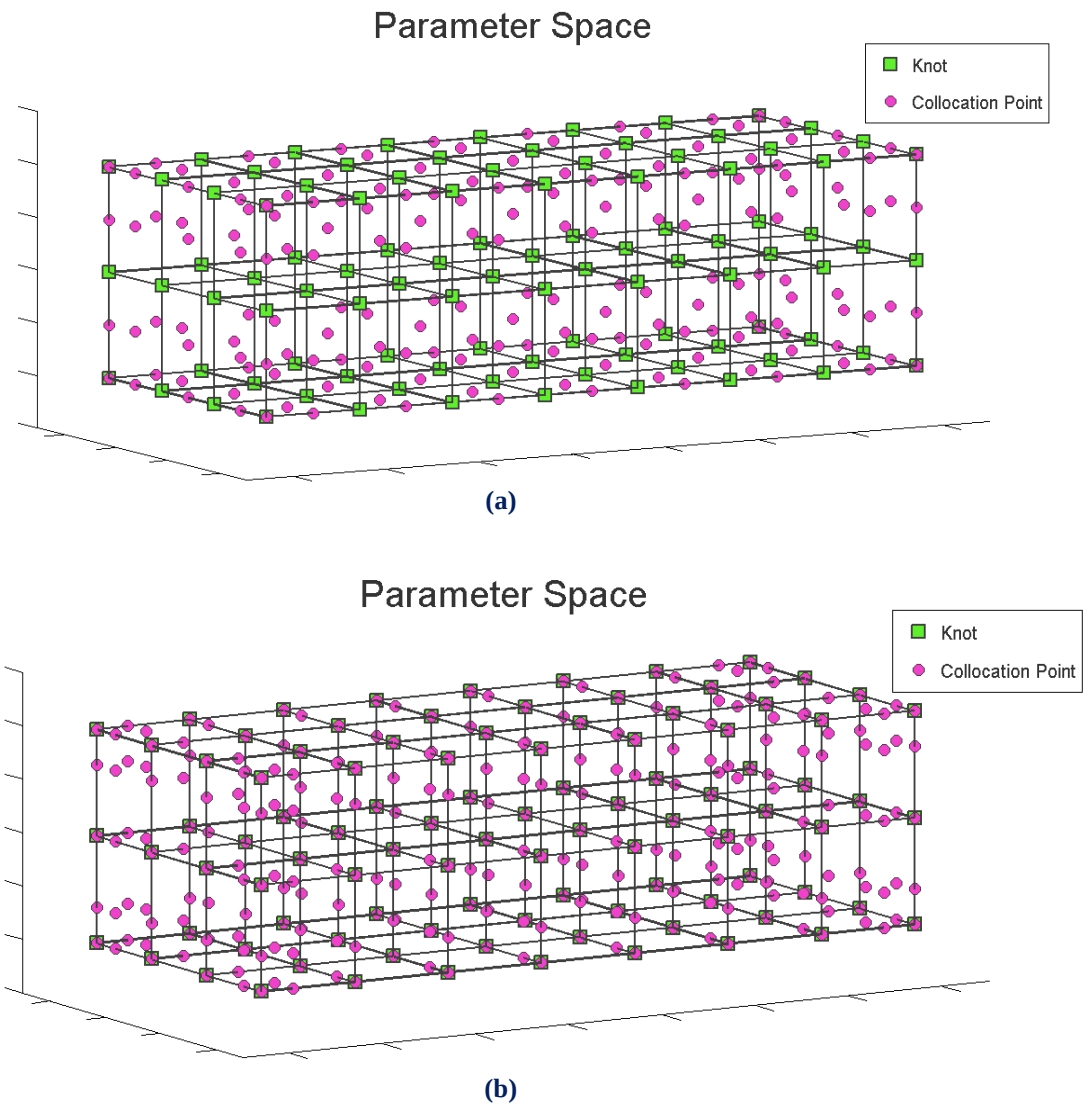


Fig. 3.10. Collocation points in parameter space:
(a) p=2 (even), **(b)** p=3 (odd)

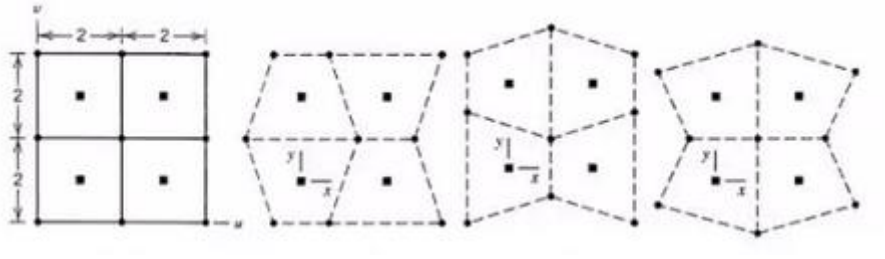
The isogeometric collocation problem is formulated as: find u_{nm} such that

$$L\left(u_{nm}\left(\tau_{ij}\right)\right) = f\left(\tau_{ij}\right), \quad i = 2,\ldots,n-1, \quad j = 2,\ldots,m-1$$
$$n \cdot D\nabla u_{nm}\left(\tau_{ij}\right) = h\left(\tau_{ij}\right), \quad (i,j) \in \{1,n\} \times \{1,\ldots,m\} \cup \{1,\ldots,n\} \times \{1,m\}$$

The scheme is based on the strong form of the partial differential equations and thus it demands the evaluation of higher derivatives, those that the operator of the problem indicates. As a matter of fact, it appeals to smoothness properties and inter-element continuity of shape functions used in isogeometric analysis.

The equal number of collocation and control points may remind the one-point quadrature, which is accused for rank deficiency of the discrete system. Sometimes one or more deformation modes happen to display zero strain at all quadrature points of an

element, when those are less than the required ones. The resulting $[k^e]$ will have no resistance to that deformation mode ending up to singularities. This phenomenon is the famous "hourglass mode" and it has created the need for ad-hoc stabilization techniques, which are time-consuming and still cannot guarantee an accurate result free of instabilities.



(a) **(b)** **(c)** **(d)**

Fig. 3.11. (a) Undeformed plane 2 by 2 four-node square elements, **(b)**, **(c)**, **(d)** "instability" displacement modes

However this is not the case for the collocation scheme. It can easily been shown that for quadratic and higher-order NURBS, with uniform knot vectors and a suitable choice of collocation points, the operator produced by collocation is rank sufficient in all dimensions.

4 Stiffness Matrix

4.1 Galerkin

4.1.1 Introduction

Based on the BVP defined in section 3.2 the stiffness matrix K and the load vector F are defined as

$$K_{ij} = \int_{\Omega} N_i \left(\boldsymbol{a} \cdot \nabla N_j \right) d\Omega + \int_{\Omega} \nabla N_i \cdot \left(D \nabla N_j \right) d\Omega$$

$$F_i = \int_{\Omega} N_i \, f d\Omega + \int_{\Gamma_N} N_i h d\Gamma$$

The general process for the global stiffness matrix assembly, as obtained from the finite element method, is shown in the following flow chart:

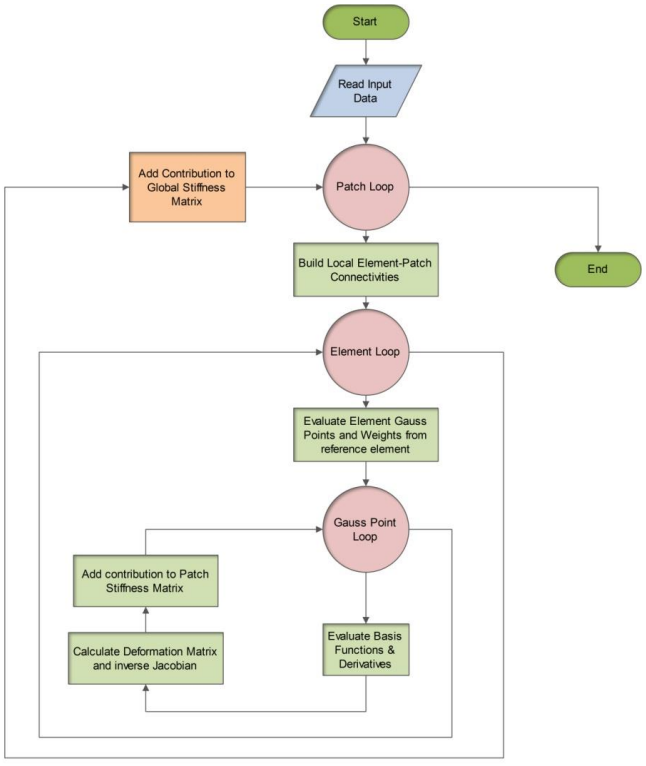


Fig. 4.1. Stiffness matrix assembly in finite element method

The procedure consists of three loops, one on a patch, one on an element and one on a Gauss point level. Indeed, stiffness matrices in both FEA and IGA are sparse matrices, since the support of each function is highly localized. As a consequence it is favorable to first build dense element stiffness matrices, which consider just the non-zero basis functions at the specific element (knot span) and they are composed from the contributions of the appropriate Gauss points.

In isogeometric analysis, however, the element loop can be neglected. Integration is enabled throughout the patch without building local element matrices separately, since B-spline functions are defined in a domain, which exceeds the limits of a single knot span. Therefore, the flow chart takes the form of:

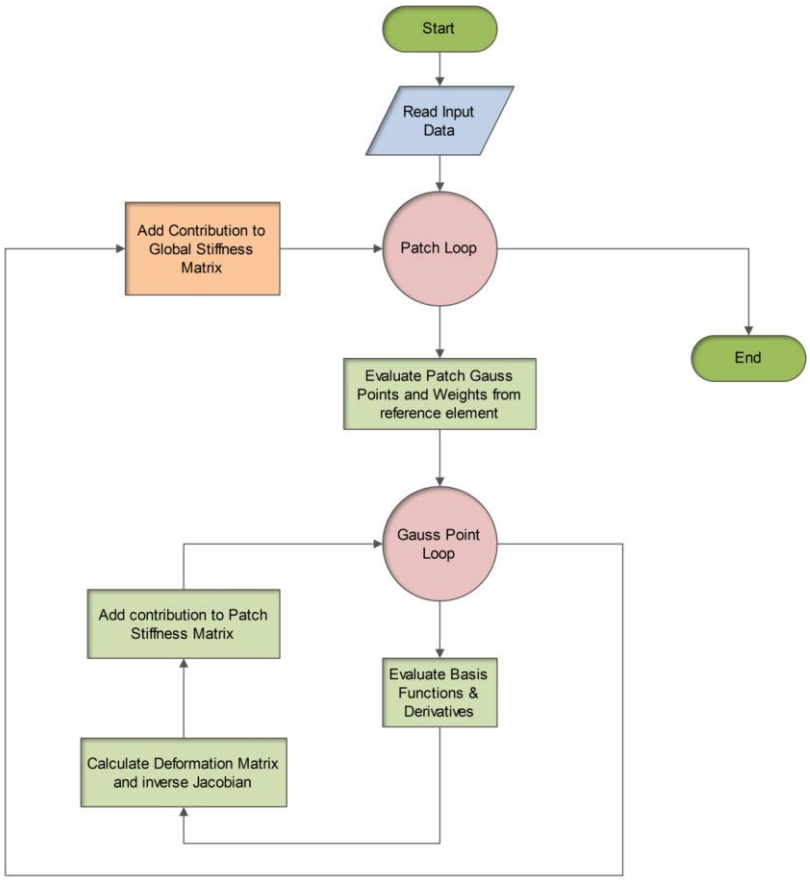


Fig. 4.2. Stiffness matrix assembly in isogeometric analysis

This alternative is released from the computation of the connectivity matrix, which is required for the transition from local to global level. Nevertheless, it is time-consuming and it is not recommended for advanced software technologies, as it implies big matrices that constrain storage, while the major part of their entries is equal to zero. It should be mentioned though that it serves well for research purposes, where a flexible quadrature code is needed in order to test and discover new methods and ideas.

4.1.2 Stiffness matrix 2D

Stiffness matrix should be expressed in terms of physical coordinates, while NURBS shape functions are initially defined with respect to parametric coordinates, thus a transition between the two spaces is required. For this purpose Jacobian matrix exists and it is evaluated as follows:

$$J=\begin{bmatrix} R_{1,\xi} & R_{2,\xi} & \dots & \dots & \dots & R_{N,\xi} \\ R_{1,\eta} & R_{2.\eta} & \dots & \dots & \dots & R_{N,\eta} \end{bmatrix} \begin{bmatrix} \hat{x}_1 & \hat{y}_1 \\ \hat{x}_2 & \hat{y}_2 \\ . & . \\ . & . \\ . & . \\ \hat{x}_N & \hat{y}_N \end{bmatrix}$$

where N is the total number of shape functions and $\{\hat{x}_i, \hat{y}_i\}$ are the physical coordinates of the i^{th} control point.

In order to calculate the deformation matrix for 2D problems, B_1 and B_2 have to be computed as usual.

$$B_1=\frac{1}{\det[J]}\begin{bmatrix} J_{22} & -J_{12} & 0 & 0 \\ 0 & 0 & -J_{21} & J_{11} \\ -J_{21} & J_{11} & J_{22} & -J_{12} \end{bmatrix}$$

$$B_2=\begin{bmatrix} R_{1,\xi} & 0 & R_{2,\xi} & 0 & \dots & \dots & \dots & R_{N,\xi} & 0 \\ R_{1,\eta} & 0 & R_{2,\eta} & 0 & \dots & \dots & \dots & R_{N,\eta} & 0 \\ 0 & R_{1,\xi} & 0 & R_{2,\xi} & \dots & \dots & \dots & 0 & R_{N,\xi} \\ 0 & R_{1,\eta} & 0 & R_{2,\eta} & \dots & \dots & \dots & 0 & R_{N,\eta} \end{bmatrix}$$

Having determined B_1 and B_2 the deformation matrix B is defined as

$$B= B_1 B_2$$

In order to evaluate the stiffness matrix, integration is required.

$$K=\int_{\xi_0}^{\xi_{n+p+1}}\int_{\eta_0}^{\eta_{m+q+1}} B^T E B t \det[J]\, d\eta d\xi$$

Numerical integration procedures for ξ, η lead to integration for tensor product Gauss points.

$$K=\sum_{i=1}^{GP\xi}\sum_{j=1}^{GP\eta} B^T E B t \det[J] w_i^{GP\xi} w_j^{GP\eta}$$

where *t* is the thickness of the cross-section, GP_ξ and GP_η is the total number of Gauss points per ξ and per η respectively and $w_i^{GP_\xi}$, $w_j^{GP_\eta}$ are the corresponding weights. E stands for elasticity matrix and its form depends on the plane stress or plane strain nature of the problem.

2D elasticity, plane stress:

$$E=\frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}$$

2D elasticity, plane strain:

$$E=\frac{E}{(1-\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$$

4.1.3 Stiffness matrix 3D

3D elasticity is merely the extension of 2D elasticity in all directions, with a complete stress field. Jacobian matrix is calculated in a similar way from the derivatives of the shape functions.

$$J=\begin{bmatrix} R_{1,\xi} & R_{2,\xi} & \dots & \dots & \dots & R_{N,\xi} \\ R_{1,\eta} & R_{2.\eta} & \dots & \dots & \dots & R_{N,\eta} \\ R_{1,\zeta} & R_{2,\zeta} & \dots & \dots & \dots & R_{N,\zeta} \end{bmatrix}\begin{bmatrix} \hat{x}_1 & \hat{y}_1 & \hat{z}_1 \\ \hat{x}_2 & \hat{y}_2 & \hat{z}_2 \\ . & . & . \\ . & . & . \\ . & . & . \\ \hat{x}_N & \hat{y}_N & \hat{z}_N \end{bmatrix}$$

where *N* is the total number of shape functions and $\{\hat{x}_i, \hat{y}_i, \hat{z}_i\}$ are the physical coordinates of the i[th] control point.

The inverse of the Jacobian Matrix is formulated as:

$$J^{-1}=\begin{bmatrix} J_{11}^* & J_{12}^* & J_{13}^* \\ J_{21}^* & J_{22}^* & J_{23}^* \\ J_{31}^* & J_{32}^* & J_{33}^* \end{bmatrix}$$

and it is used at the evaluation of B_1:

$$B_1=\begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{21} & J^*_{22} & J^*_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} \\ J^*_{21} & J^*_{22} & J^*_{23} & J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} & J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} & 0 & 0 & 0 & J^*_{11} & J^*_{12} & J^*_{13} \end{bmatrix}$$

As for B_2:

$$B_2=\begin{bmatrix} R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & 0 & \dots & \dots & \dots & R_{N,\xi} & 0 & 0 \\ R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & 0 & \dots & \dots & \dots & R_{N,\eta} & 0 & 0 \\ R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & 0 & \dots & \dots & \dots & R_{N,\zeta} & 0 & 0 \\ 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & \dots & \dots & \dots & 0 & R_{N,\xi} & 0 \\ 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & \dots & \dots & \dots & 0 & R_{N,\eta} & 0 \\ 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & \dots & \dots & \dots & 0 & R_{N,\zeta} & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & \dots & \dots & \dots & 0 & 0 & R_{N,\xi} \\ 0 & 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & \dots & \dots & \dots & 0 & 0 & R_{N,\eta} \\ 0 & 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & \dots & \dots & \dots & 0 & 0 & R_{N,\zeta} \end{bmatrix}$$

Deformation matrix for 3D elasticity is calculated as:

$$B=B_1\,B_2$$

The corresponding stiffness matrix is produced by integration

$$K=\int_{\xi_0}^{\xi_{n+p+1}}\int_{\eta_0}^{\eta_{m+q+1}}\int_{\zeta_0}^{\zeta_{l+r+1}} B^T E B \det\left[J\right] d\zeta d\eta d\xi$$

Numerical integration is used in 3D as well

$$K=\sum_{i=1}^{GP\xi}\sum_{j=1}^{GP\eta}\sum_{k=1}^{GP\zeta} B^T E B \det\left[J\right] w_i^{GP\xi} w_j^{GP\eta} w_k^{GP\zeta}$$

where GP_ξ, GP_η, and GP_ζ, is the total number of Gauss points per ξ, per η and per ζ respectively and $w_i^{GP\xi}$, $w_j^{GP\eta}$ and $w_k^{GP\zeta}$ are the corresponding weights.

Elasticity matrix E in 3-dimensional cases is of the form

$$E=\frac{E}{(1-\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$$

4.2 Collocation

4.2.1 Introduction

According to [6] the elements of the stiffness matrix *K* and load vector *F* at collocation method are defined as

$$K_{ij}=\begin{cases} L\left(N_j(\boldsymbol{x}_i)\right), & for\ 1\le i\le k, \\ \boldsymbol{n}_i\cdot D\nabla N_j(\boldsymbol{x}_i), & for\ k+1\le i\le n \end{cases}$$

$$F_i=\begin{cases} -L\left(\widetilde{u_D}(\boldsymbol{x}_i)\right)+f(\boldsymbol{x}_i), & for\ 1\le i\le k, \\ -\boldsymbol{n}_i\cdot D\nabla u_D(\boldsymbol{x}_i)+h(\boldsymbol{x}_i), & for\ k+1\le i\le n \end{cases}$$

where *n* represents the total number of basis functions, *k* the number of interior points in Ω and *n-k* the number of boundary points on Γ_N.

The rows of stiffness matrix correspond to collocation points, while columns refer to shape functions. Interior points, where the PDE is enforced, come first to computations, while boundary points, where flux condition is enforced, occupy the last rows. It should be noted that stiffness matrix is generally non-symmetric.

As far as the load vector is concerned, it is worth-mentioning that it does not require demanding calculations. Provided that initial displacement $\tilde{u}_D(\boldsymbol{x})$ is equal to zero, it simply includes the value of the distribution of the body force *f* or the surface traction *h* at each collocation point.

In case of elasticity problems the formulation of stiffness matrix is based on Navier's equations for elasticity, which are given expressed in terms of displacement.

$$\mu\nabla^2 u+(\lambda+\mu)\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)+F_x=0$$
$$\mu\nabla^2 v+(\lambda+\mu)\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)+F_y=0$$
$$\mu\nabla^2 w+(\lambda+\mu)\frac{\partial}{\partial z}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)+F_z=0$$

where λ and μ are called Lamé constants and their value depends on Young's modulus E and Poisson's ratio ν.

$$\lambda=\frac{\nu E}{(1+\nu)(1-2\nu)}$$
$$\mu=\frac{E}{2(1+\nu)}$$

Considering the following general form of a boundary value problem

$Lu = f \ \ in\ \Omega$
$B^h u = h \ \ on\ \partial\Omega^h$
$B^g u = g \ \ on\ \partial\Omega^g$

where Ω is the problem domain, $\partial\Omega^h$ is the Neumann boundary and $\partial\Omega^g$ is the Dirichlet boundary, it becomes clear that differential operators L and B^h compose stiffness matrix. Operator B^g is already known since it is equal in every occasion with the identity matrix I.

4.2.2 Stiffness matrix 2D

Both standard Galerkin and collocation require the computation of first derivatives in the physical space. The procedure for the definition of the Jacobian matrix has already been proven in section 4.1.2. The final form of the matrix is

$$J=\begin{pmatrix} R_{j,\xi}\hat{x}_j & R_{j,\eta}\hat{x}_j \\ R_{j,\xi}\hat{y}_j & R_{j,\eta}\hat{y}_j \end{pmatrix}$$

considering that index j runs over all shape functions and $\hat{\boldsymbol{x}}_j = \{\hat{x}_j, \hat{y}_j\}^T$ denote the physical coordinates of control points. The transformation from parameter to physical space is realized through the equation

$$J\mathbf{R}_x = \mathbf{R}_\xi$$

where $\boldsymbol{R}_x$ and $\boldsymbol{R}_\xi$ denote the vectors of derivatives $\{R_x, R_y\}^T$ and $\{R_\xi, R_\eta\}^T$ with respect to physical and parametric coordinates, respectively.

In collocation technique second derivatives of shape functions additionally have to be transformed in the physical space. For this purpose the hessian matrix *H* and the matrix of squared first derivatives J_2 are required. The matrix of squared first derivatives contains various combinations of squared entries of the Jacobian matrix *J*, as it can be seen from the following definitions.

$$H=\begin{pmatrix} R_{j,\xi\xi}\hat{x}_j & R_{j,\xi\eta}\hat{x}_j & R_{j,\eta\eta}\hat{x}_j \\ R_{j,\xi\xi}\hat{y}_j & R_{j,\xi\eta}\hat{y}_j & R_{j,\eta\eta}\hat{y}_j \end{pmatrix}$$

$$J_2=\begin{pmatrix} J_{11}^2 & J_{11}J_{12} & J_{12}^2 \\ 2J_{11}J_{21} & J_{11}J_{22}+J_{12}J_{21} & 2J_{12}J_{22} \\ J_{21}^2 & J_{21}J_{22} & J_{22}^2 \end{pmatrix}$$

The transformation from parameter to physical space is realized through the equation

$$J_2\mathbf{R}_{xx}=\mathbf{R}_{\xi\xi}-H^T\mathbf{R}_x$$

where $\boldsymbol{R}_{xx}$ and $\boldsymbol{R}_{\xi\xi}$ represent the vectors $\{R_{xx}, R_{xy,}, R_{yy}\}^T$ and $\{R_{\xi\xi}, R_{\xi\eta}, R_{\eta\eta}\}^T$ of second and mixed derivatives with respect to physical and parametric coordinates, respectively.

After first and second derivatives of NURBS shape functions have been defined in the physical space, the formulation of differential operators *L* and B^h is the next step.

$$L=\begin{pmatrix} (\lambda+2\mu)\dfrac{\partial^2}{\partial x^2}+\mu\dfrac{\partial^2}{\partial y^2} & (\lambda+\mu)\dfrac{\partial^2}{\partial x\partial y} \\ (\lambda+\mu)\dfrac{\partial^2}{\partial x\partial y} & (\lambda+2\mu)\dfrac{\partial^2}{\partial y^2}+\mu\dfrac{\partial^2}{\partial x^2} \end{pmatrix}$$

$$B^h=\begin{pmatrix} (\lambda+2\mu)n_x\dfrac{\partial}{\partial x}+\mu n_y\dfrac{\partial}{\partial y} & \lambda n_x\dfrac{\partial}{\partial y}+\mu n_y\dfrac{\partial}{\partial x} \\ \lambda n_y\dfrac{\partial}{\partial x}+\mu n_x\dfrac{\partial}{\partial y} & (\lambda+2\mu)n_y\dfrac{\partial}{\partial y}+\mu n_x\dfrac{\partial}{\partial x} \end{pmatrix}$$

where n_x and n_y are the direction cosines of the outward normal $\boldsymbol{n}$ along Γ. Further information about the formation of B^h and the well definition of $\boldsymbol{n}$ is provided in section 7.2.2.

4.2.3 Stiffness matrix 3D

In 3-dimensional cases, the same pattern has to be followed enriched with the required adjustments. Jacobian matrix *J*, Hessian matrix *H* and the matrix of squared first derivatives J_2 have as well to be computed for the transformation to the physical space.

$$J = \begin{pmatrix} R_{j,\xi}\hat{x}_j & R_{j,\xi}\hat{y}_j & R_{j,\xi}\hat{z}_j \\ R_{j,\eta}\hat{x}_j & R_{j,\eta}\hat{y}_j & R_{j,\eta}\hat{z}_j \\ R_{j,\zeta}\hat{x}_j & R_{j,\zeta}\hat{y}_j & R_{j,\zeta}\hat{z}_j \end{pmatrix}$$

$$H = \begin{pmatrix} R_{j,\xi\xi}\hat{x}_j & R_{j,\eta\eta}\hat{x}_j & R_{j,\zeta\zeta}\hat{x}_j & R_{j,\xi\eta}\hat{x}_j & R_{j,\xi\zeta}\hat{x}_j & R_{j,\eta\zeta}\hat{x}_j \\ R_{j,\xi\xi}\hat{y}_j & R_{j,\eta\eta}\hat{y}_j & R_{j,\zeta\zeta}\hat{y}_j & R_{j,\xi\eta}\hat{y}_j & R_{j,\xi\zeta}\hat{y}_j & R_{j,\eta\zeta}\hat{y}_j \\ R_{j,\xi\xi}\hat{z}_j & R_{j,\eta\eta}\hat{z}_j & R_{j,\zeta\zeta}\hat{z}_j & R_{j,\xi\eta}\hat{z}_j & R_{j,\xi\zeta}\hat{z}_j & R_{j,\eta\zeta}\hat{z}_j \end{pmatrix}$$

$$J_2 = \begin{pmatrix} J_{11}^2 & J_{12}^2 & J_{13}^2 & 2J_{11}J_{12} & 2J_{11}J_{13} & 2J_{12}J_{13} \\ J_{21}^2 & J_{22}^2 & J_{23}^2 & 2J_{21}J_{22} & 2J_{21}J_{23} & 2J_{22}J_{23} \\ J_{31}^2 & J_{32}^2 & J_{33}^2 & 2J_{31}J_{32} & 2J_{31}J_{33} & 2J_{32}J_{33} \\ J_{11}J_{21} & J_{12}J_{22} & J_{13}J_{23} & \left(J_{11}J_{22}+J_{21}J_{12}\right) & \left(J_{11}J_{23}+J_{21}J_{13}\right) & \left(J_{12}J_{23}+J_{22}J_{13}\right) \\ J_{11}J_{31} & J_{12}J_{32} & J_{13}J_{33} & \left(J_{11}J_{32}+J_{31}J_{12}\right) & \left(J_{11}J_{33}+J_{31}J_{13}\right) & \left(J_{12}J_{33}+J_{32}J_{13}\right) \\ J_{21}J_{31} & J_{22}J_{32} & J_{23}J_{33} & \left(J_{21}J_{32}+J_{31}J_{22}\right) & \left(J_{21}J_{33}+J_{31}J_{23}\right) & \left(J_{22}J_{33}+J_{32}J_{23}\right) \end{pmatrix}$$

The above matrices are used in the following equations

$$\mathrm{J}\mathbf{R}_{\mathrm{x}} = \mathbf{R}_{\xi}$$

$$\mathrm{J}_2\mathbf{R}_{\mathrm{xx}} = \mathbf{R}_{\xi\xi} - \mathrm{H}^{\mathrm{T}}\mathbf{R}_{\mathrm{x}}$$

$\boldsymbol{R}_x = \{R_x, R_y, R_z\}^T$ and $\boldsymbol{R}_\xi = \{R_\xi, R_\eta, R_\zeta\}^T$ denote the vectors of first derivatives expressed in physical and parametric coordinates, respectively. In a similar way, $\boldsymbol{R}_{xx} = \{R_{xx}, R_{yy}, R_{zz}, R_{xy}, R_{xz}, R_{yz}\}^T$ and $\boldsymbol{R}_{\xi\xi} = \{R_{\xi\xi}, R_{\eta\eta}, R_{\zeta\zeta}, R_{\xi\eta}, R_{\xi\zeta}, R_{\eta\zeta}\}^T$ denote the vectors of second derivatives expressed in physical and parametric coordinates, respectively.

