# ASSESSMENT OF COMPUTATIONAL EFFICIENCY OF NUMERICAL QUADRATURE SCHEMES IN THE ISOGEOMETRIC ANALYSIS

Daniel Rypl*, Bořek Patzák*

*Isogeometric analysis (IGA) has been recently introduced as a viable alternative to the standard, polynomial-based finite element analysis. One of the fundamental performance issues of the isogeometric analysis is the quadrature of individual components of the discretized governing differential equation. The capability of the isogeometric analysis to easily adopt basis functions of high degree together with the (generally) rational form of those basis functions implies that high order numerical quadrature schemes must be employed. This may become computationally prohibitive because the evaluation of the high degree basis functions and/or their derivatives at individual integration points is quite demanding. The situation tends to be critical in three-dimensional space where the total number of integration points can increase dramatically. The aim of this paper is to compare computational efficiency of several numerical quadrature concepts which are nowadays available in the isogeometric analysis. Their performance is assessed on the assembly of stiffness matrix of B-spline based problems with special geometrical arrangement allowing to determine minimum number of integration points leading to exact results.*

Keywords: *isogeometric analysis, numerical quadrature, Gaussian quadrature, Bézier extraction, half-point rule, exact B-spline quadrature rule*

## 1. Introduction

The concept of the isogeometric analysis [1, 2], initially motivated by the gap between the computer aided design (CAD) and the finite element analysis (FEA), builds upon the concept of isoparametric elements, in which the same shape functions are used to approximate the geometry and the solution on a single finite element. The IGA, as its name suggests, goes one step further because it employs the same functions for the description of the geometry and for the approximation of the solution space on that geometry. This implies that the isogeometric mesh (discretization for computational purposes) of the CAD geometry encapsulates the exact geometry no matter how coarse the mesh actually is. As a consequence, the need to have a separate representation for the original CAD model and another one for the actual computational geometry is completely eliminated.

The isogeometric approach [1] has been originally developed using the NURBS (non-uniform rational B-splines [3, 4]) which are the basic building blocks in most CAD systems and which allow precise representation of wide class of objects (e.g. conics and quadrics). To overcome several drawbacks related to handling of NURBS patches (propagation of the refinement through the entire control grid, difficult merging of adjacent patches and handling of

* doc. Dr. Ing. D. Rypl, prof. Dr. Ing. B. Patzák, Department of Mechanics, Faculty of Civil Engineering, Czech Technical University in Prague

trimmed patches, etc.), this approach has been recently extended to so-called T-splines [5, 6] which are a generalization of NURBS. The advantage of T-splines consists in the fact that they allow local refinement, without propagating the entire row of control points (by creating a T-junction), which enables efficient merging of several NURBS patches of different parameterization into a single gap free model of $C^0$ or higher order continuity [6, 7]. However, linear independence of T-spline blending functions impose some limitations [8] on local refinement, which has lead to the introduction of the class of so-called 'analysis suitable' T-splines [9]. Recently, this problem has been overcome by the concept of hierarchical B-splines [10] which seems quite promissing for the IGA.

It has been shown [1, 11, 12, 13, 14] that the IGA outperforms the classical FEA in various aspects (accuracy, robustness, system condition number, etc.), which is the consequence of several important advantages of the IGA compared to the FEA. On the other hand, the computational effort of the IGA, especially when using higher order basis functions, seems to exceed that for the FEA. The significant source of the computational inefficiency has been identified to be related to the numerical quadrature of individual components of the discretized governing differential equation (for example in the context of structural mechanics, of stiffness matrix, mass matrix, load vector, etc.). The basic computational scheme of the IGA resembles very much that of the FEA with the only difference that instead of performing the numerical quadrature on individual finite elements the quadrature is accomplished over individual non-zero knot spans[1] of the underlying B-spline based geometry. Due to the tensor product structure of the basis functions on individual knot spans of a two- and three-dimensional B-spline patch, the Gaussian quadrature schemes used for (so much popular) quadrilateral and hexahedral finite elements can be readily adopted in the IGA.