Differential operators L and B^h in 3D cases are defined as

$$L=\begin{pmatrix} (\lambda+2\mu)\frac{\partial^2}{\partial x^2}+\mu\frac{\partial^2}{\partial y^2}+\mu\frac{\partial^2}{\partial z^2} & (\lambda+\mu)\frac{\partial^2}{\partial x\partial y} & (\lambda+\mu)\frac{\partial^2}{\partial x\partial z} \\ (\lambda+\mu)\frac{\partial^2}{\partial x\partial y} & (\lambda+2\mu)\frac{\partial^2}{\partial y^2}+\mu\frac{\partial^2}{\partial x^2}+\mu\frac{\partial^2}{\partial z^2} & (\lambda+\mu)\frac{\partial^2}{\partial y\partial z} \\ (\lambda+\mu)\frac{\partial^2}{\partial x\partial z} & (\lambda+\mu)\frac{\partial^2}{\partial y\partial z} & (\lambda+2\mu)\frac{\partial^2}{\partial z^2}+\mu\frac{\partial^2}{\partial x^2}+\mu\frac{\partial^2}{\partial y^2} \end{pmatrix}$$

$$B^h=\begin{pmatrix} (\lambda+2\mu)n_x\frac{\partial}{\partial x}+\mu n_y\frac{\partial}{\partial y}+\mu n_z\frac{\partial}{\partial z} & \lambda n_x\frac{\partial}{\partial y}+\mu n_y\frac{\partial}{\partial x} & \lambda n_x\frac{\partial}{\partial z}+\mu n_z\frac{\partial}{\partial x} \\ \lambda n_y\frac{\partial}{\partial x}+\mu n_x\frac{\partial}{\partial y} & (\lambda+2\mu)n_y\frac{\partial}{\partial y}+\mu n_x\frac{\partial}{\partial x}+\mu n_z\frac{\partial}{\partial z} & \lambda n_y\frac{\partial}{\partial z}+\mu n_z\frac{\partial}{\partial y} \\ \lambda n_z\frac{\partial}{\partial x}+\mu n_x\frac{\partial}{\partial z} & \lambda n_z\frac{\partial}{\partial y}+\mu n_y\frac{\partial}{\partial z} & (\lambda+2\mu)n_z\frac{\partial}{\partial z}+\mu n_x\frac{\partial}{\partial x}+\mu n_y\frac{\partial}{\partial y} \end{pmatrix}$$

4.3 Comparison

4.3.1 Computational cost

After the methodology of the two techniques has been described, a comparison of the cost required for the formation and assembly of stiffness matrix is possible. A full oversight can be provided by counting the number of operations, with respect to the polynomial degree *p*, at one quadrature/collocation point for IGA Galerkin (IGA-G) and IGA Collocation (IGA-C) respectively. For confusions to be avoided, each multiplication and each addition is considered as one full floating point operation. In this section each collocation point is an interior one and not a point on the Neumann boundary, since the computation of interior points involves second derivatives and thus it is more time-consuming. Results for FEA Galerkin (FEA-G) are also presented for the sake of integrity. It should be noted that the tables of this section are based on [6].

Consider an elasticity problem in one, two and three dimensions. Table 4.1 presents the number of operations at one quadrature and one collocation point for all the successive steps from the evaluation of basis function in tensor product form to the transformation of first and second derivatives in physical coordinates. The cost for computation of basis function values in parametric directions is neglected, assuming it is small and comparable between methods.

d	IGA-C	IGA-G	FEA-G
	1. Form tensor product:		
1	-	-	-
2	$6(p+1)^2$	$3(p+1)^2$	$3(p+1)^2$
3	$20(p+1)^3$	$8(p+1)^3$	$8(p+1)^3$
	2. Multiply each B-spline basis function with corresponding weight:		
1	$3(p+1)$	$2(p+1)$	-
2	$6(p+1)^2$	$3(p+1)^2$	-
3	$10(p+1)^3$	$4(p+1)^3$	-
	3. Compute sums of B-spline basis functions:		
1	$3(p+1)$	$2(p+1)$	-
2	$6(p+1)^2$	$3(p+1)^2$	-
3	$10(p+1)^3$	$4(p+1)^3$	-
	4. Compute NURBS basis functions and its derivatives with respect to parametric coordinates:		
	Function, first and second derivatives	Function and first derivatives	
1	$21(p+1)$	$6(p+1)$	-
2	$57(p+1)^2$	$11(p+1)^2$	-
3	$109(p+1)^3$	$16(p+1)^3$	-
	5. Compute Jacobian matrix:		
1	$2(p+1)$	$2(p+1)$	$2(p+1)$
2	$8(p+1)^2$	$8(p+1)^2$	$8(p+1)^2$
3	$18(p+1)^3$	$18(p+1)^3$	$18(p+1)^3$
	6. Compute Hessian and the matrix of squared first derivatives:		
1	$2(p+1)+2$	-	-
2	$12(p+1)^2+13$	-	-
3	$36(p+1)^3+63$	-	-
	7. Solve for first derivatives:		
1	$(p+1)$	$(p+1)$	$(p+1)$
2	$5(p+1)^2+4$	$5(p+1)^2+4$	$5(p+1)^2+4$
3	$12(p+1)^3+20$	$12(p+1)^3+20$	$12(p+1)^3+20$
	8. Solve for second derivatives:		
1	$3(p+1)$	-	-
2	$24(p+1)^2+20$	-	-
3	$87(p+1)^3+140$	-	-
	Total number of operations per point:		
1	$35(p+1)+2$	$13(p+1)$	$3(p+1)$
2	$124(p+1)^2+37$	$33(p+1)^2+4$	$16(p+1)^2+4$
3	$302(p+1)^3+223$	$62(p+1)^3+20$	$35(p+1)^3+20$

Table 4.1. Operations for the computation of NURBS basis functions per collocation/quadrature point in IGA-C/IGA-G, FEA-G

It is obvious that second derivatives initiate new steps and more complicated calculations in case of IGA-C, increasing the total number of operations per point in comparison with IGA-G and FEA-G. However this divergence is not representative of the final cost for stiffness matrix formulation. As it can be seen in Table 4.1, IGA-C absorbs the procedure of assembly and all the computations correlated to the product $\boldsymbol{B}^T\boldsymbol{E}\boldsymbol{B}|J|w$, that turn out to be the main expense for the Galerkin method.

So far the standard technique is to first build the local stiffness matrix of each element by adding the contributions of all Gauss points affected by the element. Then each local matrix is appended to the global stiffness matrix in the appropriate positions. This process refers to the steps of formulation and assembly. In collocation the operations for one collocation point correspond to the entries of one row of the final system matrix, thus the assembly step is skipped. The total number of operations shown in the last rows of Table 4.2 confirms that the formation of stiffness matrix in IGA-G and FEA-G is orders of magnitude more expensive than in IGA-C.

d	IGA-C	IGA-G	FEA-G
	1. Operations per point transferred from Table 1:		
1	$35(p+1)+2$	$13(p+1)$	$3(p+1)$
2	$124(p+1)^2+37$	$33(p+1)^2+4$	$16(p+1)^2+4$
3	$302(p+1)^3+223$	$62(p+1)^3+20$	$38(p+1)^3+20$
	2. Set up local stiffness matrix:		
	No local stiffness matrix required.	Evaluate $\boldsymbol{B}^T\boldsymbol{E}\boldsymbol{B}\|J\|w$	
1	-	$2(p+1)^2+(p+1)$	$2(p+1)^2+(p+1)$
2	-	$20(p+1)^4+36(p+1)^2$	$20(p+1)^4+36(p+1)^2$
3	-	$99(p+1)^6+216(p+1)^3$	$99(p+1)^6+216(p+1)^3$
	3. Evaluate Navier's eqs. on global level	3. Add to local element stiffness matrix: (Final assembly to global matrix is neglected)	
1	$(p+1)$	$(p+1)^2$	$(p+1)^2$
2	$12(p+1)^2$	$4(p+1)^4$	$4(p+1)^4$
3	$21(p+1)^3$	$9(p+1)^6$	$9(p+1)^6$
	Total operations:		
1	$36(p+1)+2$	$3(p+1)^2+14(p+1)$	$3(p+1)^2+4(p+1)$
2	$136(p+1)^2+37$	$24(p+1)^4+69(p+1)^2+4$	$24(p+1)^4+52(p+1)^2+4$
3	$323(p+1)^3+223$	$108(p+1)^6+278(p+1)^3+20$	$108(p+1)^6+254(p+1)^3+20$

Table 4.2. Operations for the formation and assembly of the stiffness matrix per collocation/quadrature point in IGA-C/IGA-G, FEA-G

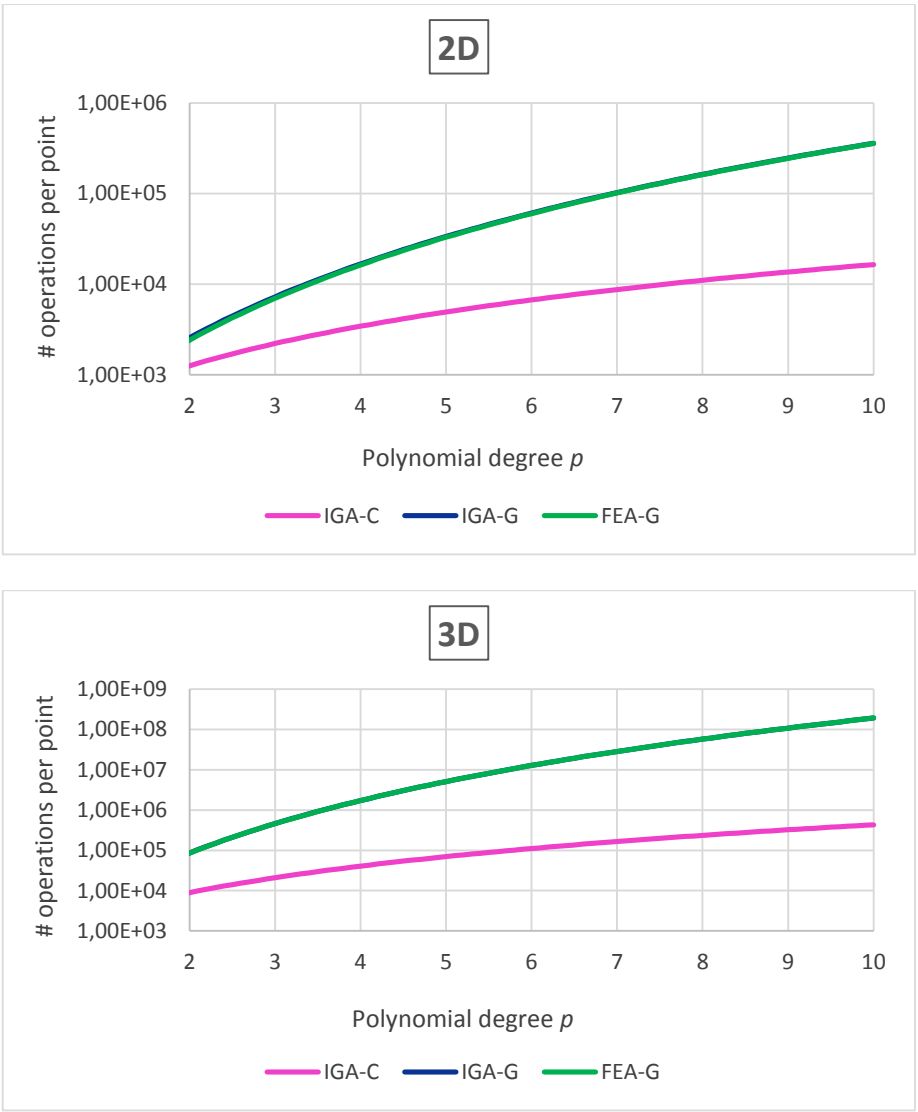


Fig. 4.3. Number of operations for the formation of the stiffness matrix per collocation/quadrature point

Table 4.2 refers to the required operations for stiffness matrix formulation. The number of operations per quadrature/collocation point should be multiplied with the total number of quadrature/collocation points respectively and the overall cost will arise. The total number of collocation points in a model discretization is equal to N^d (where N the number of control points and d the dimension of the problem) and the total number of quadrature points is equal to $\left(N-p\right)^d \left(p+1\right)^d$. Should these counts be expressed in terms of n, p and d, where n is the number of elements, we will have

	Number of quadrature/collocation points in the model discretization
IGA-C	$(n+p)^d$
IGA-G	$n^d(p+1)^d$
FEA-G	$n^d(p+1)^d$

4.3.2 Bandwidth

In IGA-G as in FEA-G as well, the bandwidth of the stiffness matrix depends on the interaction of basis functions. Each one of them has a domain of influence of p+1 knot value spans. These knot value spans, however, belong to the domain of influence of other basis functions too. Therefore, each function shares support with 2p+1 functions including itself. This statement is illustrated in Fig. 4.4 depicting some cubic B-spline basis functions. As it can be seen the function in green has overlapping support with the functions in pink as well as with itself, satisfying the count 2p+1=7.

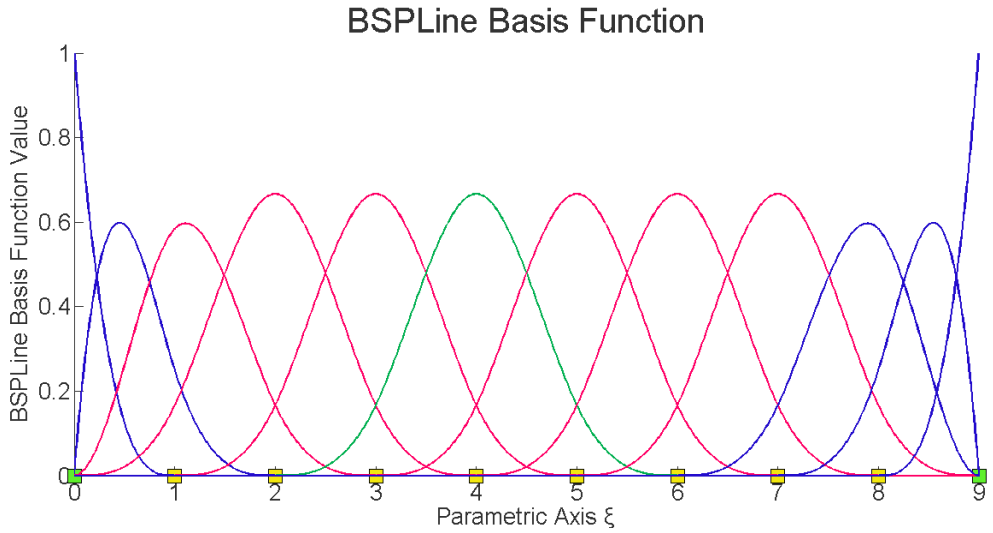
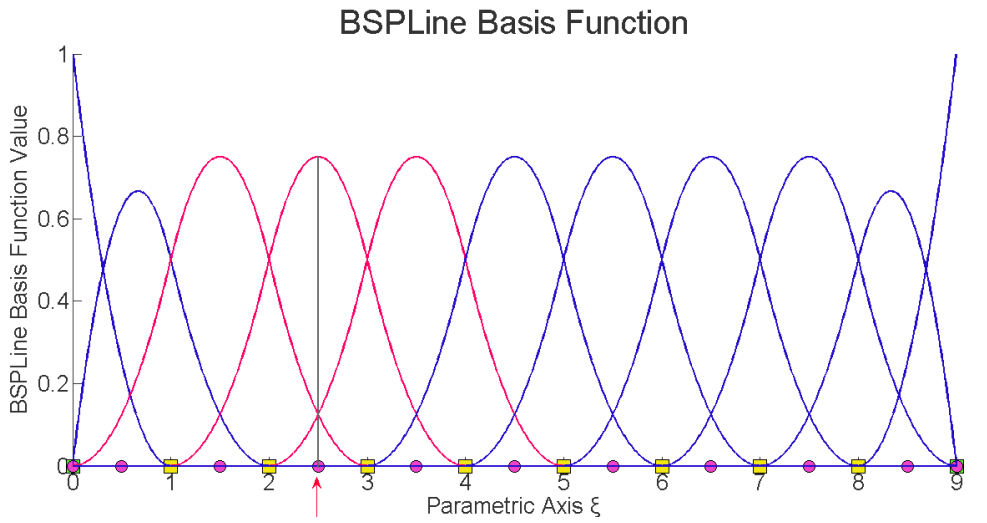


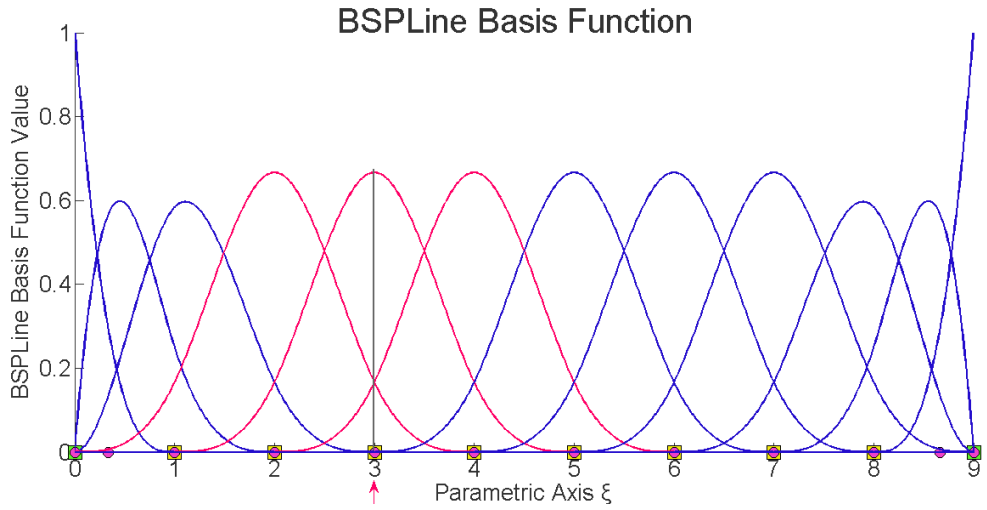
Fig. 4.4. Overlapping support of basis functions
Ξ={0 0 0 0 1 2 3 4 5 6 7 8 9 9 9 9}

Provided that p is increased, more degrees of freedom will collaborate resulting in larger bandwidth. Finally, the bandwidth of the Galerkin method is equal to 2p+1 for one-dimensional cases and $(2p+1)^d$ for 2D and 3D problems.

In collocation, however, the bandwidth is equal to p or p+1 for p odd or even, respectively, in one-dimensional cases. In stiffness matrices that collocation yields the rows correspond to collocation points and the columns to basis functions, thus the bandwidth is not determined by functions' interaction. What matters in that occasion is the number of functions that affect through their first and second derivatives each collocation point. The Greville points are located in the center of a knot span for even p and directly at a knot for odd p, so the number of non-zero spline basis functions at a collocation point is p+1 in the first case and p in the second, as it is shown in Fig. 4.5. This means that the same number of non-zero entries at one row of the stiffness matrix corresponds both to an even degree p and to the next higher degree p+1.



(a)



(b)

Fig. 4.5. Number of non-zero B-spline basis function over a collocation point
(a) p=2 (even), **(b)** p=3 (odd)

5 External Loads

5.1 External load vector

5.1.1 Galerkin

From a general point of view, the procedure for the formation of the external load vector in IGA follows the same steps as in FEM. In both methods, external loads of the structure should finally be attributed to the degrees of freedom. The main difference lies on the fact that in finite element analysis, degrees of freedom correlate with the nodes, which are material points, while in isogeometric analysis they refer to control points, which do not belong to the structure. In particular, they are placed around the object and define its geometry. Consequently, provided that external loads cannot act on control points immediately, an appropriate transfer through shape functions is necessary. Distributed loads *f* have to be transformed into equivalent concentrated loads by integration for 2D and 3D cases respectively:

$$F=\int_{\xi_0}^{\xi_{n+p+1}}\int_{\eta_0}^{\eta_{m+q+1}} R f \det\left[J\right] d\eta d\xi$$

$$F=\int_{\xi_0}^{\xi_{n+p+1}}\int_{\eta_0}^{\eta_{m+q+1}}\int_{\zeta_0}^{\zeta_{l+r+1}} R f \det\left[J\right] d\zeta d\eta d\xi$$

Quadrature rules are adopted for the elimination of integrals in the equations above. A new set of Gauss points is defined for each external load case, different from the one used in the stiffness matrix formulation. Stiffness matrix requires quadrature points that extend over the whole area of the interior of the patch, without the boundaries included. In external load cases, on the other hand, Gauss points are defined exclusively over the domain of the load distribution. For surface loads, Gauss points lying on the relevant boundary are used, while for body forces, quadrature rules address to the internal surface or volume of the structure for two-dimensional and three-dimensional cases respectively. The final external load vector for each load case is calculated by

assembling the contributions of the corresponding Gauss points. This is illustrated at the following relations

$$F=\sum_{i=1}^{a}\sum_{j=1}^{b} Rftdet\left[J\right]w_i w_j$$

$$F=\sum_{i=1}^{a}\sum_{j=1}^{b}\sum_{k=1}^{c} Rfdet\left[J\right]w_i w_j w_k$$

where a, b and c denote the total number of Gauss points defined for each load case on axis ξ, η and ζ respectively, while w_i, w_j and w_k are the corresponding weights. Suppose that more than one load distributions act on a specific patch, the total external load vector is merely the sum of the individual vectors. This addition is enabled regardless of the number and position of Gauss points that each load case uses, since it takes place on a control point level.

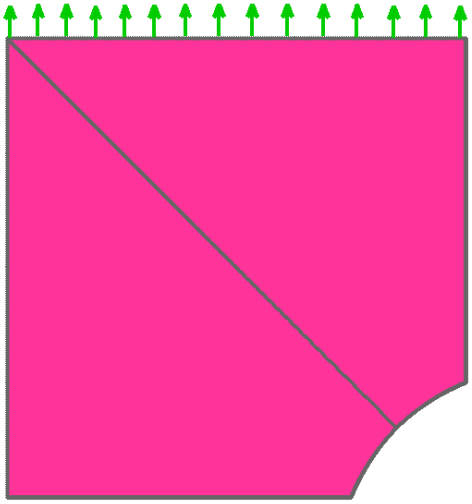


Fig. 5.1. Plate with hole: physical space with linear load

5.1.2 Collocation

The fundamental idea of the collocation method consists of the discretization of the governing partial differential equations in strong form. The boundary value problem is strongly imposed on collocation points and thus, the external load vector constitutes of the value of the load distribution at each one of them. Let us remind the equation formulated at Chapter 4:

$$F_i = \begin{cases} f(\boldsymbol{x}_i) & for\ 1 \le i \le k \\ h(\boldsymbol{x}_i) & for\ k+1 \le i \le n \end{cases}$$

where $\boldsymbol{x}_i$ represents the interior collocation points in Ω for $1 \le i \le k$, and the boundary collocation points on Γ_N for $k+1 \le i \le n$. When two dimensional problems are concerned, *f* denotes a load per unit surface and *h* a load per unit length, while in three dimensional cases *f* denotes a load per unit volume and *h* a load per unit surface. It becomes clear that body forces and surface tractions are handled separately in collocation method, since the first pertain to the governing differential equation of the problem, whereas the second to Neumann conditions.

The fact that the external load vector is formed with respect to collocation points and not to control points does not cause any confusion to the analysis process. The unknown displacements are still control point variables and this claim is ensured by the concept of the collocation scheme. In particular, the method of weighted residuals considers approximations u^* to the exact solution u of the form

$$u^* = \sum_{j=1}^{n} N_j(\boldsymbol{x}_i) c_j$$

where *c* is the vector of control variables associated to the discrete displacement. So, the unknown approximate solution u^* is expressed as a combination of the unknown control points displacements, which will be determined through the equation $Kc = F$.

From the above it can be concluded that the collocation technique requires no computations for the formulation of the external load vector. The whole procedure for each collocation point is confined to the fill of the corresponding rows of the vector with the appropriate value of the load at each direction.

Table 5.1 presents the number of operations at each collocation/quadrature point for the evaluation of the external load vector in IGA-C, IGA-G and FEA-G. The number of operations in IGA-C arises from the number of rows that each collocation point occupies depending on the dimensions of the problem.

d	IGA-C	IGA-G	FEA-G
	Evaluate external load vector	Evaluate external load vector Rf|J|w	
1	1	2(p+1)+2	2(p+1)+2
2	2	$4(p+1)^2+3$	$4(p+1)^2+3$
3	3	$6(p+1)^3+4$	$6(p+1)^3+4$

Table 5.1. Operations for the formation of external load vector per collocation/quadrature point in IGA-C/IGA-G, FEA-G

Apparently the operations required for the formation of the final linear system $Ku = F$ are orders of magnitude more expensive in the Galerkin method than in the

collocation formulation. Consequently, suppose that the computational cost is the criterion, IGA-C is a much more efficient method compared with IGA-G for the computation of displacements.

5.2 Boundary conditions

Certain degrees of freedom are fixed, in that their displacements are zero. These are called stationary and the corresponding rows and columns are erased from the stiffness matrix and the load vector. This leaves a stiffness matrix and a load vector having only free degrees of freedom, $\left[\mathrm{K_{ff}}\right]$ and $\left\{\mathrm{F_f}\right\}$ respectively. The solution of the equation is the final step in analysis:

$$\left\{\mathrm{F_f}\right\}=\left[\mathrm{K_{ff}}\right]\cdot\left\{\mathrm{D_f}\right\}\Rightarrow\left\{\mathrm{D_f}\right\}=\left[\mathrm{K_{ff}}\right]^{-1}\cdot\left\{\mathrm{F_f}\right\}$$

The (zero) displacements for the stationary degrees of freedom are added back to the result, thus creating the displacement vector $\left\{\mathrm{D}\right\}$.

5.3 Displacement

As it was previously underlined control points do not mainly belong to the model, thus their displacements do not identify with the displacements of the material points. Analysis results are called “pseudo-displacements” and they play an auxiliary role in defining the displacement field of the structure. The distribution is achieved once again via shape functions:

1D:

$$\mathrm{d}(\xi)=\sum_{\mathrm{i=1}}^{\mathrm{N}}\left\{\mathrm{R_i}\left(\xi\right)\cdot\mathrm{D_i}\right\}=\underset{(1\times\mathrm{N})}{\left\{\mathrm{R_i}\left(\xi\right)\right\}^{\mathrm{T}}}\cdot\underset{(\mathrm{N}\times1)}{\left\{\mathrm{D}\right\}}$$

2D:

$$\mathrm{d}(\xi,\eta)=\sum_{\mathrm{i=1}}^{\mathrm{N}}\left\{\mathrm{R_i}\left(\xi,\eta\right)\cdot\mathrm{D_i}\right\}=\underset{(1\times\mathrm{N})}{\left\{\mathrm{R_i}\left(\xi,\eta\right)\right\}^{\mathrm{T}}}\cdot\underset{(\mathrm{N}\times1)}{\left\{\mathrm{D}\right\}}$$

3D:

$$d(\xi,\eta,\zeta)=\sum_{i=1}^{N}\left\{R_i(\xi,\eta,\zeta)\cdot D_i\right\}=\underset{(1\times N)}{\left\{R_i(\xi,\eta,\zeta)\right\}^T}\cdot\underset{(N\times 1)}{\{D\}}$$

where N is the total number of control points and D_i is the displacement of the ith control point.

Control points that are interpolatory to the curve at (ξ_c, η_c ,ζ_c) belong to the physical model and as a result their variables correspond to actual displacements.

$$d(\xi_c,\eta_c,\zeta_c)=\sum_{i=1}^{N}\left\{R_i(\xi_c,\eta_c,\zeta_c)\cdot D_i\right\}=1\cdot D_c=D_c$$

6 Applications

6.1 Plane strain plate

The plate below is presented under plane strain conditions. The dimension across axis Z (thickness) is significantly bigger compared to the other two directions, while loads act on the plane XY. Each edge has length equal to 1m. E=100 kPa and v=0.3 are considered for the computations.

In order to emphasize the different handling of interior collocation points in Ω and boundary collocation points on Γ_N, the plate will be solved twice. In the first case all four edges will be fully supported and therefore no Neumann boundary collocation points will exist. In the second case the support will remain only across the left edge and the plate will act as a cantilever. Points lying across the other three edges will occupy the last rows of the stiffness matrix, where the strong imposition of Neumann conditions will take place.

In Fig. 6.1 the plate is divided into 9 knot spans in each direction, making a total of 81 two-dimensional knot spans (elements). Element boundaries are displayed in yellow.

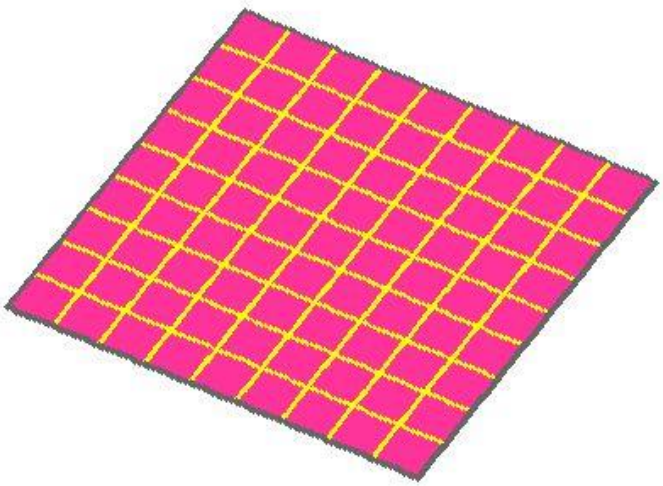


Fig. 6.1. Plane strain plate: physical space

The polynomial order is equal to 3. The same knot value vector is defined in both parametric directions: Ξ=H={0 0 0 0 1 2 3 4 5 6 7 8 9 9 9 9}. The corresponding basis functions are depicted below.

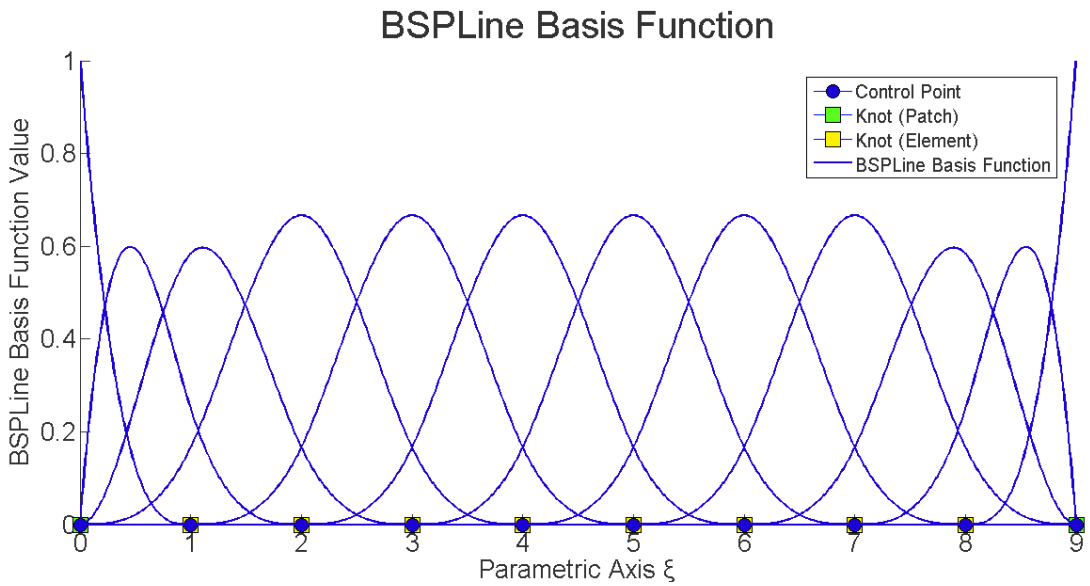
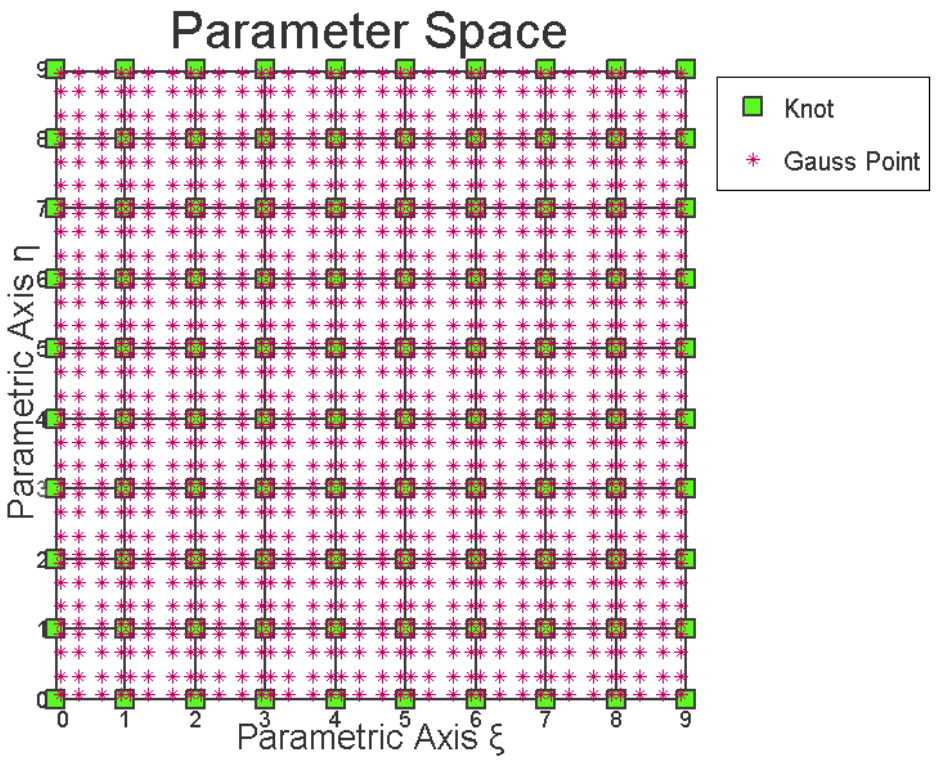


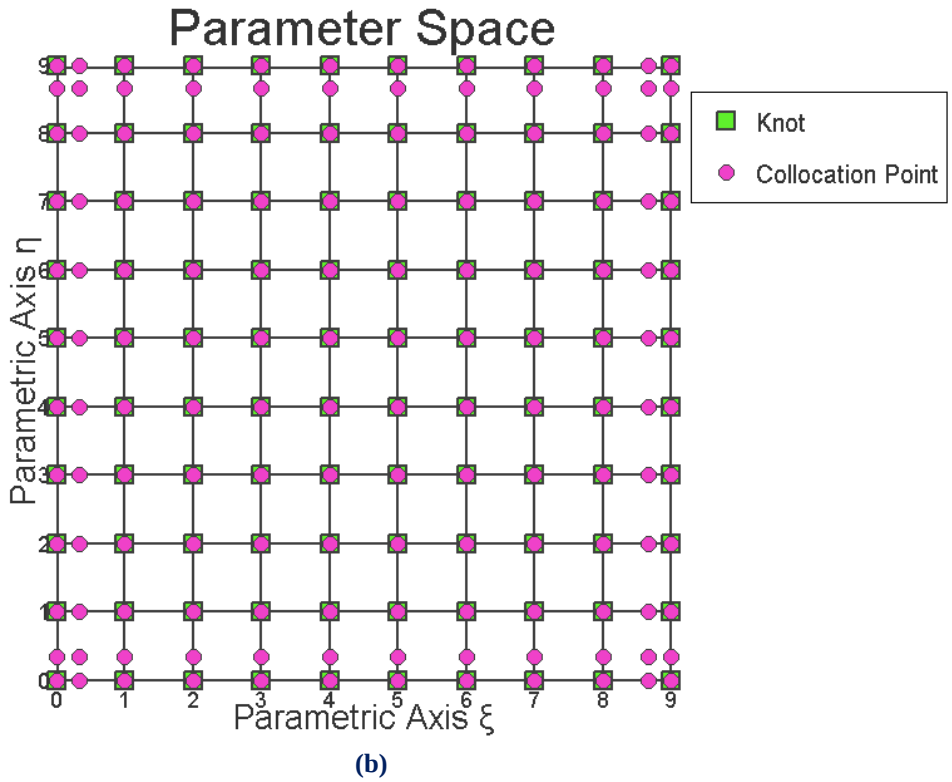
Fig. 6.2. B-spline basis functions over axes ξ, η

Stiffness matrix will be formed for both Galerkin and collocation method. Gauss quadrature suggests that $(p+1)^d = (3+1)^2 = 16$ Gauss points per element are defined. This makes a total of 81×16=1296 Gauss points over the entire domain. As far as collocation points are concerned, their number is equal to the number of control points. There are *n*=*m*=12 control points on each parametric axis, so collocation points amount to 12×12=144.

Collocation encourages one point evaluation per basis function. Dirichlet boundary conditions are satisfied strongly by the displacement approximation u^* that the method of weighted residuals introduces. Therefore, Greville points located on Γ_D do not have to be taken into account. Besides, the degrees of freedom of the corresponding control points are supported and as a result they do not participate in the final stiffness matrix either. The procedure is familiar. Rows and columns representing supported degrees of freedom are erased. The only difference at the collocation scheme is that now rows refer to collocation and not control points. However, the fact that they are both treated in the same way (they are neglected) ensures that the process yields a square matrix. This is vital for the calculation of displacements.



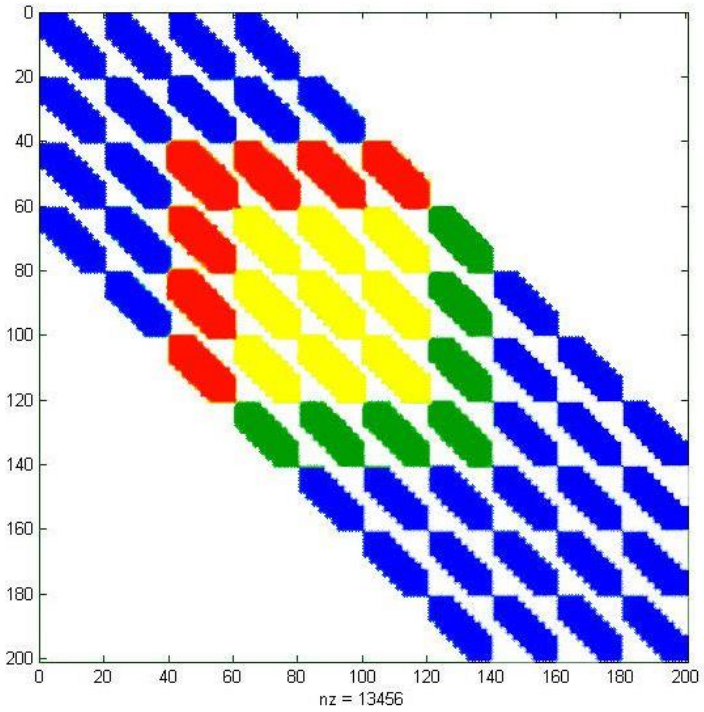
(a)



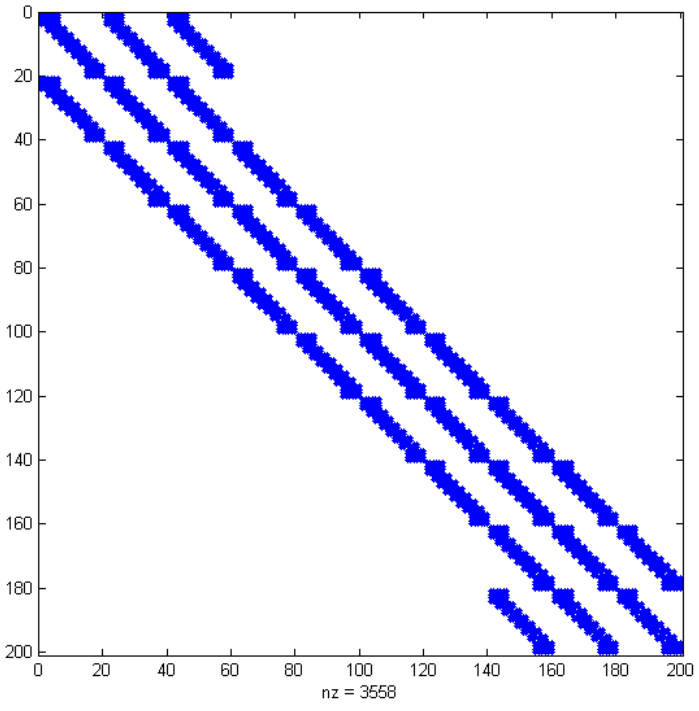
(b)

Fig. 6.3. Plane strain plate: parameter space
(a) Gauss points, **(b)** collocation points

Fig. 6.4 presents stiffness matrix based on (a) Galerkin and (b) collocation method. The non-zero elements in the first case are equal to 13456, while in the second case they are only 3558. In addition, the bandwidth is markedly reduced in collocation.



(a)



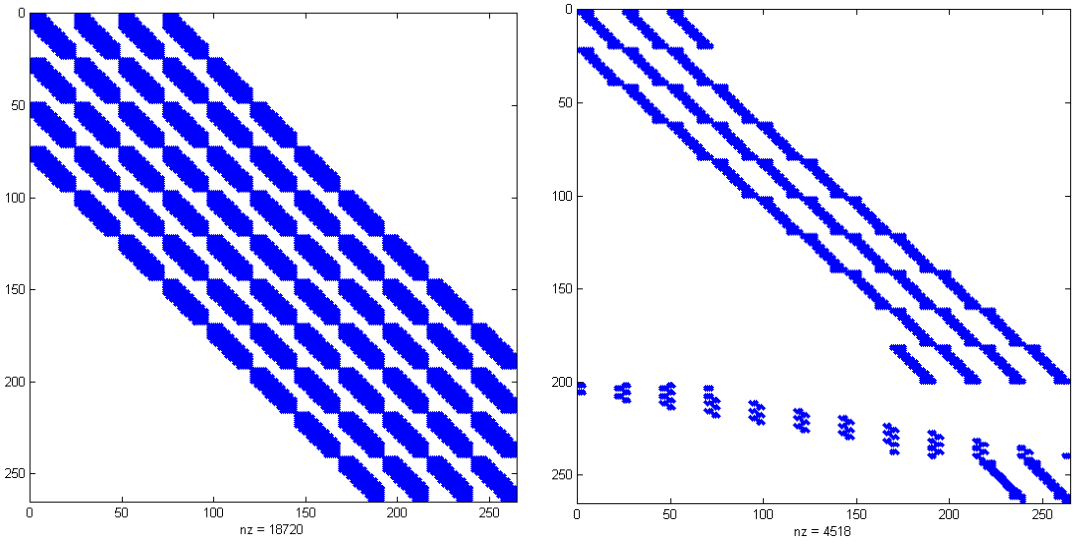
(b)

Fig. 6.4. Plane strain plate supported on four edges: stiffness matrix in **(a)** Galerkin, **(b)** collocation

The sub-matrices in color in Fig. 6.4a stand for consecutive knot spans across ξ. Their overlapping is due to the fact that they activate same degrees of freedom. In the example in question each element is affected by $(p+1)^d=(3+1)^2=16$ control points. Two successive elements involve the same one-dimensional knot spans across one parametric direction and they just differ by one across the other. This means that only $(p+1)^1=4$ different control points correspond to the second element, while the rest of them (12) are the same with the previous one. Stiffness matrix is formed by associating degrees of freedom representing rows with those degrees of freedom representing columns that they are involved with over the same elements. Consequently, successive elements occupy same positions in the system matrix and the phenomenon of overlapping appears.

This phenomenon is eliminated in collocation. In Fig. 6.4b the first twenty rows correspond to the first Greville point across ξ, the following twenty to the second point across ξ and so on. Respectively, the first two rows correlate with the first point across η, the following two correlate with the second point across η and so on. The two rows related to every collocation point are explained by the two-dimensional problem. It is obvious that the operator *L* is formed for each point separately and it contains all entries for the corresponding rows of the system matrix. There is no interaction between points and thus no overlapping.

Suppose now that the plate is supported on the left edge and that it is subjected to volume load, $f_x^v=5\ \mathrm{kN/m^2}$, $f_y^v=5\ \mathrm{kN/m^2}$. The domain discretization remains the same and the polynomial order is here also equal to 3.

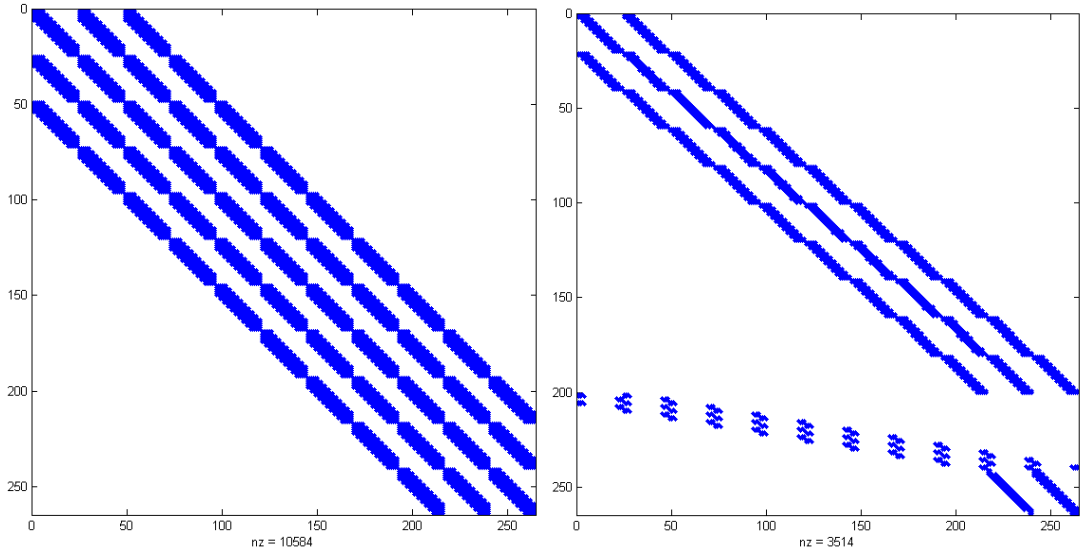


(a) **(b)**

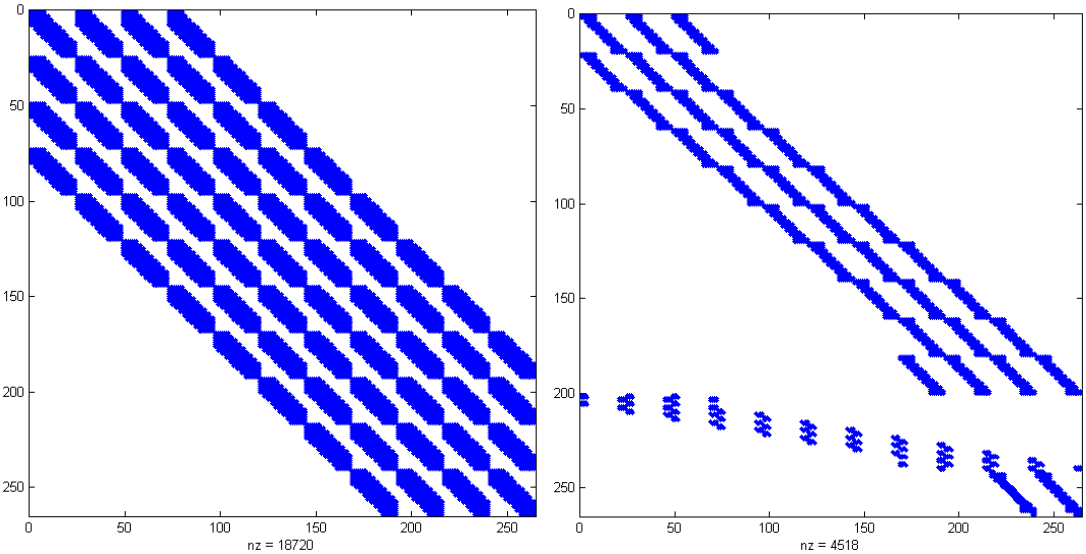
Fig. 6.5. Plane strain plate supported on the left edge: stiffness matrix in **(a)** Galerkin, **(b)** collocation

In Fig. 6.5b the first 200 rows correspond to interior collocation points in Ω and the rest of them to boundary collocation points lying across bottom, upper and right edge. The strong form of the differential equation is satisfied at the first category of points, while strong imposition of Neumann conditions takes place at the second. Despite the "not normal" form of collocation matrix, its bandwidth is still reduced compared to Galerkin matrix (p instead of $2p+1$). Additionally, the non-zero elements in Galerkin are more than four times the non-zero entries in collocation.

In order to better illustrate the differences in the matrix of the two methods, Fig. 6.6 presents the stiffness matrices for polynomial degree ranging from 2 to 5, for a standard number of degrees of freedom (288 in total, 264 free).



(a)



(b)

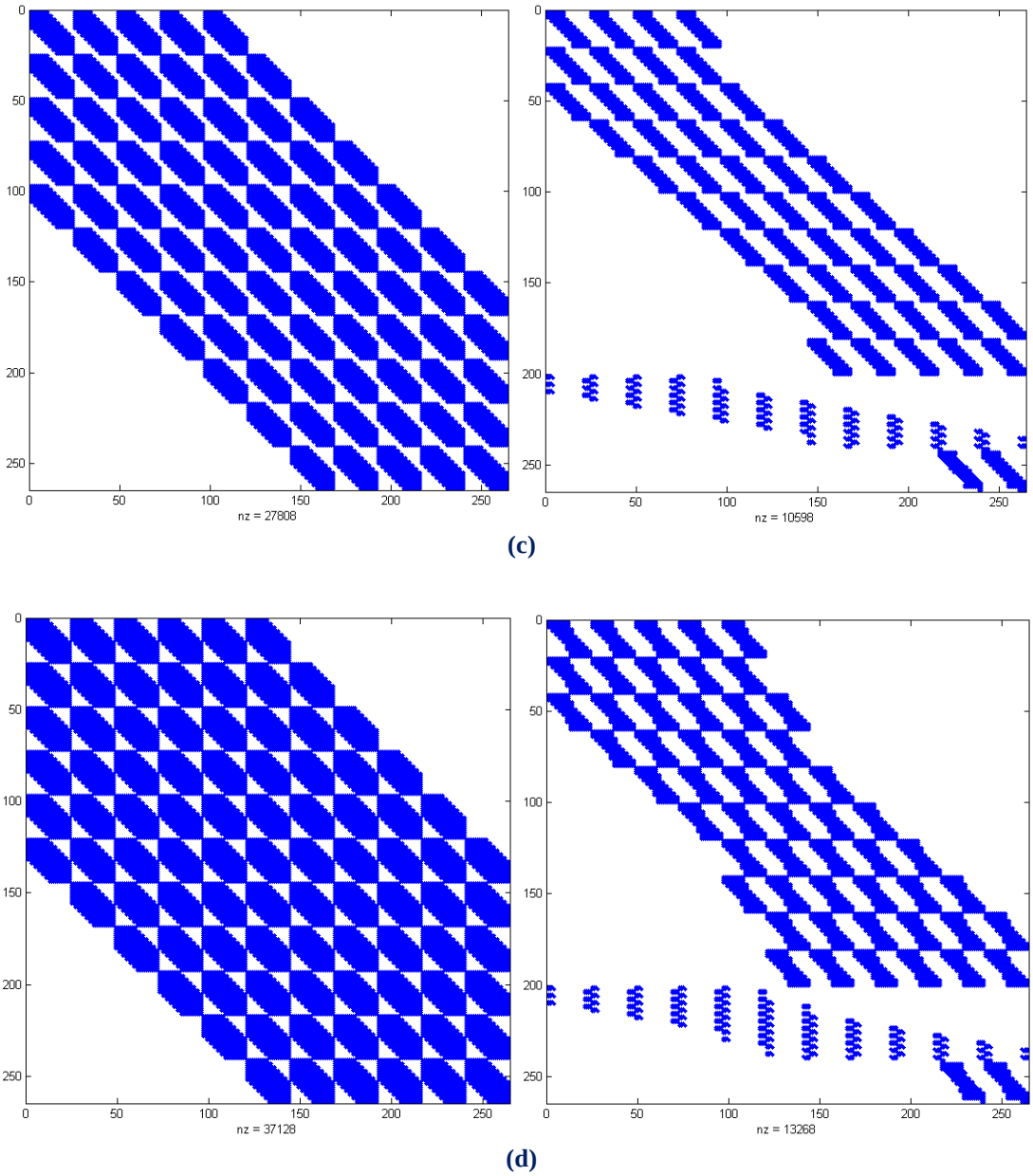


Fig. 6.6. Stiffness matrix in Galerkin (left column) and collocation (right column)
(a) p=2, **(b)** p=3, **(c)** p=4, **(d)** p=5

In collocation it is obvious that basis functions of even polynomial degrees lead to the same bandwidth as those of the next higher odd *p*. The matrix of *p*=2 has the same bandwidth with the matrix of *p*=3 (*p*+1 and *p* respectively). This outcome applies for *p* equal to 4 and 5 as well.

Matrices representing both methods tend to be significantly denser as the discretization order increases. Greater degree implies that the domain of influence of each function is extended to a larger area. In case of Galerkin this is reflected in more intense interaction between elements and therefore more intense overlapping. In terms

of collocation this is associated with the fact that the number of basis functions having non-zero value over each collocation point, or else the number of non-zero entries at each row, is increased with the order elevation. In both methods the increasingly denser structure of the system matrix is completely justified.

Next, displacements are calculated for meshes of 4x4, 6x6, 9x9, 12x12 and 14x14 control points, using quadratic B-spline basis functions. The calculations will then be repeated for cubic basis. For each case the total time and the displacement error are recorded. Total time pertains to the operations required for the formation of the stiffness matrix and external load vector until displacements are defined. The code used for the analysis purposes was written in MatLab.

It should be underlined that no program optimization has been considered, therefore the results presented below are not representative of real times. In case of IGA-G in particular there are solvers, compatible to parallelization techniques, which can drastically accelerate computations occupying in the same time less storage space [5]. Apart from that, the factor of symmetry in the Galerkin method has been neglected. It is known that the utilization of the symmetry properties can reduce the operations required at each quadrature point, since in this case only the upper triangular part of the local stiffness matrix has to be formed. It follows from the above that the Galerkin times presented in this section could probably be decreased by orders of magnitude.

However, isogeometric collocation also looks really promising for parallel implementations. Undoubtedly, optimization algorithms adapted to the needs of the method will improve its efficiency and even better computing times will be reached. Consequently, the "simplified" code used in this thesis serves the fair treatment of the two formulations and the acquired times are indicative of the attitude of each one of them. Of course both methods have to be programmed in the optimum way in order to end up with a reliable conclusion.

For the present example, no closed-form solution is available, thus a reference solution is computed for a mesh with degree 3 in both parametric directions and 14x14 control points solved with the Galerkin method. The normalized displacement error in the L^2 norm is evaluated as follows

$$e_{L^2}^{\mathrm{Re}f} = \frac{\left\| \boldsymbol{u} - \boldsymbol{u}^{ref} \right\|_{L^2}}{\left\| \boldsymbol{u}^{ref} \right\|_{L^2}}$$

where $\boldsymbol{u}^{\boldsymbol{ref}}$ is the reference displacement solution.

It should be mentioned that the "collocation points" column of Table 6.1 refers to those Greville points that are involved in the formation of the stiffness matrix. This

means that points located on Γ_D are not taken into account. As far as Gauss points are concerned, their calculation follows the rule of $(p+1)^d$ points per element.

Mesh	ndof	Gauss/collocation points		Total time (s)		L^2 norm	
		Galerkin	Collocation	Galerkin	Collocation	Galerkin	Collocation
4x4	32	36	12	0,36	0,15	0,037	0,050
6x6	72	144	30	1,00	0,19	0,010	0,020
9x9	162	441	72	4,76	0,42	0,003	0,006
12x12	288	900	132	16,42	0,94	0,002	0,005
14x14	392	1296	182	57,74	2,24	-	0,003

Table 6.1. Plane strain plate with discretization order equal to 2: computing times and displacement errors for IGA-G, IGA-C

It can certainly be claimed that IGA-C is a faster method than IGA-G. For sparse discretization and low order, the time the two schemes consume is comparable. In more complicated problems, however, where denser meshes or larger p is used, IGA-C increasingly dominates over IGA-G, since the cost of the first is invariant to the number of unknowns per node. The number of Gauss points and consequently the computational cost of IGA-G is increased in a significantly fast rate with the knot insertion.

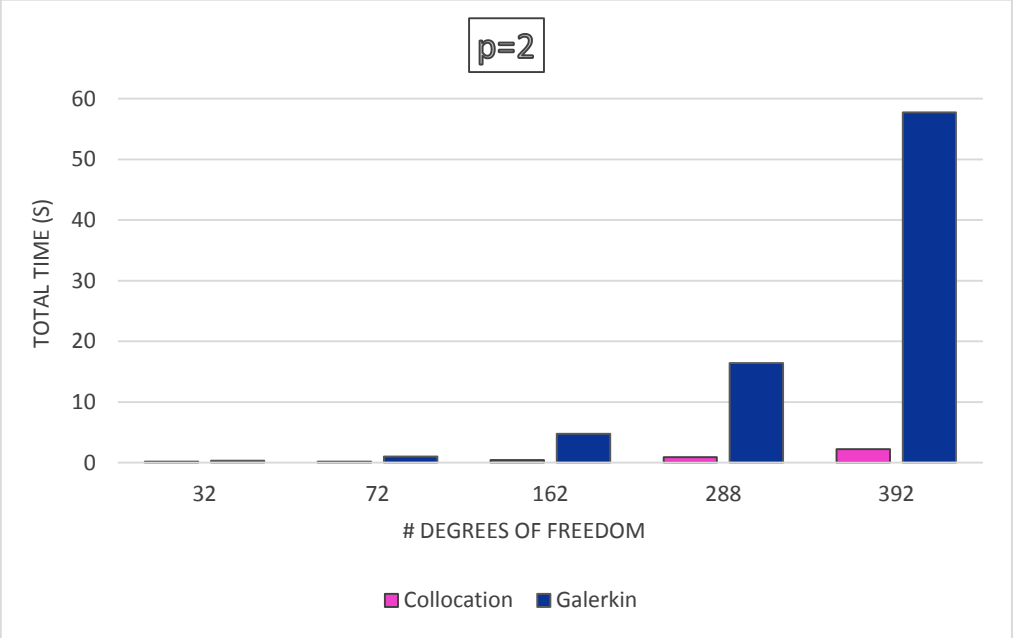


Fig. 6.7. Computing times for the formation and solution of the system KD=F

With respect to the convergence properties, collocation approaches the performance of IGA-G for discretization of degree 2. As the mesh becomes denser the corresponding errors tend to be equal. These results are hopeful enough, still the method should also be tested in more complex and large-scale problems. Besides it will later be shown that in three-dimensional applications, as well as in two dimensional cases of p=3 the divergence between the errors of the two methods is significantly increased.

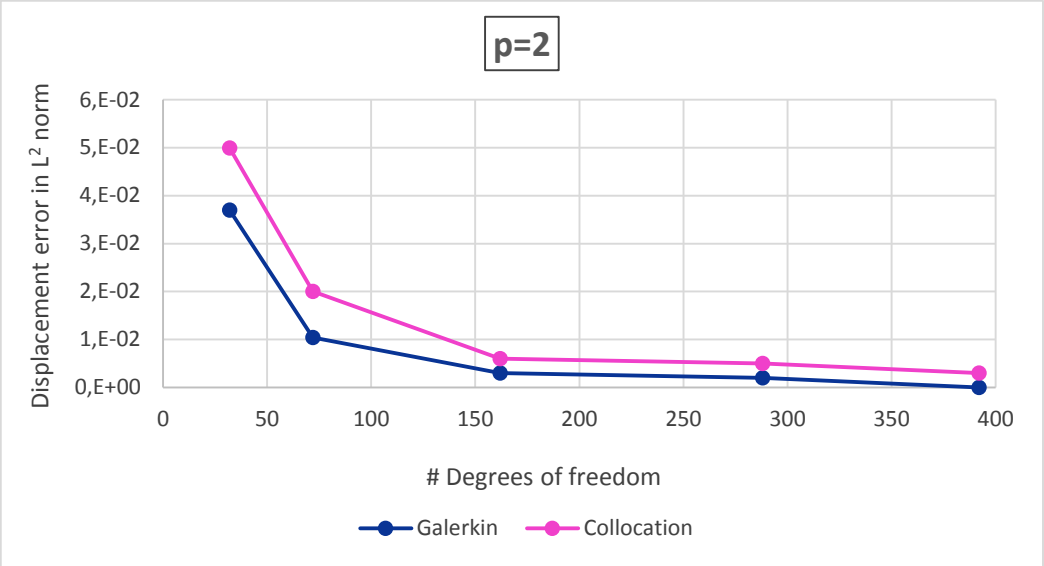
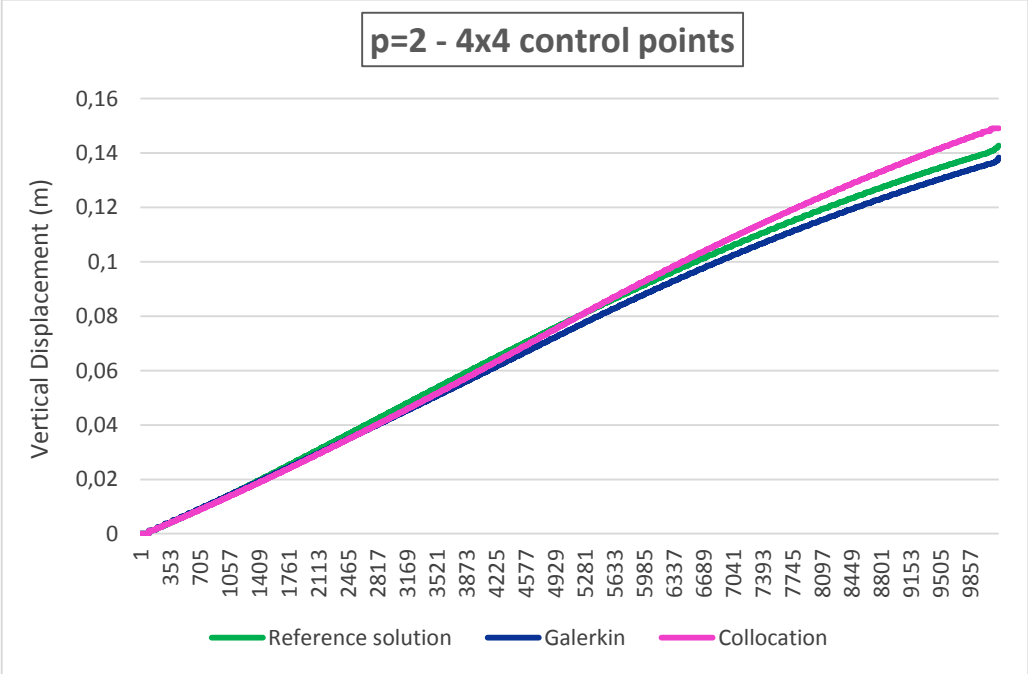
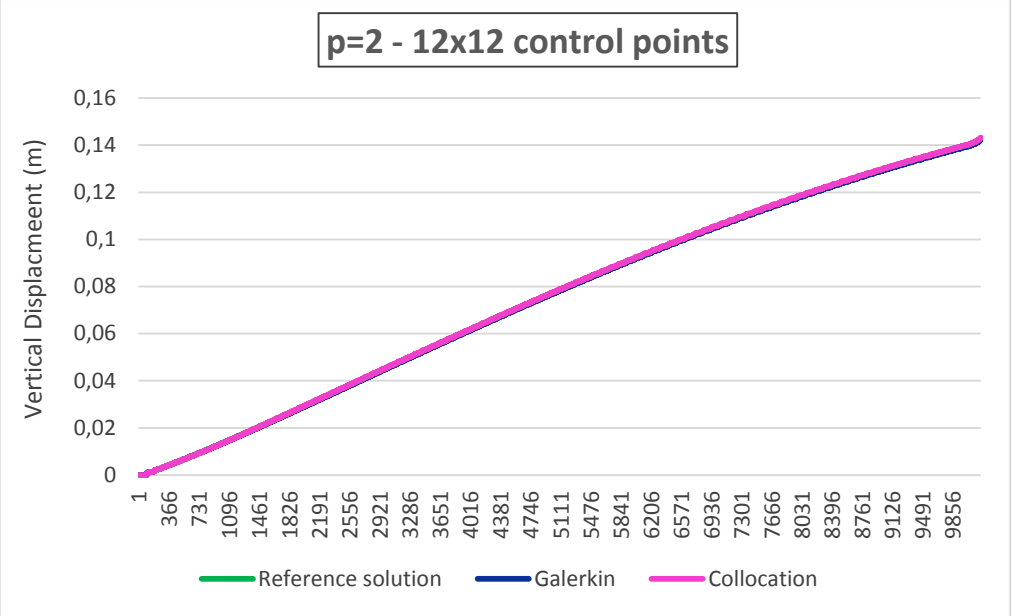


Fig. 6.8. Displacement error in L^2 norm vs. degrees of freedom



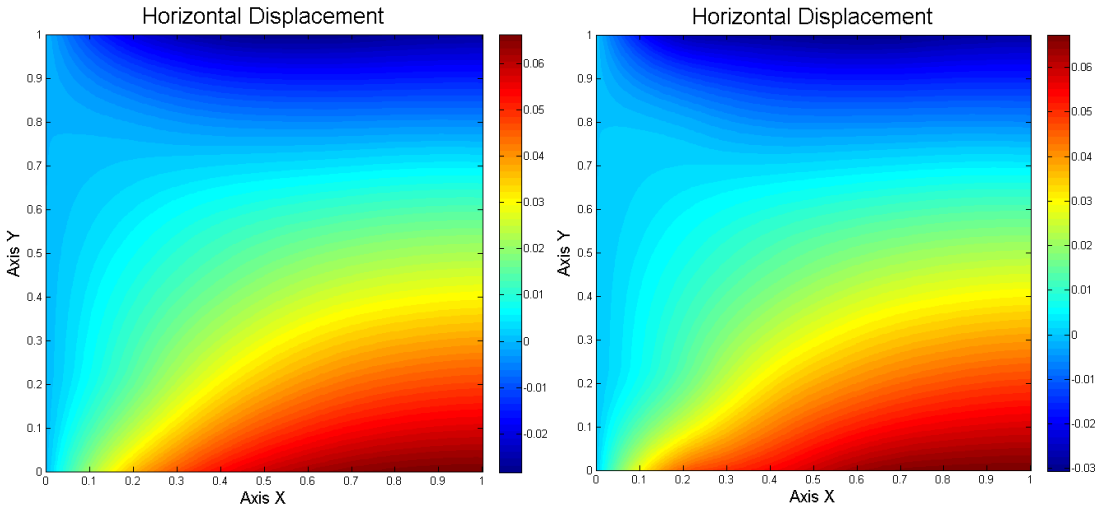
(a)



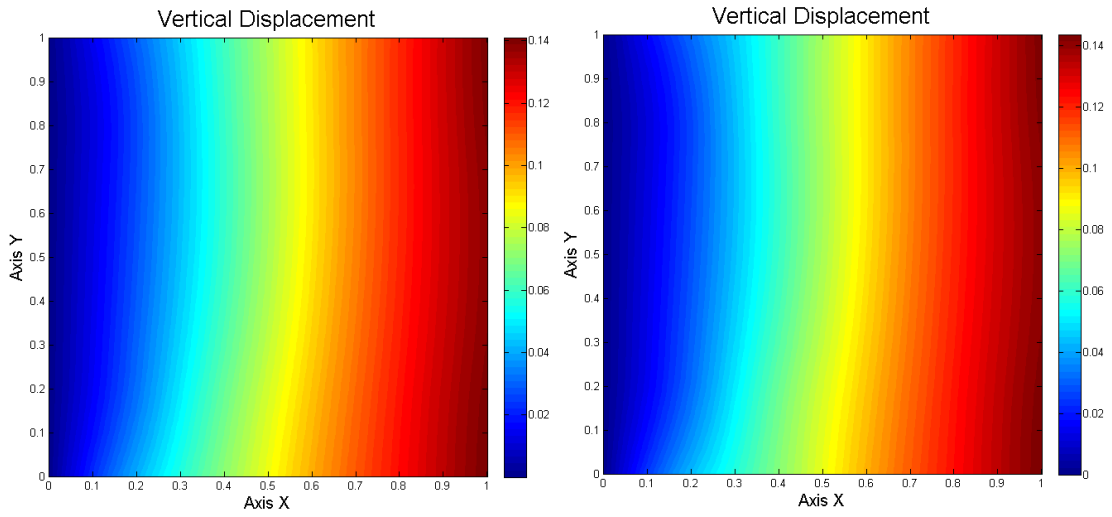
(b)

Fig. 6.9. Plane strain plate: vertical displacement
(a) p=2, 4x4 control points, **(b)** p=2, 12x12 control points

Displacement contours corresponding to the mesh of 6x6 control points are presented below for both horizontal and vertical direction. The level of convergence that each mesh density reaches ranges between small numbers and the difference cannot be clearly caught by this type of graphs. As a result, the contours of the specific mesh can be thought as representative of the graphs of all the five occasions, which are omitted for the sake of brevity.



(a)



(b)

Fig. 6.10. Displacement contour in Galerkin (left column) and collocation (right column)
(a) horizontal displacement, **(b)** vertical displacement
p=2, 6x6 control points

The above examples are now repeated for polynomial order equal to 3. The number of degrees of freedom remains the same in each case.

Mesh	ndof	Gauss/collocation points		Total time (s)		L^2 norm	
		Galerkin	Collocation	Galerkin	Collocation	Galerkin	Collocation
4x4	32	16	12	0,3	0,15	0,026	0,391
6x6	72	144	30	1,00	0,19	0,007	0,023
9x9	162	576	72	6,28	0,42	0,002	0,017
12x12	288	1296	132	24,00	0,94	$4{,}11 \times 10^{-4}$	0,012
14x14	392	1936	182	138,00	2,24	-	0,010

Table 6.2. Plane strain plate with discretization order equal to 3:
computing times and displacement errors for IGA-G, IGA-C

The analysis procedure is more time-consuming in case of cubic functions as far as Galerkin is concerned. This follows from the increased number of quadrature points compared to the corresponding mesh of quadratic splines. Indeed, Gauss points per element are now equal to $(p+1)^d = (3+1)^2 = 16$, while in the previous calculations they were equal to $(p+1)^d = (2+1)^2 = 9$. Of course, it should also be kept in mind that for the same number of degrees of freedom p=3 produces less elements than p=2. Otherwise, an even more pronounced difference between consecutive orders would be noted. On the other hand, collocation's cost keeps the same values, since it is not

affected by *p*. The number of Greville points is equal to the number of control points, which remains unchanged in problems sharing the same degrees of freedom.

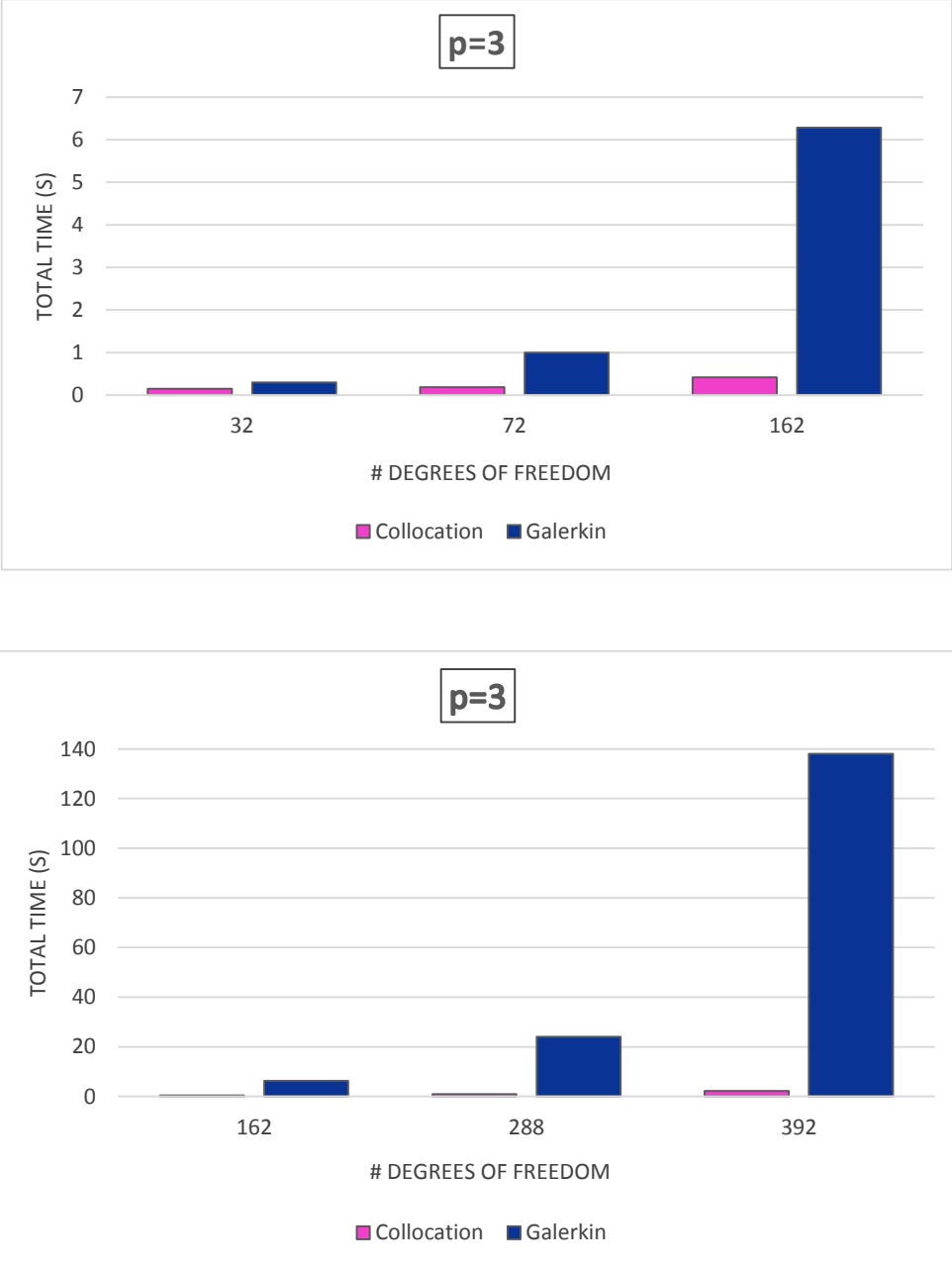


Fig. 6.11. Computing times for the formation and solution of the system KD=F

As it was expected, Galerkin succeeds better accuracy as the polynomial order increases. In case of collocation though, higher displacement errors are observed. In general, IGA-C is said to be more efficient for even than odd degrees, so this could probably be a reason for this misbehavior.

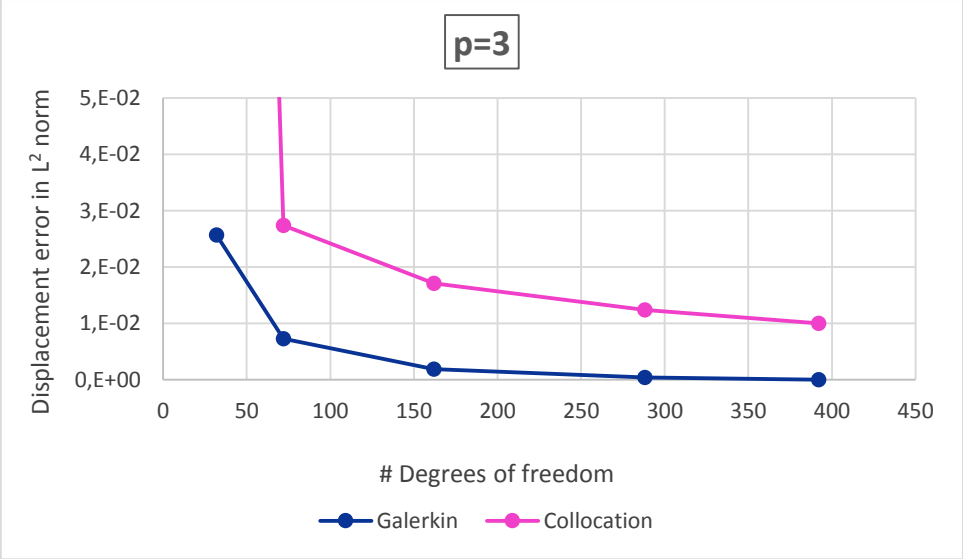
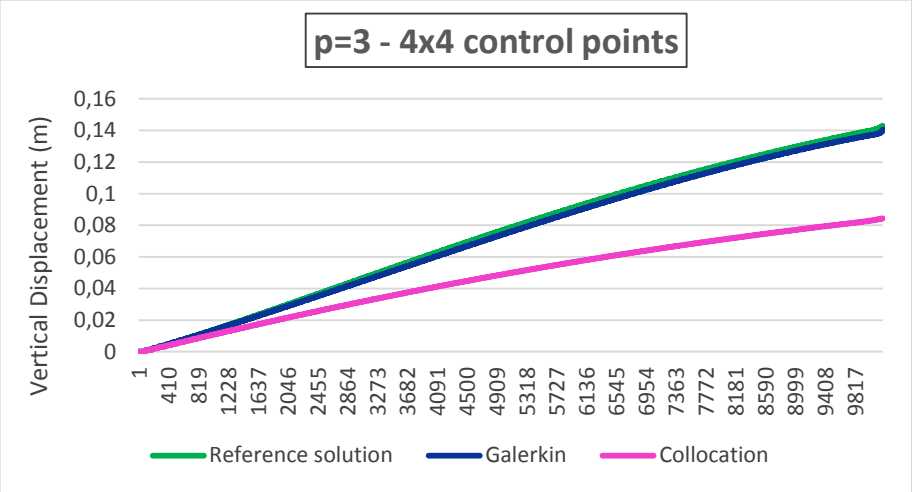
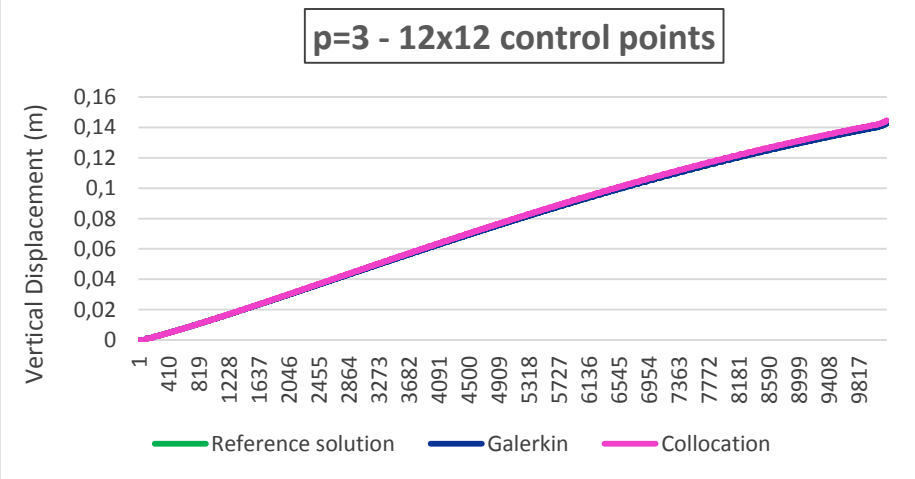


Fig. 6.12. Displacement error in L^2 norm vs. degrees of freedom



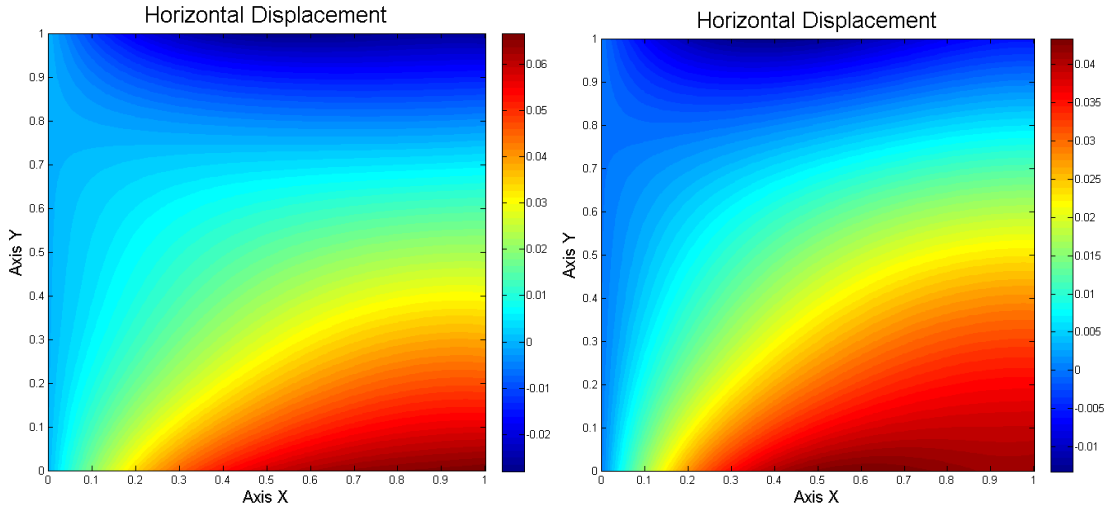
(a)



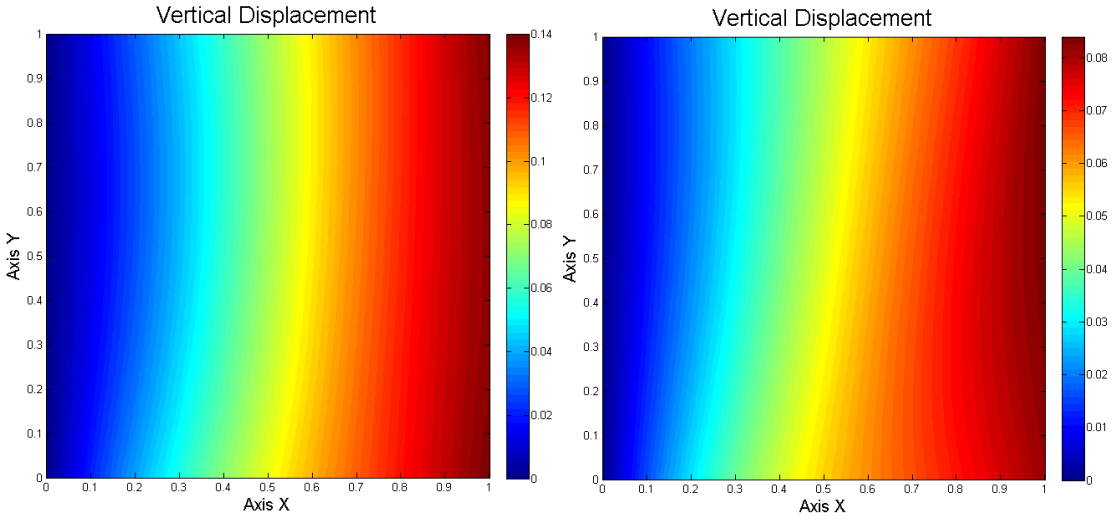
(b)

Fig. 6.13. Plane strain plate: vertical displacement
(a) p=3, 4x4 control points, **(b)** p=3, 12x12 control points

Displacement contours are displayed in the following figures in order to illustrate the response of IGA-G and IGA-C in both sparse and dense meshes.

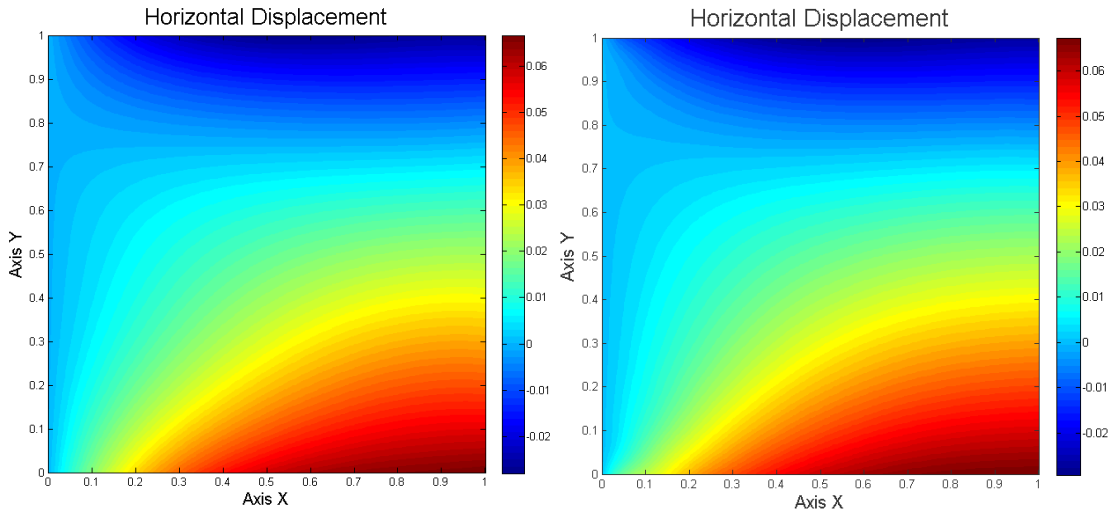


(a)

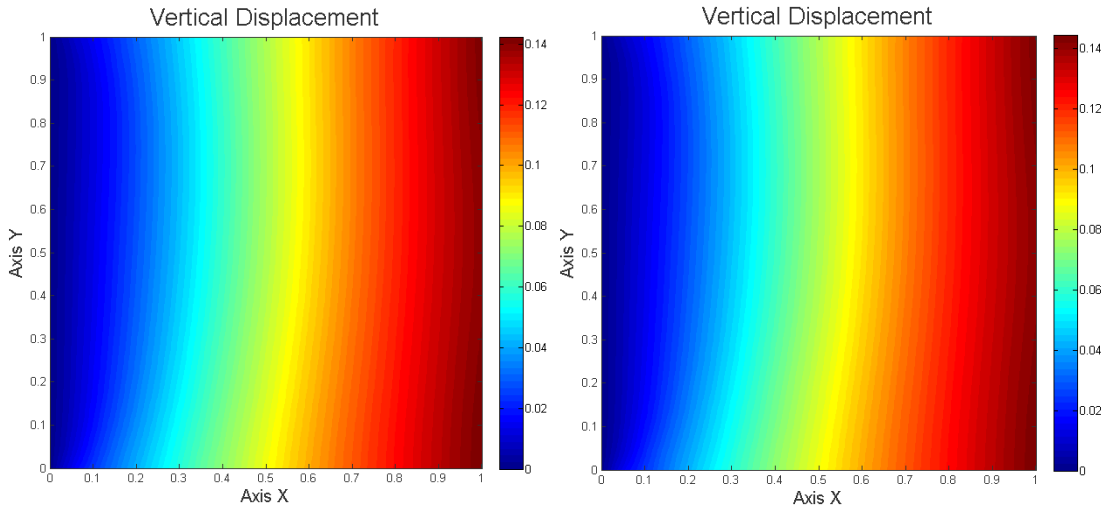


(b)

Fig. 6.14. Displacement contour in Galerkin (left column) and collocation (right column)
(a) horizontal displacement, **(b)** vertical displacement
p=3, 4x4 control points



(a)

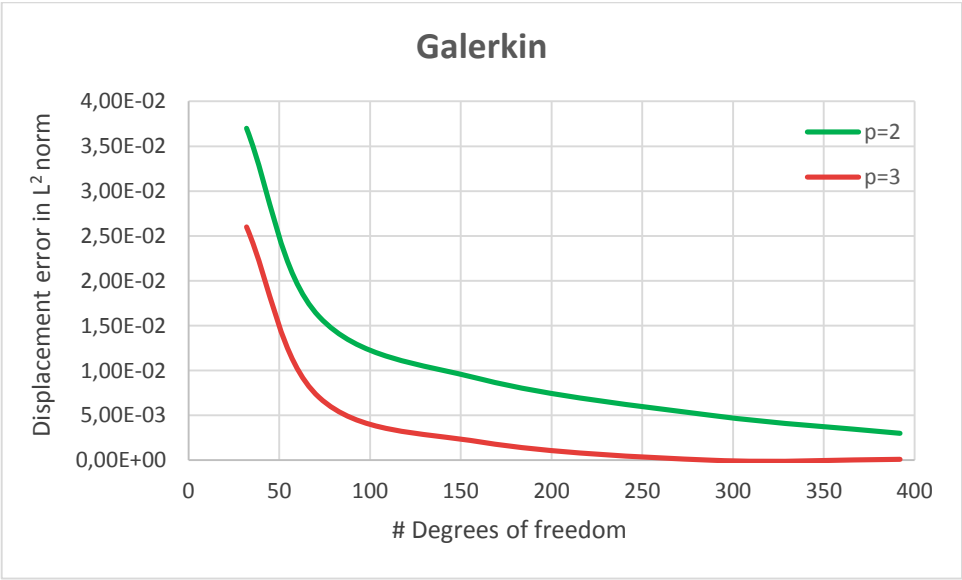


(b)

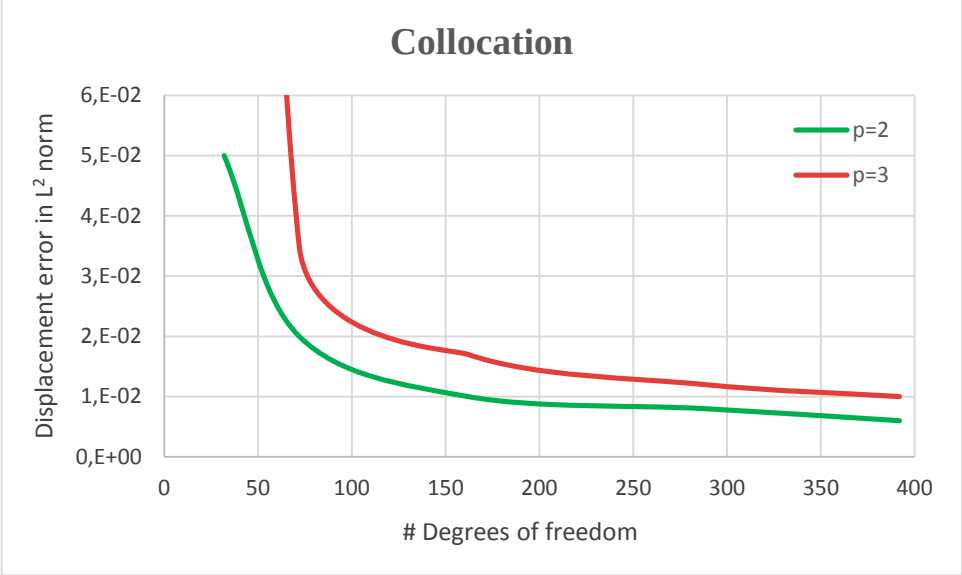
Fig. 6.15 Displacement contour in Galerkin (left column) and collocation (right column)
(a) horizontal displacement, **(b)** vertical displacement
p=3, 12x12 control points

As the degrees of freedom increase, Galerkin becomes much more expensive than collocation. However, Galerkin can reach a certain level of accuracy even in sparse meshes, where few degrees of freedom are defined. Collocation on the other hand, will need to use refinement techniques so as to approach Galerkin results. Even under these circumstances though, it is still not clear which method outperforms as far as computational cost is concerned.

A thorough comparison between the two methods would entail the estimation of the time required by each one of them in order to achieve a specified level of accuracy regardless of the polynomial degree and the number of degrees of freedom.



(a)



(b)

Fig. 6.16. Displacement error in L^2 norm vs. degrees of freedom for various p
(a) Galerkin, **(b)** collocation

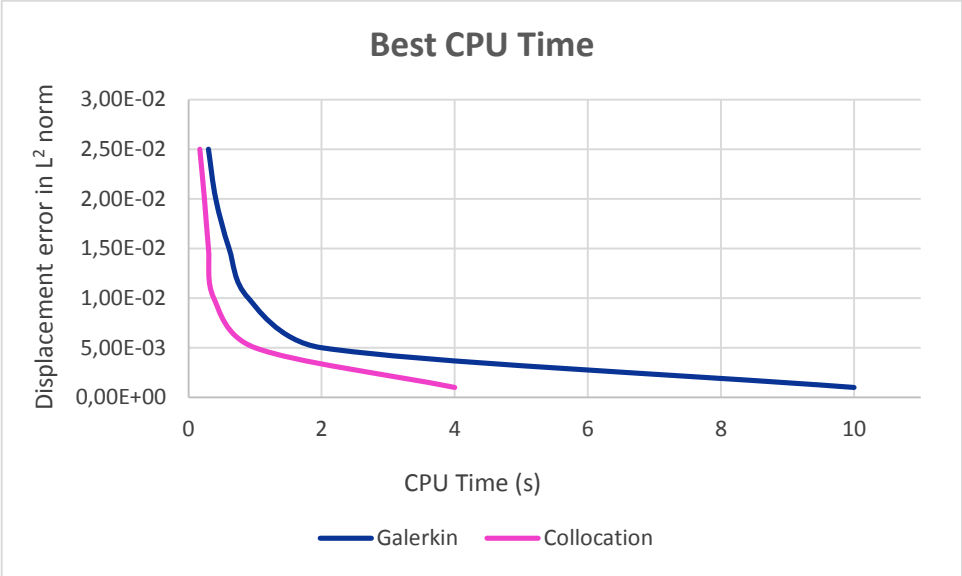


Fig. 6.17. Displacement error in L^2 norm vs. time

Fig. 6.17 demonstrates a superiority of IGA-C compared to IGA-G, considering that it offers the best accuracy to computing time ratios. Nevertheless, more reliable results will arise provided that the above charts are enriched with more polynomial degrees and number of degrees of freedom. Besides for small p the cost between the methods is expected to be comparable under optimization techniques. It is the increase of $(p+1)^d$ (number of gauss points per element) that makes the assembly of the stiffness matrix a computationally demanding task.

6.2 Plane strain cantilever

The cantilever presented in Fig. 6.18 obeys to plane strain conditions. Its length is equal to 0.9 m and its width is equal to 0.3 m. It is subjected to a surface load of 40 kN/m on its upper edge. E=8000 kPa and v=0.3.

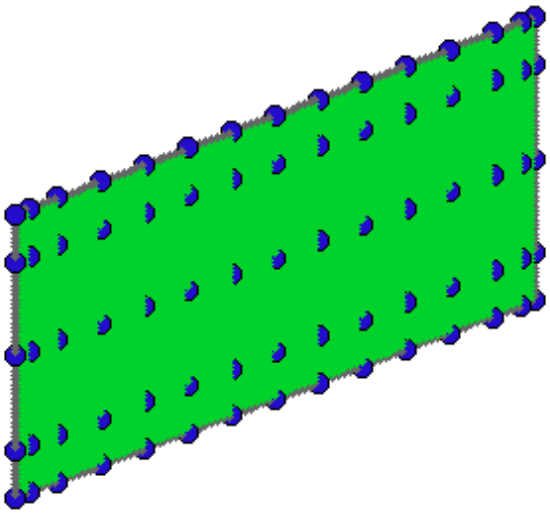
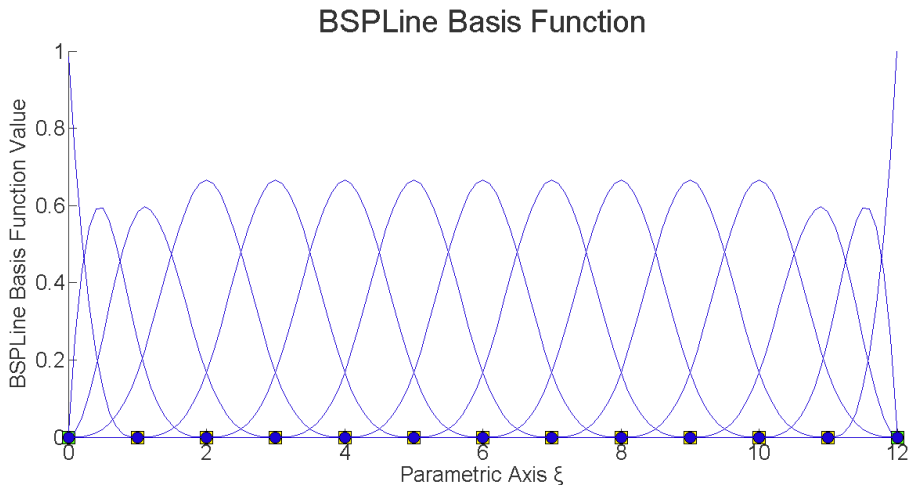
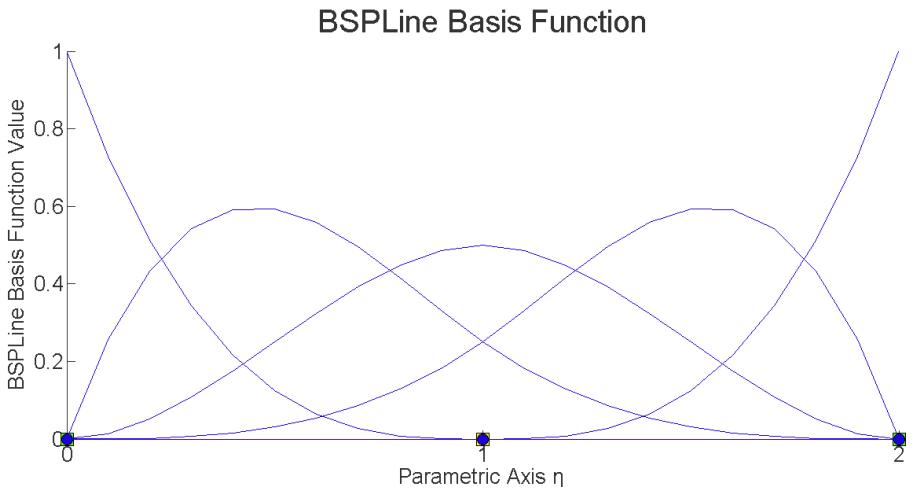


Fig. 6.18. Plane strain cantilever: physical space

The control points defining the geometry are shown in blue. There are 15 control points across axis ξ and 5 control points across axis η. The polynomial order is equal to 3. This makes a total of 384 Gauss points and 75 collocation points.

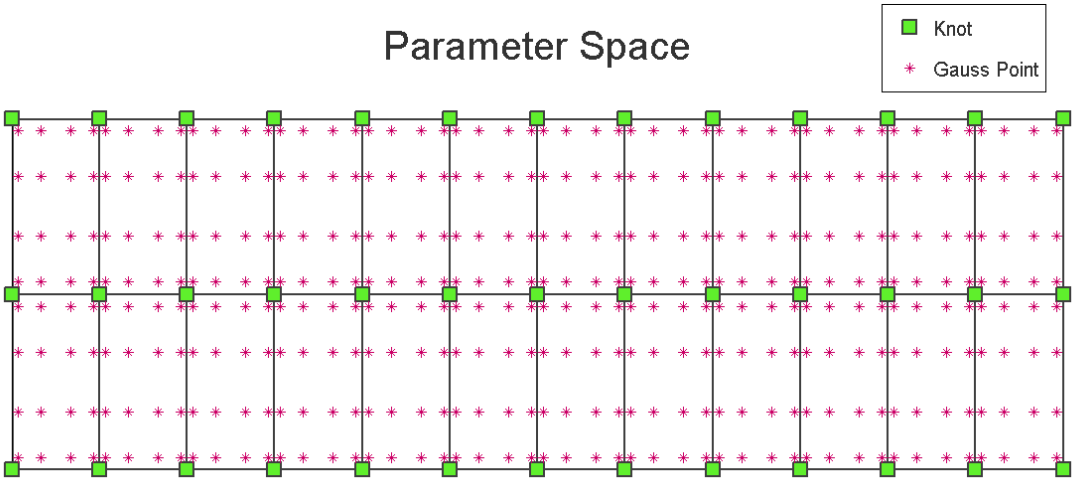


(a)

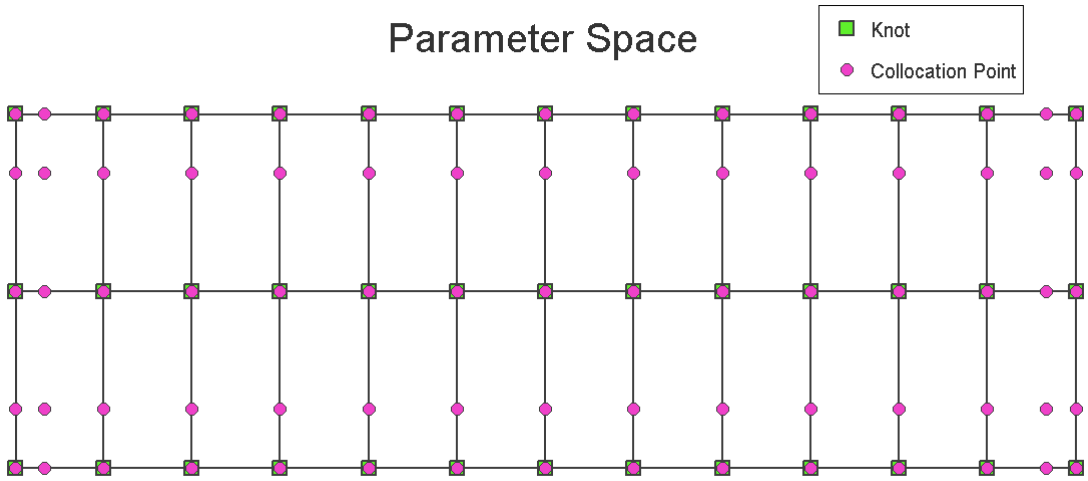


(b)

Fig. 6.19. B-spline basis functions over **(a)** axis ξ, **(b)** axis η



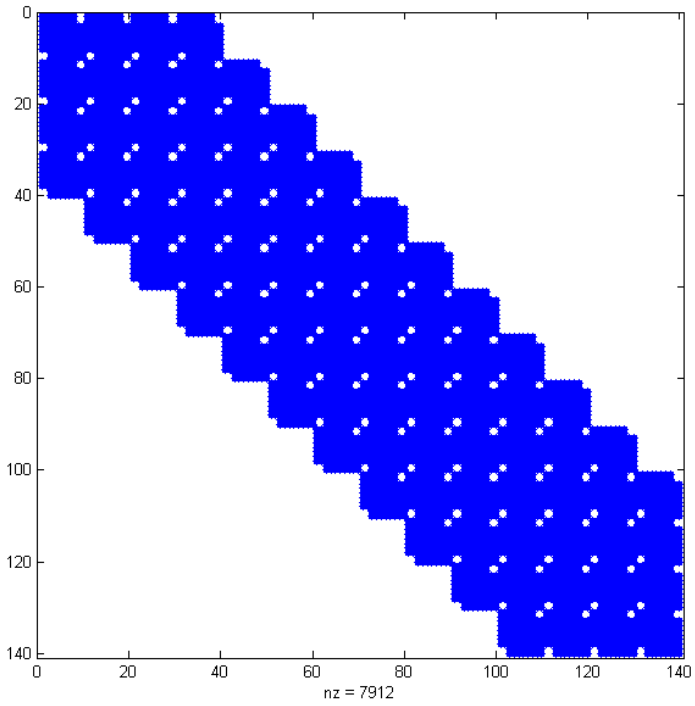
(a)



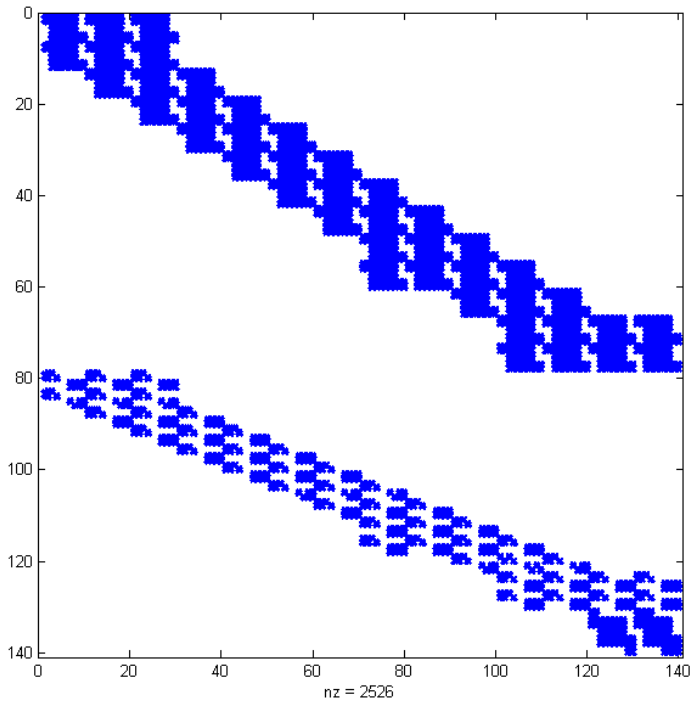
(b)

Fig. 6.20. Plane strain cantilever: parameter space
(a) Gauss points, **(b)** collocation points

The stiffness matrix deriving from the Galerkin formulation has 7912 non-zero elements, while the corresponding number for the collocation scheme is just 2526. The relative displacement error between the two methods is defined in the L^2 norm and it is equal to 7×10^{-3}.

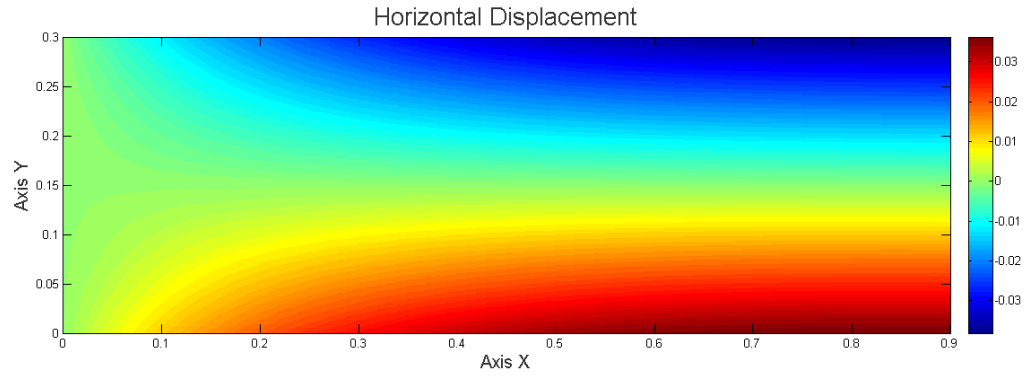


(a)

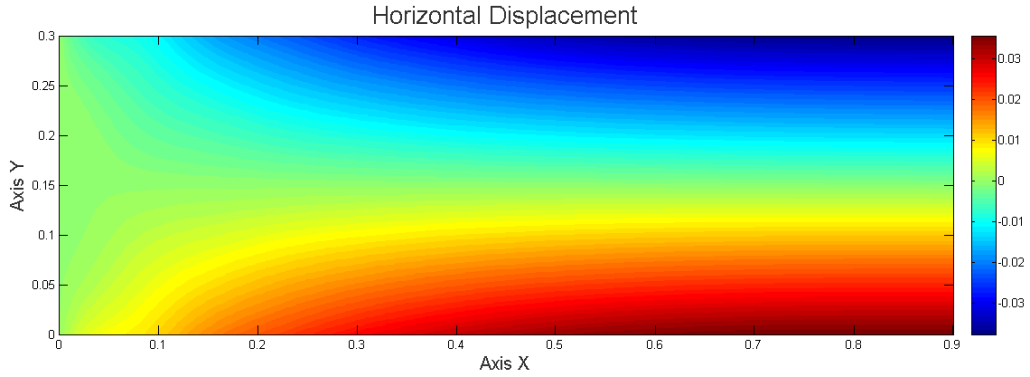


(b)

Fig. 6.21. Plane strain cantilever:
stiffness matrix in **(a)** Galerkin, **(b)** collocation

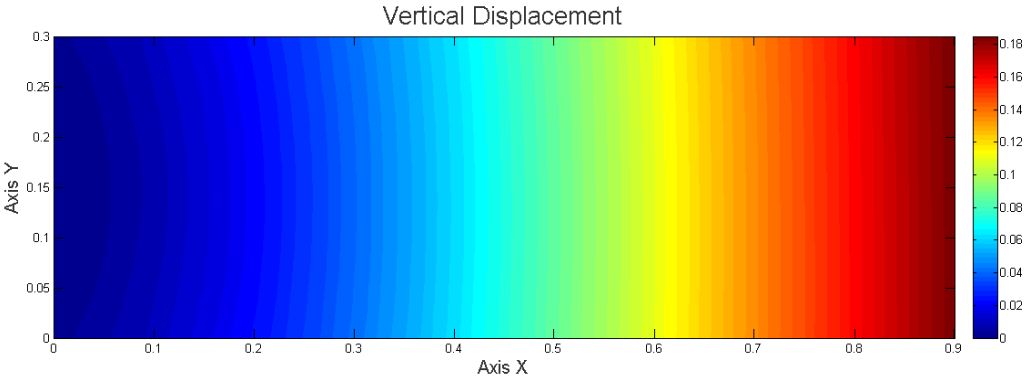


(a)

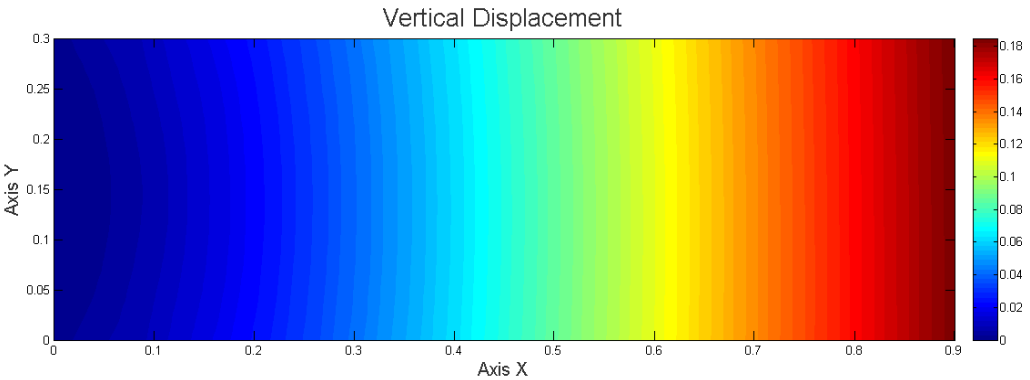


(b)

Fig. 6.22. Horizontal displacement contour in **(a)** Galerkin, **(b)** collocation



(a)



(b)

Fig. 6.23. Vertical displacement contour in **(a)** Galerkin, **(b)** collocation

6.3 Cube

The cube of Fig. 6.24 has sides of length 1m and it rests on the left side. It is subjected to a volume load with values $f_x^v = 30\ \text{kN/m}^3$, $f_y^v = 30\ \text{kN/m}^3$ and $f_z^v = 30\ \text{kN/m}^3$. E=1000 kPa and v=0.3.

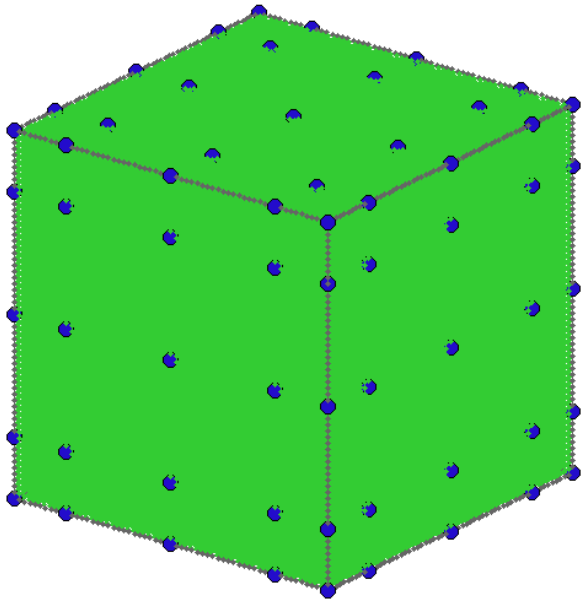


Fig. 6.24. Cube: physical space and the corresponding control points

3 knot spans are defined at each parametric direction. This number in combination with quadratic basis functions yields 5 control points per axis. The minimum number of required Gauss points rises to 729, while there are 125 collocation points. Those lying on the left side (25 Greville points) are used for the strong imposition of Dirichlet boundary conditions, consequently only 100 contribute to the formation of the stiffness matrix. The problem is specified by 375 degrees of freedom.

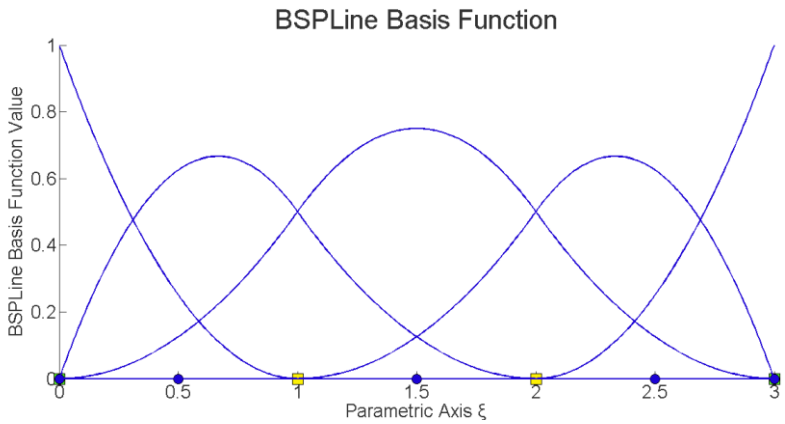
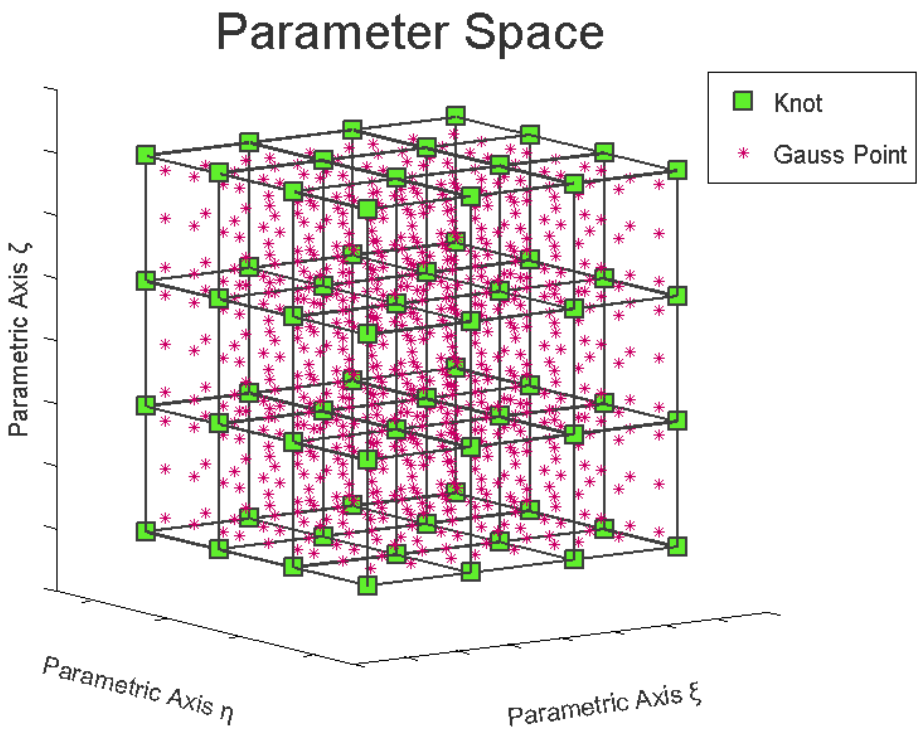
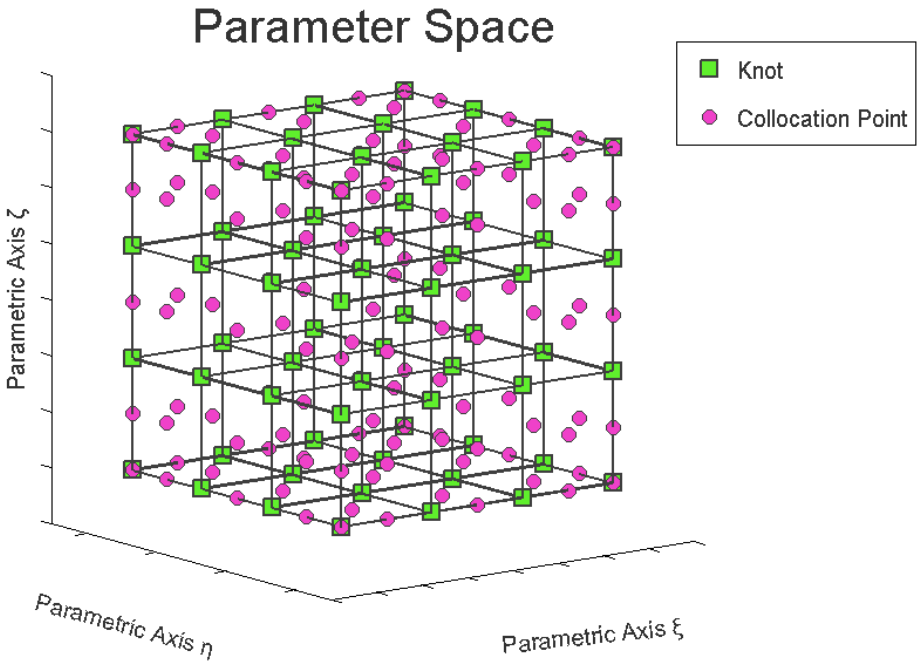


Fig. 6.25. B-spline basis functions over axes ξ, η and ζ
Ξ=H=Z={0 0 0 1 2 3 3 3}



(a)



(b)

Fig. 6.26. Cube: parameter space
(a) Gauss points, **(b)** collocation points

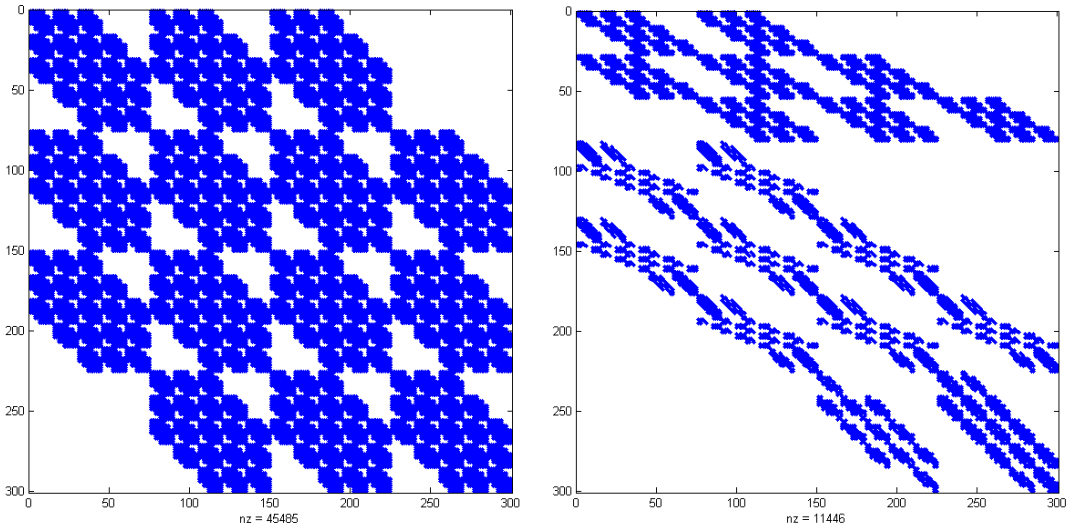
In this example no reference solution is defined. Based on the high accuracy that IGA-G offers, the relative error of the two methods is calculated in the L_2 norm as an indicator for the approximation power and convergence properties of IGA-C.

$$e_{L^2} = \frac{\left\| \boldsymbol{u}^c - \boldsymbol{u}^G \right\|_{L^2}}{\left\| \boldsymbol{u}^G \right\|_{L^2}}$$

where $\boldsymbol{u}^c$ is the collocation displacement solution and $\boldsymbol{u}^G$ is the Galerkin displacement solution. All the information relevant to the analysis procedure is outlined in the following table.

Method	Galerkin	Collocation
Degrees of freedom	375	
Gauss/collocation points	729	100
Total time (s)	18,64	1,40
Norm	0,041	

Stiffness matrix has the same features as in two-dimensional cases. In Galerkin the increased density reflects the overlapping of basis functions, which is connected to the C^1 continuity across element boundaries. In collocation, each Greville point holds three consecutive rows corresponding to u_x, u_y and u_z displacements. Interior and boundary points are treated in a different way.



(a) **(b)**

Fig. 6.27. Cube:
stiffness matrix in **(a)** Galerkin, **(b)** collocation

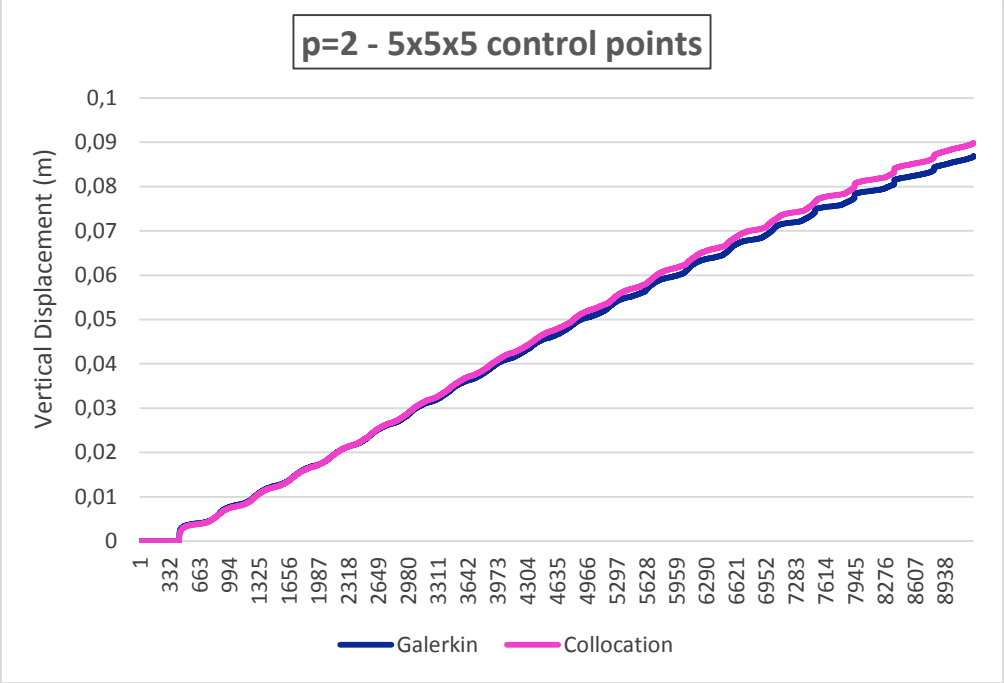


Fig. 6.28. Cube: vertical displacement

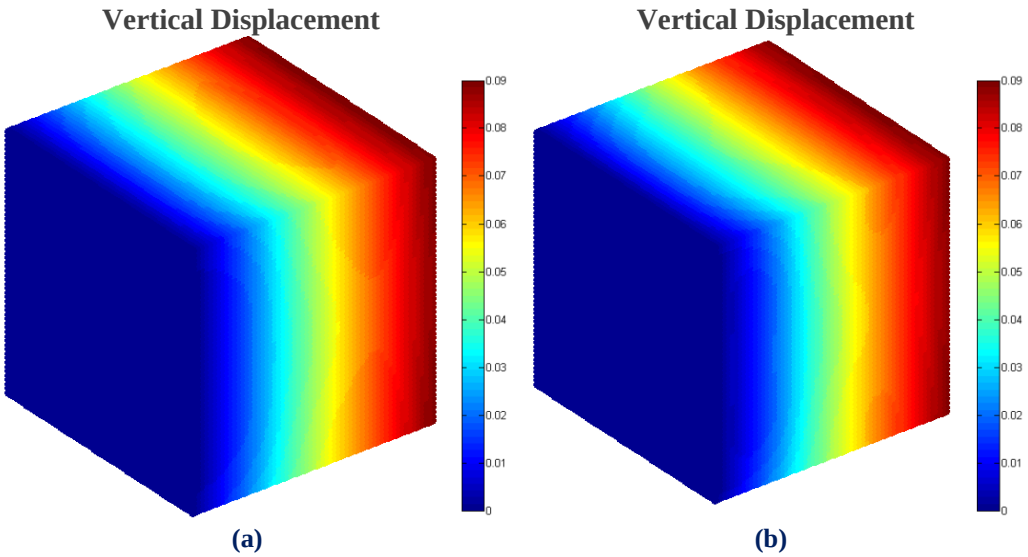


Fig. 6.29. Vertical displacement contour in **(a)** Galerkin, **(b)** collocation

6.4 Cantilever 3D

The cantilever has length 2,5m, while its width and height are equal to 1m. It is subjected to a volume load at z direction, $f_z^v = 10\ \text{kN/m}^3$. E=5000 kPa and v=0.3.

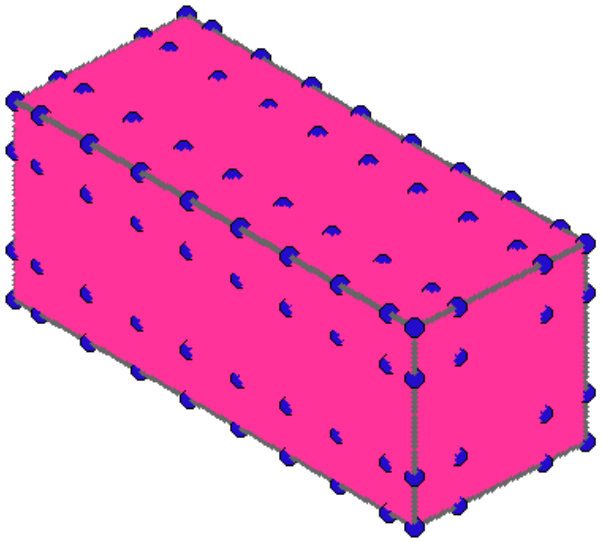
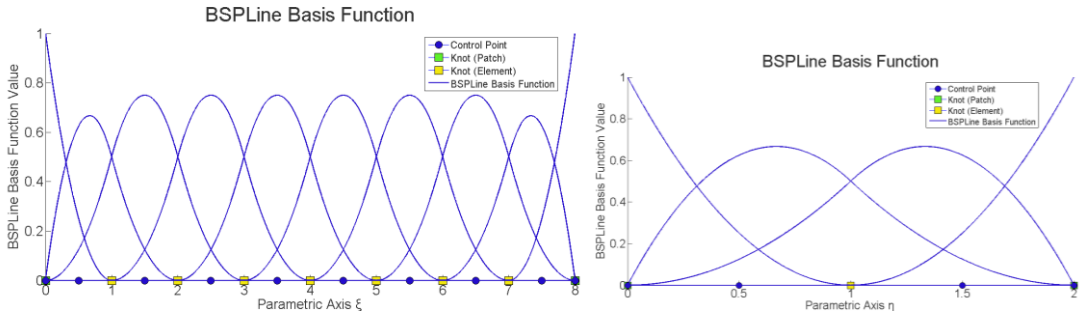


Figure 6.30. Cantilever 3D: physical space and the corresponding control points

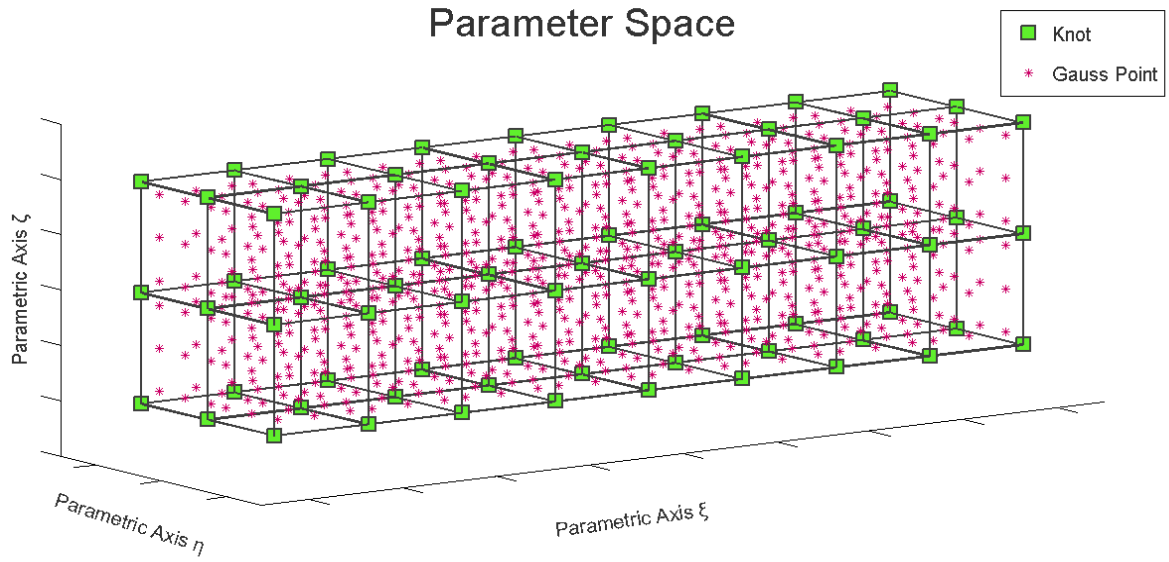
Quadratic B-splines are used here as well. 10 control points are defined across axis ξ and 4 across the remaining two axes. This makes a total of 480 degrees of freedom, 864 Gauss points and 160 collocation points (16 of them belong to the supported side). The displacement error is here deteriorated compared the previous three-dimensional example. This is probably attributed to the fact that in the current problem the ratio between boundary collocation points to the total number of collocation points is increased. As it will be mentioned in the next chapter the strong imposition of Neumann conditions may lead to inaccuracies in some occasions.

Method	Galerkin	Collocation
Degrees of freedom	480	
Gauss/collocation points	864	144
Total time (s)	30	2
Norm	0,1146	

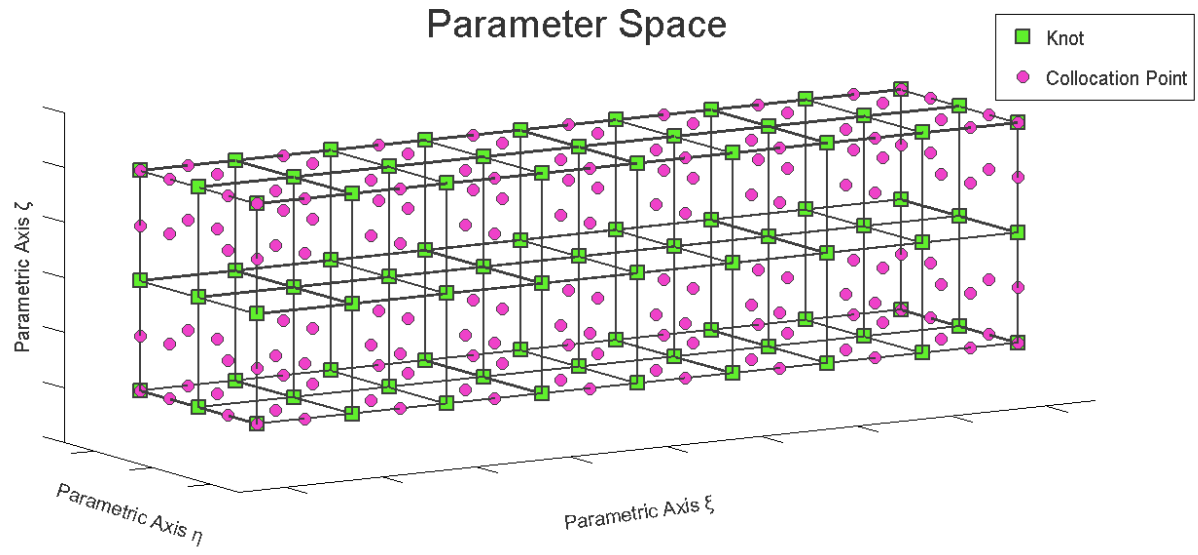


(a) **(b)**

Figure 6.31. B-spline basis functions over axes **(a)** ξ, **(b)** η, ζ
Ξ={0 0 0 1 2 3 4 5 6 7 8 8 8}
H=Z={0 0 0 1 2 2 2}

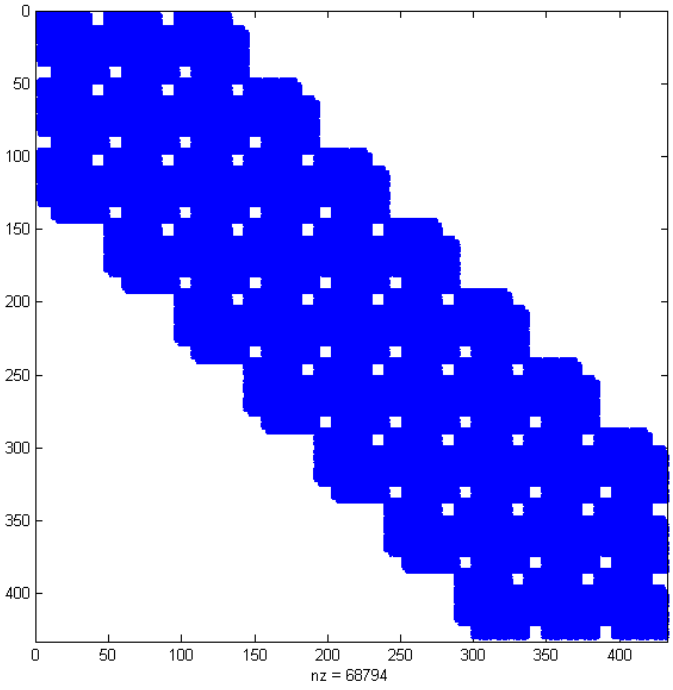


(a)

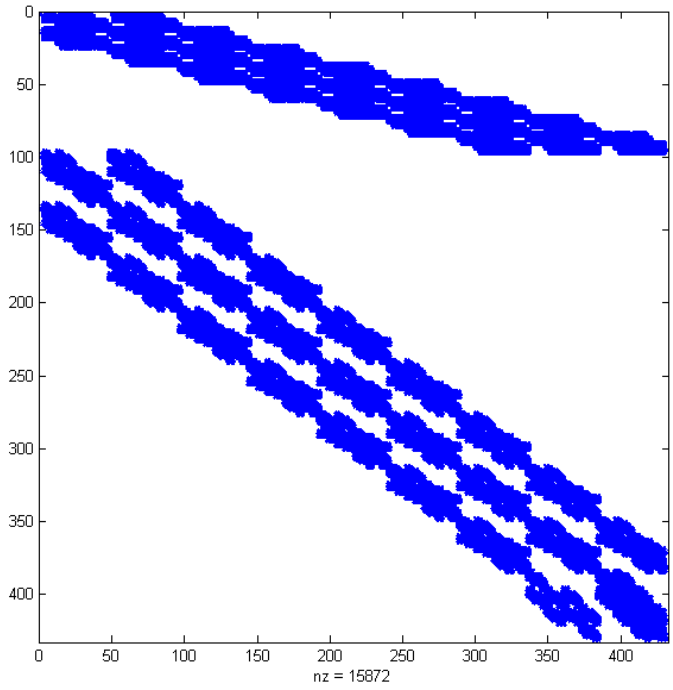


(b)

Figure 6.32. Cantilever 3D: parameter space
(a) Gauss points, **(b)** collocation points



(a)



(b)

Fig. 6.33. Cantilever 3D:
stiffness matrix in **(a)** Galerkin, **(b)** collocation

Observe that Galerkin matrix have almost 4,5 times more non-zero elements than the matrix deriving from collocation. The form of the second matrix becomes more and more unfamiliar as the number of boundary collocation points increases. Yet it completely complies with the principles of the methodology in question. The entries of

the first rows of the global array are the non-zero values of NURBS shape functions second derivatives over each interior collocation point with respect to the pattern of L operator. Rows from 100 to 432 refer to Greville points located on Γ_N. This time the first derivatives of NURBS shape functions are inserted at suitable positions of the stiffness matrix under the layout of B^h operator.

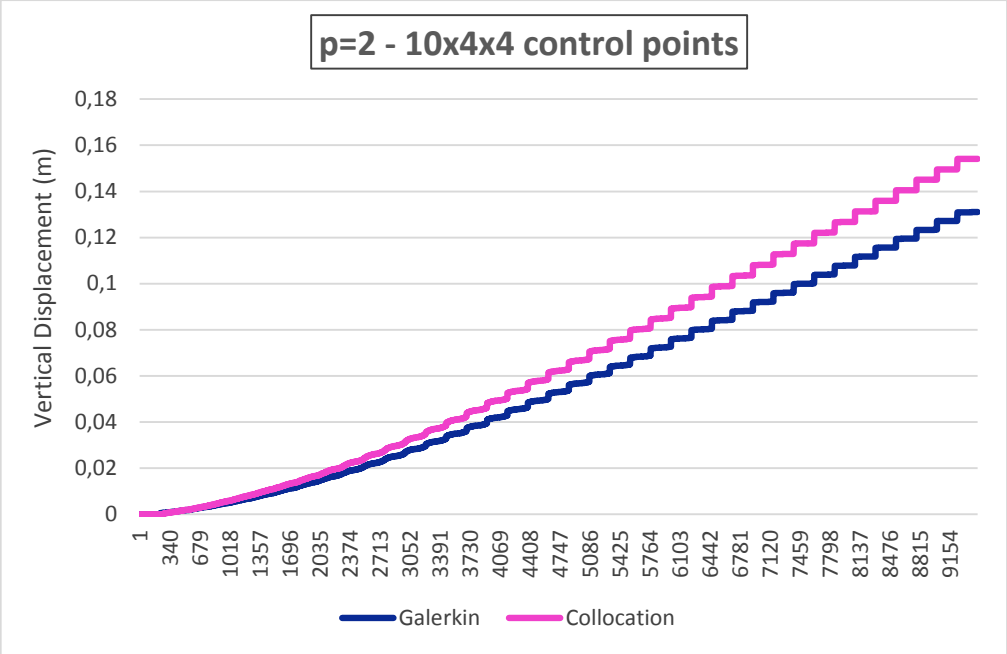
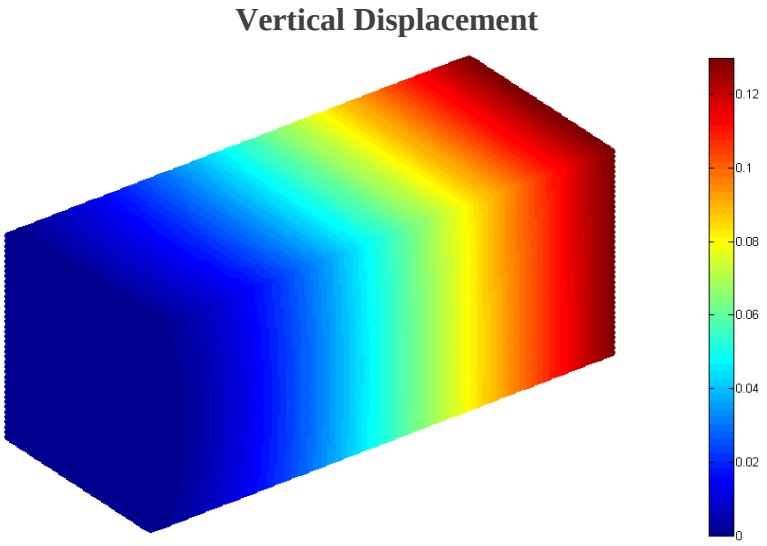
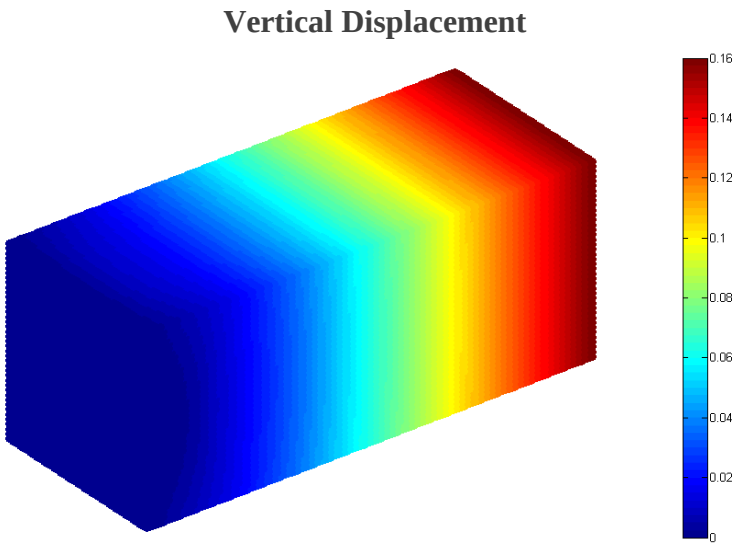


Fig. 6.34. Cantilever 3D: vertical displacement



(a)



(b)

Fig. 6.35. Vertical displacement contour in **(a)** Galerkin, **(b)** collocation

7 Drawbacks of Collocation Scheme

7.1 Accuracy

As it can be concluded from the previous chapters, collocation formulation in isogeometric analysis outperforms IGA-G and FEA-G from the aspect of computational cost. The strong imposition of the boundary value problem eliminates integrals and the need for numerical quadrature and simplifies the procedure for the stiffness matrix formation. However, for the efficiency of one method to be decided, the crucial parameter of accuracy should also be taken into consideration. Engineering problems in particular, that the aforementioned numerical schemes are meant to serve, exhibit very small tolerance in accuracy issues and they are highly sensitive in instabilities.

Unfortunately, the theoretical background of the isogeometric collocation method has only been established for the one-dimensional case. In [8] it is demonstrated that if the collocation points are chosen suitably, the collocation method converges with optimal rate, while proofs of stability and error estimates are also included. For 2 and 3-dimensional cases some individual attempts have only been realized, which cannot compose an abstract mathematical framework able to support a thorough numerical analysis of collocation methods. As a consequence, convergence results for 2D and 3D NURBS discretization, that are currently available, derive only from numerical studies.

Accuracy issues are under investigation via a 3D elasticity problem in [6]. Of course the success of a collocation method mostly depends on the choice of collocation points. Greville abscissae is adopted here too. Assuming the PDE system of linear elasticity, a set of exact smooth and rough solutions are defined over a cube $\Omega=[0,1]^3$. The corresponding smooth and rough solution, respectively, reads

$$u = v = w = \sin\left(2\pi x\right)\sin\left(2\pi y\right)\sin\left(2\pi z\right) \tag{1}$$

$$u = v = w = xyz\left(\left(x-1\right)^2 + \left(y-1\right)^2 + \left(z-1\right)^2\right)^{\frac{1}{4}} \tag{2}$$

Substitution of the above equations into Navier's equations of elasticity yields the body forces f_x, f_y and f_z for the two sets of displacement fields. In the first case, homogenous Dirichlet boundary conditions are considered over all surfaces of the cube. For the rough case, Neumann boundary conditions need to partially be imposed at the surfaces x=1, y=1 and z=1. They can be derived by inserting Eq. (2) in the strain-displacement and constitutive relations. For the computations, Young's modulus E=1 and Poisson's ratio ν=0.3 were assumed.

The problem was solved many times for different polynomial degrees of the discretization functions. In addition the degrees of freedom were gradually increased from about 200 to about 200.000 by consecutive uniform mesh refinements. For each resolution the relative displacement errors of the three methods (IGA-C, IGA-G, FEA-G) were recorded in the L^2 norm, which is evaluated as follows

$$e_{L_2} = \frac{\left\| \boldsymbol{u} - \tilde{\boldsymbol{u}} \right\|_{L^2}}{\left\| \boldsymbol{u} \right\|_{L^2}}$$

$\boldsymbol{u}$ denotes the actual solution, while $\tilde{\boldsymbol{u}}$ is the approximation to the actual solution. The convergence results with respect to the degrees of freedom are shown in Fig. 7.1 for the smooth elasticity problem and in Fig. 7.2 for the rough elasticity problem.

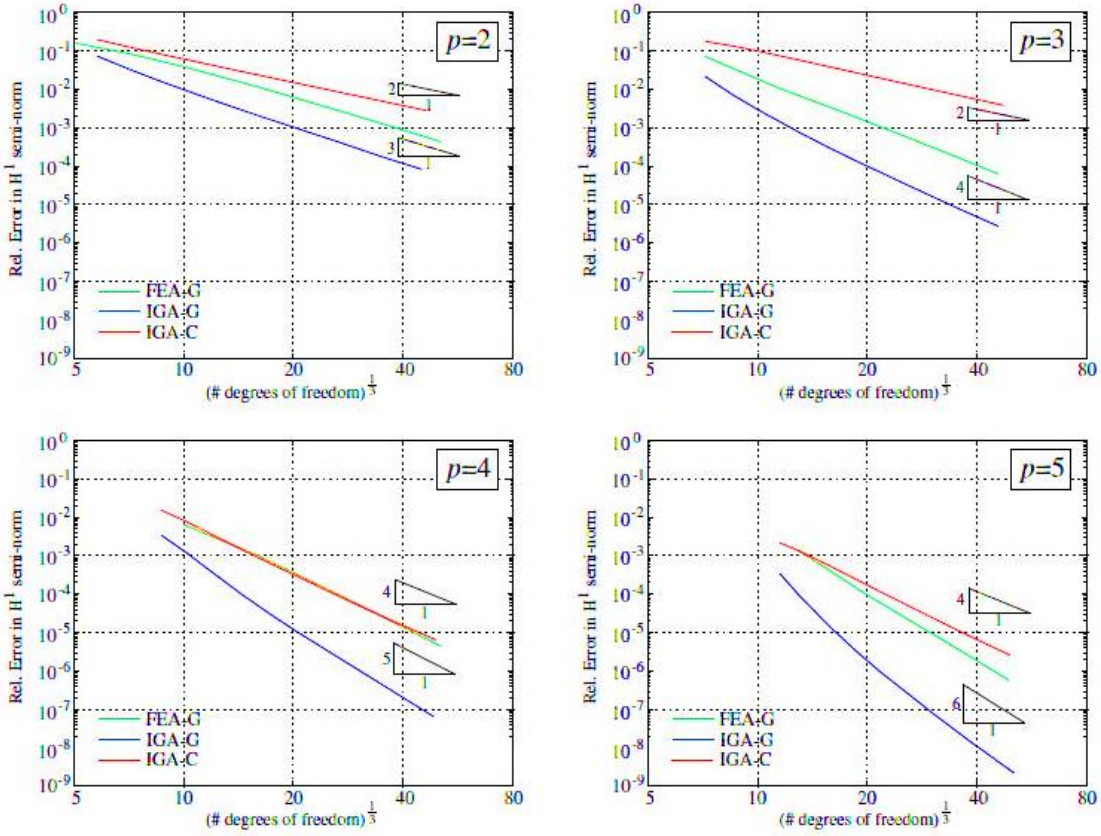


Fig. 7.1. Smooth 3D elasticity: L^2 error vs. degrees of freedom

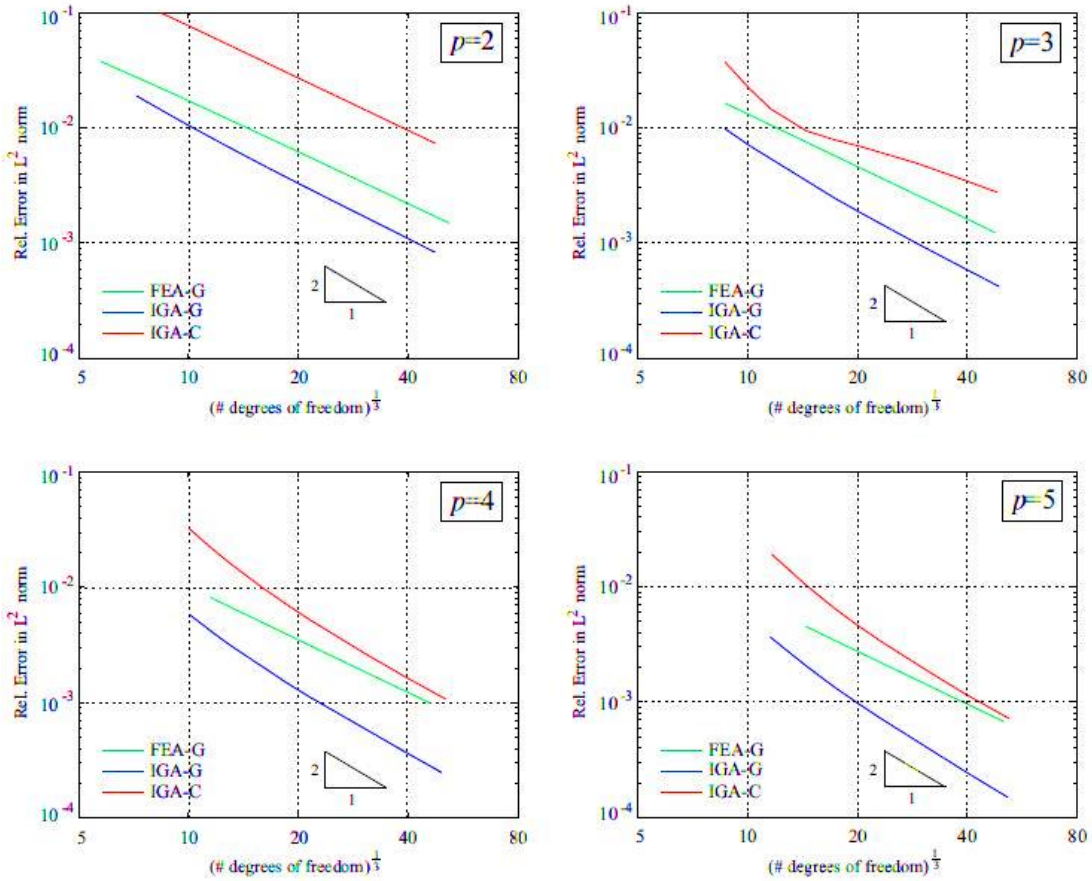


Fig. 7.2. Rough 3D elasticity: L^2 error vs. degrees of freedom

It is obvious that IGA-G achieves the best performance compared to FEA-G and IGA-C for both smooth and rough 3D elasticity problems. In the first case, in particular, IGA-G can be proud of very small normalized displacement errors, while it approaches the exact solution with a convergence rate of O(p+1). IGA-C, on the other hand, proves to be inferior of the other two methods in accuracy issues, since it always lags behind them. It should be mentioned that IGA-C performs better for even than odd polynomial degrees, considering that even degrees come with a convergence rate of O(p) in contrast to O(p-1) of odd degrees. In addition, the efficiency of the method tends to improve for higher order functions, as it can be concluded from the Fig. 7.1 focusing on charts for p=4 and p=5. However, this is not comforting enough, as the standard analysis procedure is to begin with a solution generated on a coarse mesh with a low-order basis. Provided that this solution does not satisfy the desired level of accuracy, then refinement strategies are applied, which may involve an order elevation.

Fig. 7.2, which refers to rough vector problems, reveals considerably lower rates of convergence in all methods. This phenomenon was expected, since the derivatives of the exact rough solution exhibit a singularity in the corner {x,y,z}={1,1,1}. However, IGA-G accomplishes again the most accurate results, while the reliability of IGA-C is

restored gradually as degree p is increased and its results tend to approach the attitude of FEA-G.

In defense of collocation method, what frequently matters in practical problems is the computing time required in order to achieve a specified level of accuracy. Besides, it can be claimed that for higher orders $(p > 3)$, IGA-C offers the best accuracy-to-computing-time ratios, as far as displacements and stresses are concerned.

7.2 Neumann boundary conditions

7.2.1 Introduction

As it has already been mentioned in Chapter 4, Neumann boundary conditions are strongly imposed on boundary collocation points in the isogeometric collocation method. According to [7], however, this treatment may cause a significant loss of accuracy in some problems, especially when a reduced regularity of the solution is met and when non-uniform meshes are used. In the same paper two alternative techniques are proposed, promising to reach better accuracy while preserving the computational cost at the initial level.

For the sake of simplicity a small-strain linear elasticity problem will be considered at this section, since the extension to 3-dimensional situations is analogous. The elastostatic problem in strong form is defined by

$$div\left(\mathbb{C}\nabla^{S}\boldsymbol{u}\right)+\boldsymbol{f}=\boldsymbol{0} \quad in\ \Omega$$
$$\boldsymbol{u}=\boldsymbol{g} \quad on\ \Gamma_{g}$$
$$\left(\mathbb{C}\nabla^{S}\boldsymbol{u}\right)\boldsymbol{n}=\boldsymbol{p} \quad on\ \Gamma_{p}$$

where $\boldsymbol{f}$ represents body forces, $\boldsymbol{g}$ represents prescribed displacements on a portion of the boundary Γ_g and $\boldsymbol{p}$ denotes prescribed tractions (possibly zero) on the remaining portion Γ_p. Thus $\Gamma=\Gamma_g\cup\Gamma_p$ is the boundary of the domain and $\Gamma_g\cap\Gamma_p=\varnothing$. In addition, **u(x)** is the unknown displacement field, ∇^S is the symmetric part of the gradient operator, $\mathbb{C}$ is the fourth-order elasticity tensor, *div* is the divergence operator, and $\boldsymbol{n}$ is the outward unit normal to the boundary of the domain.

Consider a 2-dimensional domain determined by $N=nm$ control points, where n and m are the number of control points in the parametric axis ξ and in the parametric axis η, respectively. It is clear that the definition of the unknown control point displacements requires $2N$ scalar equations. Greville abscissae as well as Demko abscissae yield N

collocation points τ_{kl}, k={1,…,n}, l={1,…,m} located at the tensor product structure of the knot vectors. Different sets of equations are destined for the interior collocation points and the collocation points on the Neumann boundary Γ_p. Having chosen as test functions the Dirac δ functions, the interior collocation points τ_{kl}, k={2,…,n-1}, l={2,…,m-1}, should satisfy the equations

$$\left[div\left(\mathbb{C}\nabla^S \boldsymbol{u}^h\right) + \boldsymbol{f} \right](\tau_{kl}) = 0 \qquad \tau_{kl} \subset \Omega$$

Collocation points on the Dirichlet boundary do not have to be defined, as it is imposed a priori that $u_i^h(\tau_{kl}) = g_i(\tau_{kl})$.

7.2.2 Basic collocation treatment

Before the two new proppsed techniques are described, it would be helpful to recall and further explain the "basic collocation" (BC) treatment, as it is addressed to in [7]. The evaluation of Neumann conditions at boundary and interface collocation points supposes that the outward unit normal $\boldsymbol{n}$ is well-defined. Therefore, a special treatment should be predicted for collocation points lying on areas with reduced regularity, such as corners or sharp edges where one or more planes are involved. Such a point and the corresponding outward normal vectors $\boldsymbol{n}'$ and $\boldsymbol{n}''$ are depicted in Fig. 7.3. It should be noted that this case is restricted to a single patch domain. Considering a geometry composed of multiple patches, attention should be paid in a similar way at points that belong to the common boundary of two patches, or to the common corner of two or more patches.

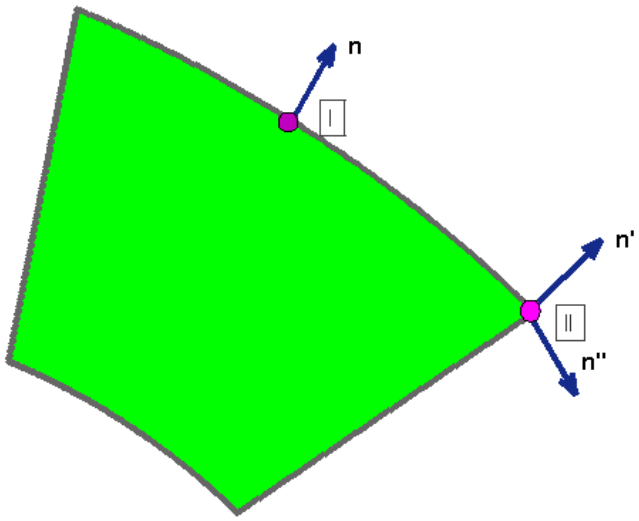


Fig. 7.3. Boundary collocation point types in a domain described by a single patch

The types shown in Fig. 7.3 refer to the following cases:

(I) point τ_{kl} is on the boundary of a single patch domain and it is not a corner
(II) point τ_{kl} is on the boundary of a single patch and it is a corner

For the points of type (II) the procedure indicates that the evaluation of the Neumann condition should take place for each one of the planes that are crossed at the corner or at the sharp edge, as a different outward normal $\boldsymbol{n}$ is defined at each one of them. The final contribution to the stiffness matrix constitutes an average of the above evaluations. Therefore, for collocation points located at the edges (k=1,n and l={2,…,m-1}, or l=1,m and k={2,…,n-1}) the strong imposition of the Neumann bcs is of the form

$$\left[\mathbb{C}\left(\nabla^S \boldsymbol{u}_h\right)\boldsymbol{n} - \boldsymbol{p}\right](\tau_{kl}) = 0 \qquad \tau_{kl} \subset \text{edge} \subset \Gamma_p$$

For collocation points located at corners (k=1,n and l=1,m) [11] showed that the suitable equations are

$$\left[\mathbb{C}\left(\nabla^S \boldsymbol{u}_h\right)\boldsymbol{n}' - \boldsymbol{p}'\right](\tau_{kl}) + \left[\mathbb{C}\left(\nabla^S \boldsymbol{u}_h\right)\boldsymbol{n}'' - \boldsymbol{p}''\right](\tau_{kl}) = 0 \qquad \tau_{kl} \equiv \text{edge} \subset \Gamma_p$$

where $\boldsymbol{p}'$ and $\boldsymbol{p}''$ are the tractions imposed on the edges meeting at the corner.

7.2.3 Hybrid collocation treatment

Hybrid collocation (HC) as described at [7] suggests that the Galerkin treatment is adopted for the equations at the Neumann boundary, while the operations up to this point continue complying with the collocation rules. Some of the shape functions used for the problem discretization are used as test functions and thus the integrations are not eliminated. Specifically, these shape functions pertain to the control points located at the edges of the patch domain. For instance, edges with $k = \bar{k} = 1, n$ and $l = \{2,...m-1\}$ stimulate 2(m-2) equations of the form

$$\int_{\Omega}\left[div\left(\mathbb{C}\nabla^S \boldsymbol{u}^h\right) + \boldsymbol{f}\right]R_b d\Omega - \int_{\Gamma_{p\bar{k}}}\left[\mathbb{C}\left(\nabla^S \boldsymbol{u}^h\right)\boldsymbol{n}_{\bar{k}} - \boldsymbol{p}_{\bar{k}}\right]R_b d\Gamma = 0 \tag{3}$$

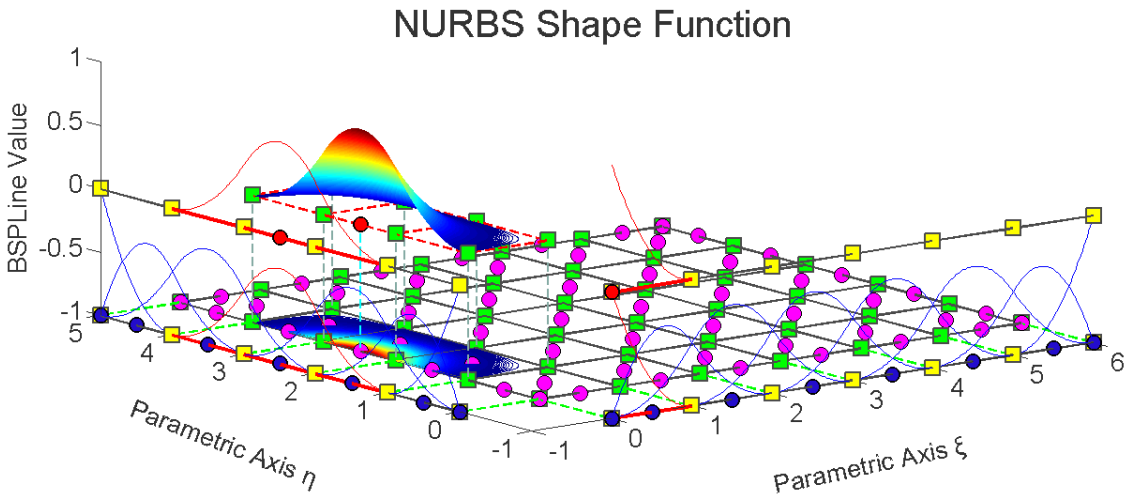
where $\Gamma_{p\bar{k}}$ denotes the considered edge within the Neumann boundary and $\boldsymbol{n}_{\bar{k}}$ and $\boldsymbol{p}_{\bar{k}}$ are the respective outward unit normal and applied traction. Index b is equal to $m\left(\bar{k}-1\right)+l$, with the assumption that the counting runs first the parametric axis η and

then moves to the next point of the parametric axis ξ. In a similar way, the points at edges with $l=\bar{l}=1,m$ and $k=\{2,...n-1\}$ should satisfy the 2(*n*-2) equations

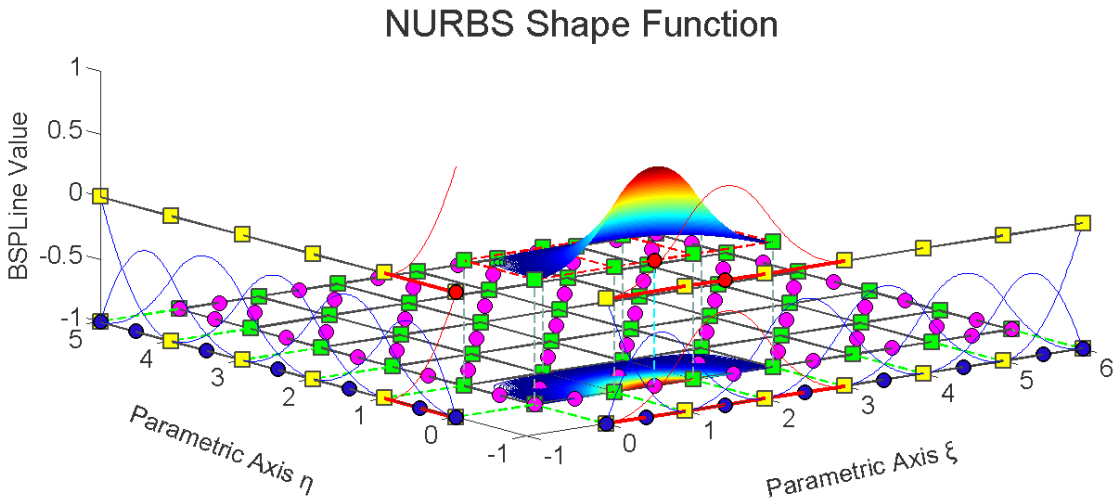
$$\int_{\Omega}\left[div\left(\mathbb{C}\nabla^{S}\boldsymbol{u}^{h}\right)+\boldsymbol{f}\right]R_{c}d\Omega-\int_{\Gamma_{p\bar{l}}}\left[\mathbb{C}\left(\nabla^{S}\boldsymbol{u}^{h}\right)\boldsymbol{n}_{\bar{l}}-\boldsymbol{p}_{\bar{l}}\right]R_{c}d\Gamma=0 \tag{4}$$

where $\Gamma_{p\bar{l}}$ denotes the considered edge within the Neumann boundary and $\boldsymbol{n}_{\bar{l}}$ and $\boldsymbol{p}_{\bar{l}}$ are the respective outward unit normal and applied traction. Index *c* is equal to $m(k-1)+\bar{l}$.

Note that integration is required over both the area and edge domains. The integrals can be evaluated with Gauss-Legendre quadrature. It should be mentioned that the shape functions R_b and R_c, which are involved in Eqs. (3) and (4) correspond to the control points lying on the Neumann edges of the patch domain. Due to the nature of the open knot vector, where the first and the last knot are repeated *p*+1 times creating this way trivial knot value spans, R_b and R_c have a reduced support relative to (*p*+1)(*q*+1) knot value spans. In particular, shape functions R_b have support over (1)(*q*+1) knot value spans, while shape functions R_c have support over (*p*+1)(1) knot value spans, where *p* and *q* are the polynomial degrees in axes ξ and η, respectively. Consequently, the number of Gauss points, which will participate in the numerical quadrature, is reduced.



(a)



(b)

Fig. 7.4. Reduced support of shape functions:

(a) $R_{i,j}^{p,q} = R_{1,4}^{2,2}$ with a support of (1)(q+1)=1(2+1)=3 elements

(b) $R_{i,j}^{p,q} = R_{3,1}^{2,2}$ with a support of (p+1)(1)=(2+1)(1)=3 elements

Without a doubt, the insertion of integrals in the resolution process increases the computational cost in comparison with the BC approach. However, the hybrid treatment still benefits from the advantageous, from the aspect of time, terms of the traditional collocation method in the majority of points. Furthermore, it leads to better accuracy than the BC in certain occasions of non-uniform meshes.

7.2.4 Enhanced collocation treatment

Still, another approach, which aims at the accuracy of the Galerkin scheme via the cost of the pure collocation technique, is proposed at [7] as an enhanced collocation treatment. The procedure is free of integrals, it tries to comply though with the results of HC with the help of a suitable constant C^*. As in the HC case, the Neumann bcs are written considering a combination of both area and edge terms, as follows

$$\left[div\left(\mathbb{C}\nabla^S \boldsymbol{u}^h\right) + f \right](\tau_{kl}) - \frac{C^*}{h}\left[\mathbb{C}\left(\nabla^S \boldsymbol{u}^h\right)\boldsymbol{n} - \boldsymbol{p} \right](\tau_{kl}) = 0 \qquad \tau_{kl} \subset \text{edge} \subset \Gamma p \tag{5}$$

where h is the mesh size in the direction perpendicular to the edge. This size is here computed as the distance between the first two collocation points encountered starting from the edge and traveling in the parametric direction perpendicular to the edge. It

should be noted that the first term of Eq. (5) involves the evaluation of the second-order operator for the boundary collocation points too, while it was so far required only for interior points. As a result, the minimum continuity of C^1 on the Neumann boundary, that assures the well-definition of the first derivatives of the shape functions, should be now raised at C^2.

As far as the constant C^* is concerned, it is defined so as to minimize the divergence of the results with respect to those obtained with the classical Galerkin formulation. Some simple and typical problems are solved with both the Galerkin method and the enhanced collocation treatment as a function of C^* as well. Then the displacement error relative to the Galerkin solution, defined as

$$e_{L^2}^{Gal} = \frac{\left\| \boldsymbol{u}^h - \boldsymbol{u}_G^h \right\|_{L^2}}{\left\| \boldsymbol{u}_G^h \right\|_{L^2}},$$

is computed for various values of the constant C^*. The optimal value of C^* is estimated when the norm $e_{L^2}^{Gal}$ exhibits a definite minimum and therefore the best convergence is achieved. This optimal value is considered to be operator-dependent but no problem-dependent, consequently it is expected to suit all problems of the same type.

7.2.5 Performance

A crucial factor for the collocation efficiency is the geometry of the mesh. The strong enforcement of Neumann bcs leads to inaccurate results when the distance between consecutive collocation points perpendicular to the boundary is sufficiently larger than parallel to the boundary. This proportion is determined by the ratio of control points in each direction. Numerical examples showed that BC exhibits significant spurious oscillations as the aspect ratio increases, while HC and EC have a stable performance in all occasions. This phenomenon of oscillations tends to diminish when denser meshes are used. However it can still be distinguishable at BC results, mainly in the vicinity of the corners, while it disappears when HC or EC are used.

Undoubtedly, the Galerkin method outperforms basic collocation technique and its improved variations regardless of the degree and the mesh aspect ratio. EC and HC achieve a satisfying approach to the Galerkin performance for high orders (higher than 4). EC, though, delivers the same level of accuracy, at a significantly lower computational cost. Thus EC appears as the best choice among collocation methods as the polynomial order increases with respect to both accuracy and computational cost.

Probably, the efficiency of EC treatment should be dealt with cautiousness due to the need of calibration of the constant C^*. This constant may be expected to depend

exclusively on the operator and not on the problem, but this prediction cannot be confirmed, since the method has not been tested yet in a wide range of examples. Besides, the relative paper addresses exclusively the small-deformation elasticity operator. Equivalent thorough numerical studies are required in order to enable EC to be applicable in other operator types as well, such as temperature variation and electromagnetism. Of course the success is not guaranteed.

Apart from that, suppose that EC has a good perspective in all conditions, the problem of increased inaccuracy for low discretization orders still remains. According to the numerical applications HC and EC have no gains compared to BC for p=2 and p=3. As it has already been explained though, these degrees are of major importance for the analysis strategy.

7.3 Required continuity

Galerkin scheme forms the stiffness matrix via the numerical quadrature of integrals, thus the minimum continuity required by the basis functions is reduced to C^0. This is why Galerkin has dominated over the standard C^0-continuous FEA technology. Besides, Gauss points lie at the interior of each element where C^∞ is ensured due to the polynomial nature of the basis. Computations over inter-element boundaries are not involved in the analysis procedure. Therefore, the continuity over knots is allowed to be as minimum as possible without affecting the result.

On the other hand, collocation collaborates with the strong form of the PDE system. Higher derivatives have to be well-defined at each collocation point, thus basis functions with specific smoothness properties are required. These properties should enable the evaluation of the highest differential operator that appears at each PDE. Despite the fact that isogeometric analysis comes with some degree of smoothness across element boundaries, the use of high discretization orders is neither convenient nor desirable for the analysis purposes.

For the scope of this thesis small-strain linear elasticity problems were studied, which correspond to the second-order operator L. Consequently, the basis functions should be at least C^2 at interior collocation points and at least C^1 on the Neumann boundary. Assuming that other types of problems are considered, it is possible that the demands for high continuity will be increased. For instance, the Bernoulli-Euler beam and Kirchhoff plate models represent thin structural problems, where the differential equation governing the boundary value system is of fourth-order. In this case, the collocation scheme would require a basis of polynomial degree at least equal to 4.

In order to be more precise, it should be noted that the requisite smoothness properties differ slightly from odd to even degrees. When odd degrees are called on, collocation points are located directly at knots, therefore the continuity should identify with the order of the operator. Provided that a linear elasticity problem is concerned and *p* is odd, the continuity should be set at least to C^2 at all interior collocation points. Even degrees on the contrary situate collocation points in the center of a knot span, thus C^1 would be sufficient for the example of elasticity. Probably, someone would expect that C^0 would also be suitable, considering the C^∞ continuity in the element interior. However this is not the case, since the regularity of the knots determines the location of collocation points. If a knot is repeated up to *p* times, the Greville formula yields a collocation point on the specific knot. Clearly C^0 continuity does not support the evaluation of the differential operator and it is thus excluded from collocation formulations.

To better illustrate this, consider the following quadratic basis functions:

$$N_{0,2} = (1-\xi)^2 \qquad 0 \le \xi < 1$$

$$N_{1,2} = \begin{cases} 2\xi - {}^3\!/_2\,\xi^2 & 0 \le \xi < 1 \\ {}^1\!/_2\,(2-\xi)^2 & 1 \le \xi < 2 \end{cases}$$

$$N_{2,2} = \begin{cases} {}^1\!/_2\,\xi^2 & 0 \le \xi < 1 \\ -{}^3\!/_2 + 3\xi - \xi^2 & 1 \le \xi < 2 \\ {}^1\!/_2\,(3-\xi)^2 & 2 \le \xi < 3 \end{cases}$$

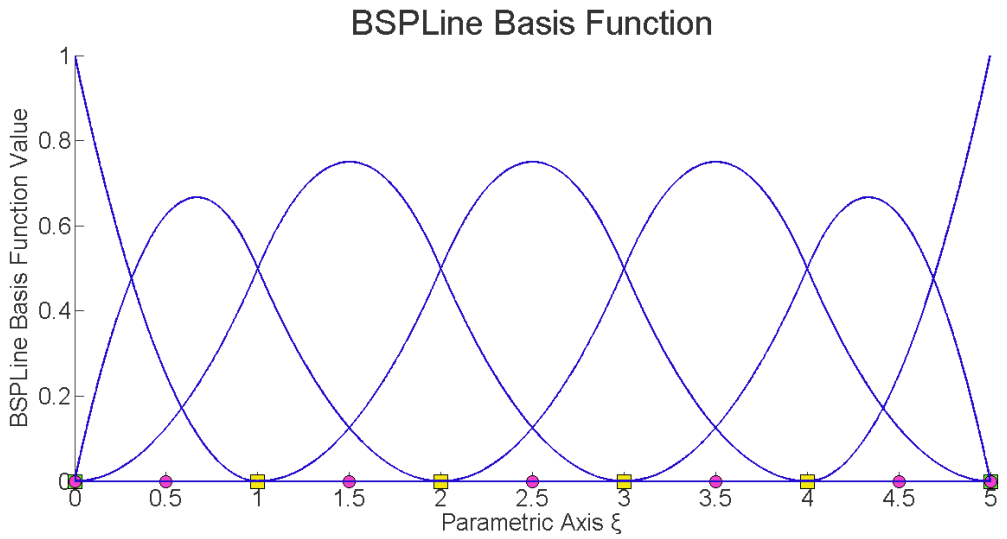
$$N_{3,2} = \begin{cases} {}^1\!/_2\,(\xi-1)^2 & 1 \le \xi < 2 \\ -{}^{11}\!/_2 + 5\xi - \xi^2 & 2 \le \xi < 3 \\ {}^1\!/_2\,(4-\xi)^2 & 3 \le \xi < 4 \end{cases}$$

$$N_{4,2} = \begin{cases} {}^1\!/_2\,(\xi-2)^2 & 2 \le \xi < 3 \\ -16 + 10\xi - {}^3\!/_2\,\xi^2 & 3 \le \xi < 4 \end{cases}$$

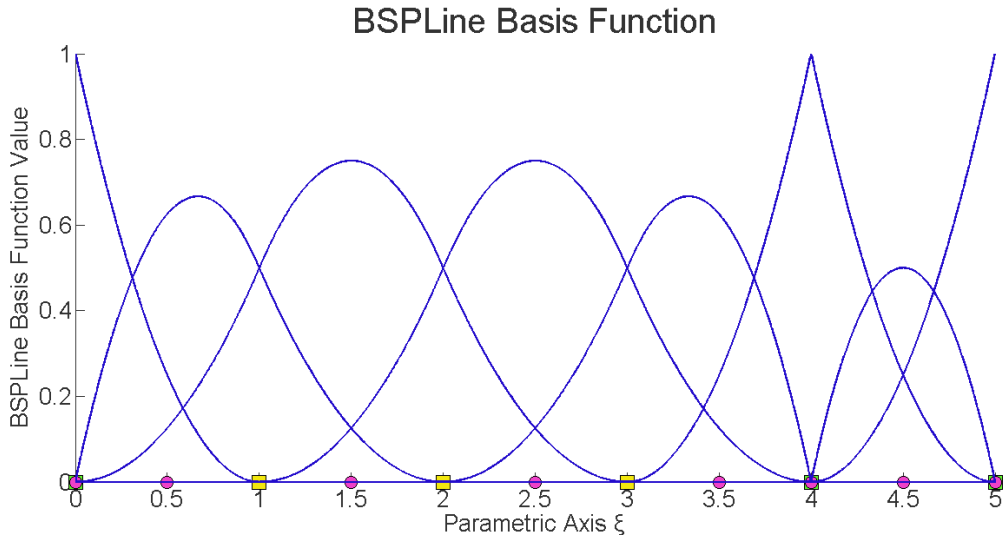
$$N_{5,2} = \begin{cases} (\xi-3)^2 & 3 \le \xi < 4 \\ (5-\xi)^2 & 4 \le \xi < 5 \end{cases}$$

$$N_{6,2} = 2(\xi-4)(5-\xi) \qquad 4 \le \xi < 5$$

$$N_{7,2} = (\xi-4)^2 \qquad 4 \le \xi < 5$$



(a)



(b)

Fig. 7.5. Alteration to collocation points in case of C^0 continuity
(a) Ξ={0 0 0 1 2 3 4 5 5 5}
(b) Ξ={0 0 0 1 2 3 4 4 5 5 5}

Observe that the repetition of the knot with parametric coordinate ξ=4, creates the need for one additional control point and therefore one additional collocation point. The extra collocation point lies at ξ=4, while the rest of them are located in the middle of the corresponding elements. The fact that the curve is interpolatory at that point indicates that not even the first derivative is well defined. This claim is further supported by the following computation:

$$\left.\frac{dN_{5,2}}{du}\right|_{\xi=4} = \left.\begin{cases} 2(\xi-3) & 3 \le u < 4 \\ -2(5-\xi) & 4 \le u < 5 \end{cases}\right|_{\xi=4} = \begin{cases} 2 \\ -2 \end{cases}$$

In general, in collocation formulations the level of continuity should always comply with the order of the highest differential operator. This may lead to overdone polynomial degrees, even in problems where the desired accuracy could be achieved with much lower basis order.

It also becomes clear that collocation offers no practical usefulness to the finite element method. FEA lacks of smoothness, as it addresses exclusively to C^0-continuous basis functions. It cannot support the development of methods based on the strong form of differential equations.

8 Conclusions

Isogeometric analysis

Isogeometric analysis is an innovative evolution of the standard FEM that manages to bridge the gap between CAD and CAE technologies. The exact geometrical mesh and not an approximation to the physical model feeds the analysis process, while all the required alterations to the geometry are adapted to the initial representation. The geometric error is thus eliminated and the method outperforms FEM in terms of accuracy.

The concept of IGA is implemented through the technologies used to describe the solution filed. NURBS and B-splines have dominated CAD industry and they are fairly the first candidate for isogeometric applications. Local support and C^{p-m} continuity across each knot are some major properties that differentiate current basis functions from those used in classical FEA. These properties imply that shape functions hold an overlapping that leads to greater interconnectivity among elements and they also enable a more accurate approach to the natural response of an object, since C^0 continuity is not met in a continuum model.

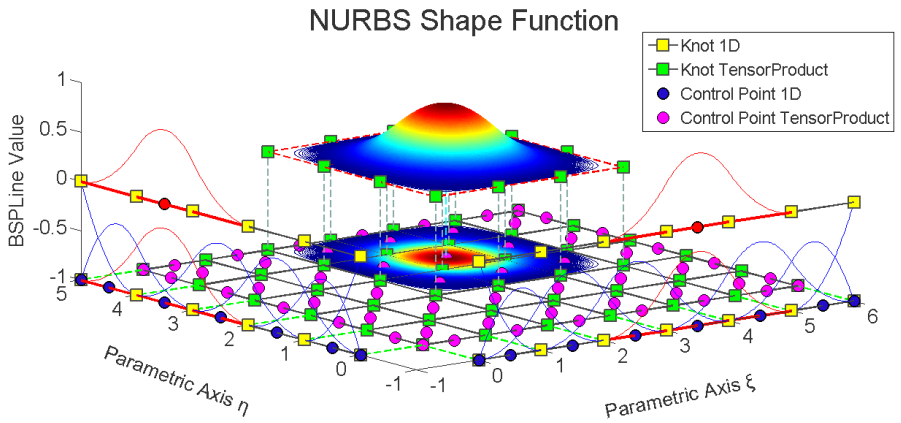


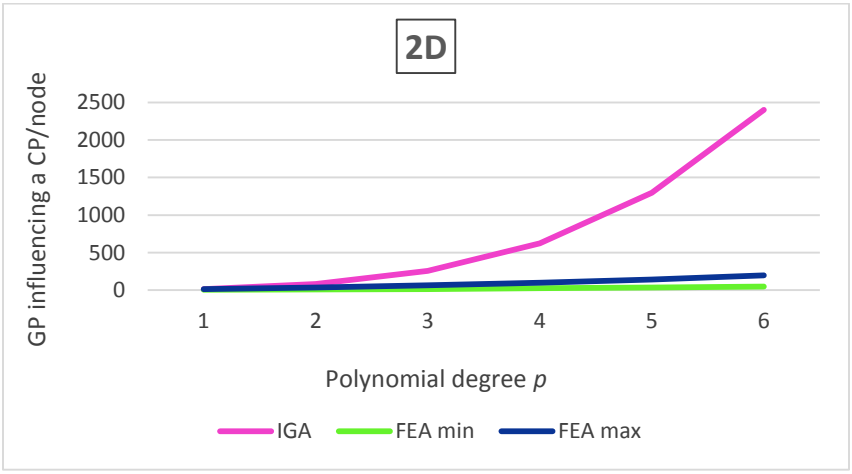
Fig. 8.1. Shape function $R_{i,j}^{p,q} = R_{5,5}^{2,2}(\xi,\eta)$ as a tensor product of $N_{5,2}(\xi)$ and $M_{5,2}(\eta)$
Ξ={0 0 0 1 2 3 4 5 6 6 6}
H={0 0 0 1 2 3 4 5 5 5}

The price of higher accuracy per degree of freedom compared to finite element analysis is the increased computational cost for the formation and assembly of the

stiffness matrix. Overlapping comes with denser system matrices, therefore more elaborate and time-consuming calculations are required.

Numerical quadrature

The blame for the high cost of isogeometric analysis is imputed to numerical integration. Galerkin scheme, adopted in both FEM and IGA in order to approximate solutions to the boundary value problems, introduces integrals which have to be evaluated by numerical quadrature rules. Gauss rules suggest that $(p+1)^d$ points per knot span are defined for two and three dimensional cases. This applies to both methods, however the number of control point-Gauss point correlations is significantly increased in IGA due to the overlapping of basis functions. Overlapping raises the number of elements and subsequently the number of Gauss points influencing a specific control point by orders of magnitude compared to FEA. Besides, IGA produces quite more elements than FEA for the same number of degrees of freedom, because of its higher inter-element continuity. This means that the total number of Gauss points is dramatically increased.



(a)

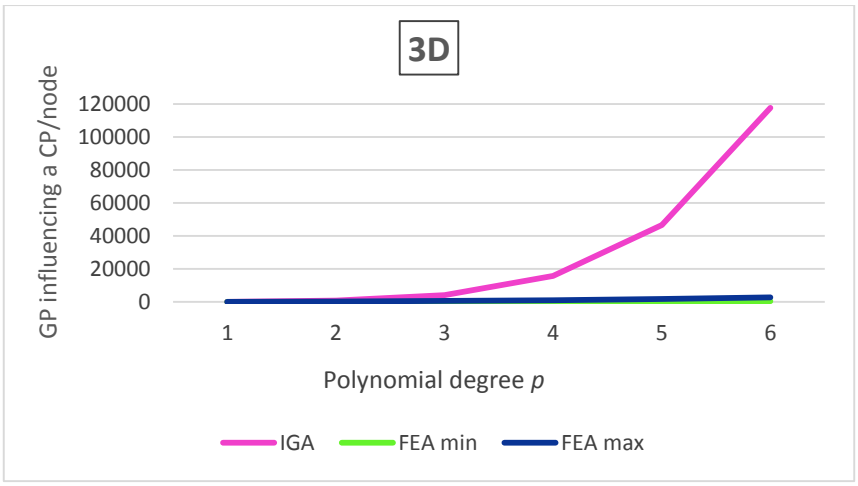


Fig. 8.2. Number of Gauss points influencing a control point/node with respect to *p*

Apparently, efficient quadrature for NURBS-based isogeometric analysis is not translated into the use of standard element-wise Gauss rules. Recently proposed integration rules recommend that the number of quadrature points complies with the number of degrees of freedom and it is no more so closely related to the polynomial order *p*. These methodologies respect the smoothness properties of B-splines across element boundaries and pursue an optimal number of point evaluations.

Collocation scheme

Collocation scheme is an alternative to Galerkin, which supports the discretization of the strong form of the governing partial differential equations. This means that integrals are eliminated and no quadrature techniques are required. The number of collocation points is equal to the number of control points, so IGA-C achieves one point evaluation per basis function. The calculations over each Greville point correspond to the entries of one row of the global stiffness matrix. The procedure of assembly is now pointless, something that leads to a profound reduction of the operations required for the system matrix formation. The final matrix is generally non symmetric, considering that rows refer to collocation points and columns to shape functions. This also affects bandwidth, which is reduced in case of IGA-C.

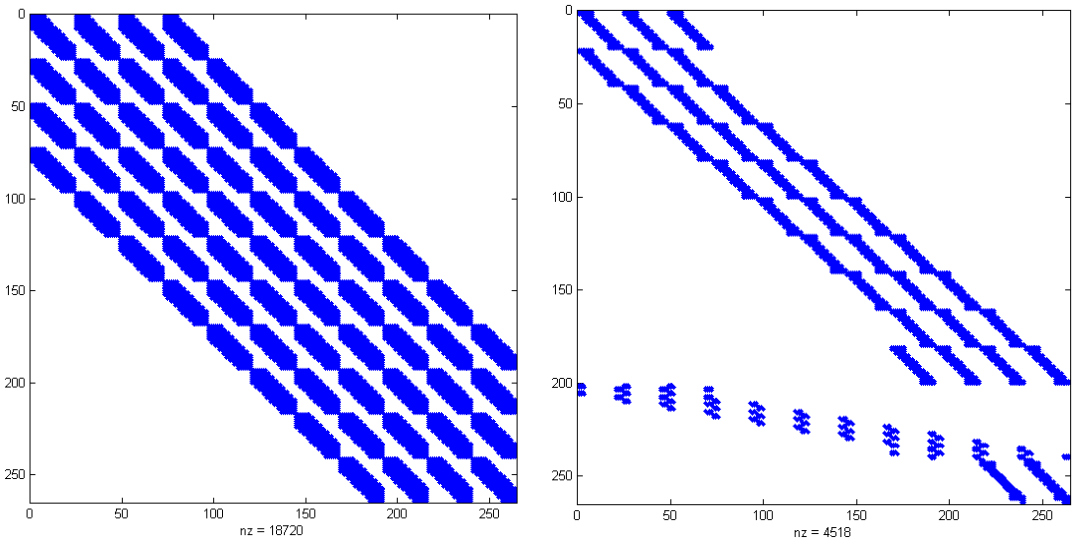


Fig. 8.3. Plane strain plate supported on the left edge: stiffness matrix in **(a)** Galerkin, **(b)** collocation

Collocation entails the evaluation of the differential operators of the PDE, as a result basis functions have to come with specific smoothness properties depending on the problem. C^0 continuity is not permitted under any circumstances, while low discretization orders are excluded from a wide range of problems, where higher derivatives are required.

Comparison with Galerkin

Galerkin formulation achieves better accuracy per degree of freedom compared to IGA-C. Displacement errors in the L^2 norm prove that isogeometric collocation will need much more degrees of freedom than FEA-G in order to reach a certain level of accuracy.

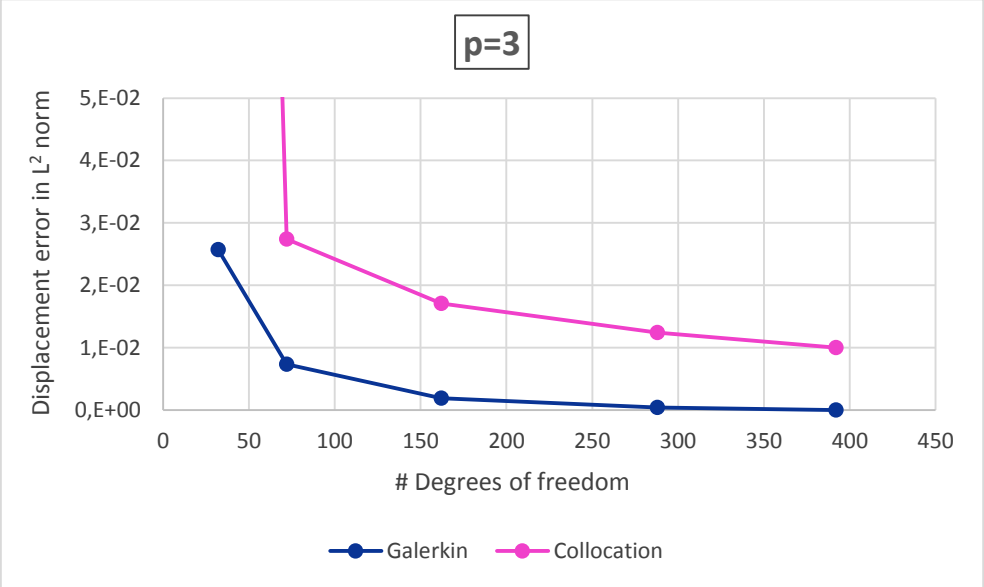


Fig. 8.4. Displacement error in L^2 norm vs. degrees of freedom

As far as computing time is concerned, IGA-C is expected to be faster than IGA-G, despite the fact it may require denser meshes in order to ensure the desired convergence. This conclusion mostly applies to high orders (p>3).

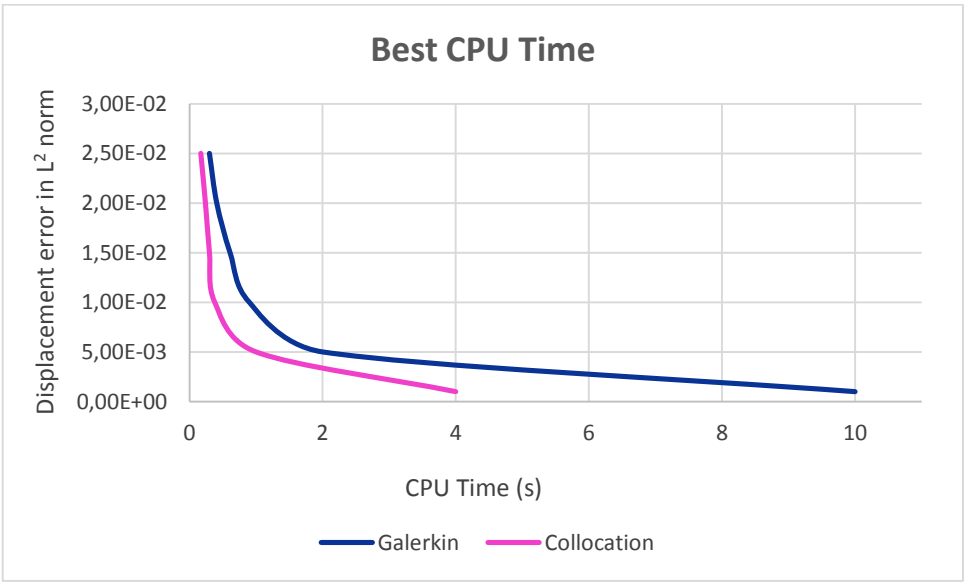


Fig. 8.5. Displacement error in L^2 norm vs. time

Of course the conclusions concerning computational cost are based on a primary effort to evaluate the two methodologies, which does not take advantage of existing optimization algorithms and thus not the optimum efficiency is acquired. Nevertheless, these results offer a clue of each method's prospects and they stimulate further theoretical and numerical research on stability and convergence behavior of collocation scheme.

Future research

Isogeometric collocation seems to be really promising for engineering analysis with many significant potential benefits. Its major advantage is the reduced number of evaluation points, therefore it can dominate fields where the efficiency of the analysis technology is closely related to the cost of quadrature. Explicit structural dynamics is a perfect match for the above description, considering that the calculation of the residual of the force vector via stress divergence evaluation at quadrature points consumes the larger part of the total analysis time. A first approach to dynamic cases has already been achieved for one-dimensional and two-dimensional configurations. However, a complete mathematical analysis has not been reached yet, while much research is demanded for the extension to three-dimensional solids.

In addition to that, the methodology of isogeometric collocation should be adapted to hierarchical refinement techniques. Hierarchical building of NURBS basis functions proves to be of major importance for improving the performance of IGA, as it opens the door to local refinement of NURBS parameterizations. Unfortunately, collocation as described in the present study fails to serve the needs of a hierarchical basis. Greville points of different levels may be coincident and in this case linear independence is unavoidable. For this reason the option of a weighted collocation scheme should be explored. A scheme like that suggests that the evaluation of the PDE takes place at several collocation points and then a weighted average of their contributions is used. This treatment will not impact the privilege of a minimum number of point evaluations, since it will be restricted to the transition regions.

Finally, IGA-C is expected to have a great performance in parallel implementations, considering that it does not involve the procedure of assembly. During this procedure rows of the element stiffness matrices are sent to the processor of the global array and complications are possible to arise. Probably, IGA-C subjected to parallelization techniques will be able to reach even greater performance as far as computation cost is concerned, while it will offer advantages for minimizing memory storage and access.

References

[1] Isogeometric Analysis: Toward Integration of CAD and FEA - J. Austin Cottrell, Thomas J.R. Hughes, Yuri Bazilevs - Wiley, 2009

[2] Analysis of Structures with the Finite Element Method - M. Papadrakakis - Papasotiriou, 2001

[3] The NURBS Book - L. Piegl, W. Tiller - Springer, 1997

[4] Efficient quadrature for NURBS- based Isogeometric Analysis - T. J.R. Hughes, A. Realli, G. Sangalli - 2008

[5] GPU accelerated computation of the isogeometric analysis stiffness matrix- A. Karatarakis, P. Karakitsios, M. Papadrakakis- Elsevier, 2013

[6] Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations – Dominik Schillinger, John A. Evans, Alessandro Reali, Michael A. Scott, Thomas J.R. Hughes – Elsevier, 2013

[7] Isogeometric collocation: Neumann boundary conditions and contact – L. De Lorenzis, J.A. Evans, T.J.R. Hughes, A. Reali - 2014

[8] Isogeometric Collocation Methods – F. Auricchio, L. Beirão da Veiga, T.J.R. Hughes, A. Reali and G. Sangalli - 2010

[9] An Introduction to Isogeometric Collocation Methods – Alessandro Reali, Thomas J.R. Hughes - 2014

[10] An isogeometric collocation approach for Bernoulli-Euler beams and Kirchhoff plates - Alessandro Reali, Hector Gomez – Elsevier, 2014

[11] Isogeometric collocation for elastostatics and explicit dynamics – F. Auricchio, L. Beirão da Veiga, T.J.R. Hughes, A. Reali, G. Sangalli – Elsevier, 2012

[12] A weighted collocation on the strong form with mixed radial basis approximations for incompressible linear elasticity – Sheng-Wei Chi, Jiun-Shyan Chen, Hsin-Yun Hu – Springer, 2013

[13] Isogeometric Analysis, CAD, finite elements, NURBS, exact geometry and mesh refinement - Hughes, Cottrell, Bazilevs - 2004

[14] Collocation On Hp Finite Element Meshes: Reduced Quadrature Perspective, Cost Comparison With Standard Finite Elements, And Explicit Structural Dynamics - Dominik Schillinger, John A. Evans, Felix Frischmann, Rene R. Hiemstra, Ming-Chen Hsu, Thomas J.R. Hughes - 2014

[15] Assessment of Computational Efficiency of Numerical Quadrature Schemes in The Isogeometric Analysis – Daniel Rypl, Borek Patzák – Engineering MECHANICS, 2012

[16] A point collocation method based on reproducing kernel approximations – N.R. Aluru - 2000

[17] A simple algorithm for obtaining nearly optimal quadrature rules for NURBS-based isogeometric analysis - F. Auricchio, F. Calabrò, T.J.R. Hughes, A. Reali, G. Sangalli

[18] Isogeometric contact: a review – L. De Lorenzis, P. Wriggers, T.J.R. Hughes - 2014
[19] Isogeometric collocation methods for the Reissner-Mindlin plate problem – J. Kiendl, F. Auricchio, L. Beirão da Veiga, C. Lovadina, A. Reali – Elsevier, 2014
[20] A fully ‘‘locking-free’’ isogeometric approach for plane linear elasticity problems - Auricchio, Veiga, Buffa, Lovadina, Reali, Sangalli - 2007
[21] An introduction to IGA with Matlab implementation, FEM and XFEM Formulations - Nguyen, Simpson, Bordas, Rabczuk - 2012
[22] Isogeometric Analysis Primer – Bernd Simeon, Anh-Vu Vuong
[23] Accurate, efficient and (iso)geometrically flexible collocation methods for phase-field models – Hector Gomez, Alessandro Reali, Giancarlo Sangalli – Elsevier, 2014
[24] Finite Element Method – István Moharos, István Oldal, András Szekrényes