Analogical concept of Gaussian quadrature is also offered by the Bézier extraction approach [15, 16] typically used when implementing the IGA into existing finite element computational codes. This approach utilizes the fact that the smooth B-spline basis can be constructed as a linear combination of a $C^0$ Bernstein polynomials which are the basis functions on the so-called Bézier element. Note that the coefficients of the linear combination are dependent only on the parameterization of the B-spline patch and are independent of the geometry (position of control points) itself. The beauty of this approach consists in the fact that the code does not have to implement the B-spline technology. It is enough to implement rather simple Bernstein polynomials in the interpolation engine (similarly as the standard Lagrange polynomials) and to apply appropriate linear operator (the so-called extraction operator) which hides the transformation between the $C^0$ Bernstein basis and smooth B-spline basis and which is typically part of the input data. Since the individual Bézier elements correspond to the individual non-zero knot-spans, the Gaussian integration over individual Bézier elements seems to be equivalent to the Gaussian integration over the individual non-zero knot spans. There is, however, one important difference. Because the Bernstein polynomials are defined over the same parametric domain (typically from 0 to 1) and because the degree of Bernstein basis is the same for all Bézier elements within a single B-spline patch, the values of individual Bernstein basis functions and their derivatives are the same at individual Gauss integration points on all Bézier elements and can be therefore precomputed (only once) and stored (also only once) thus saving potentionally a huge number (depending on the number of integration points) of their evaluations.

[1] In the context of the IGA, the non-zero knot spans are often called elements.

Recently, there has been initiated a study [17] on efficient quadrature schemes for the NURBS-based IGA which profits from the continuity of higher degree B-spline basis functions between adjacent knot spans compared to the $C^0$ continuity of basis functions between classical finite elements. While the Gaussian quadrature is optimal for the $C^0$ continuous finite elements, it is far from optimal for $C^{p-1}$ continuous B-spline basis functions of order $p$ spanning several consecutive knot spans. By taking into account the smoothness of the basis functions across boundaries of infinite number of uniform knot spans, a simple integration rule (the so-called half-point rule) independent (in terms of the number of integration points, not in terms of their location) of the degree of the polynomial basis and having (in 1D) just one integration point per two knot spans has been derived. For practical purposes, however, integration rules corresponding to open non-uniform finite knot vector are desirable. These rules can be obtained by the numerical solution of a system of non-linear equations which is computationally demanding and which is worth only if the rules are applied repeatedly many times within the same analysis. Therefore only rules on 2, 3, 4, or 5 consecutive uniform knot spans for few cases of degree of practical interest (thereafter denoted as exact B-spline integration rules) have been derived. Although these rules only approach the best possible performance, the savings, especially in 3D, are significant.

The aim of this paper is to compare the efficiency of the above three approaches within the same software (OOFEM [18]) using the same programming techniques. The results of the comparison are given in the following Section. The discussion of the results together with the concluding remarks are given in Section 3.

## 2. Comparison of computational efficiency of individual quadrature schemes

Although the discussed quadrature schemes are used generally for (only approximate) integration of rational functions, they can handle precisely only polynomials. Therefore the examples (see Figure 1) on which the computational efficiency of quadrature schemes has been assessed are chosen to be B-spline patches with orthogonal system of isoparametric curves with control points defined by Greville's coordinates [19], which are defined as an average of $p$ consecutive knots in the knot vector, from which the first and last knot are exluded. This ensures that all entries of the Jacobian matrix are constant. In order to enable application of exact B-spline integration rules (derived in [17]), the (open) knot vectors describing the parameterization of the B-spline patch are always uniform having from 2 to 5 non-zero consecutive knot spans. Since the efficiency of these rules is dependent on the actual number of knot spans, the same number of knot spans is used for each spatial dimension. Only a limited set of degrees of B-spline basis functions has been employed. The selected values, namely degrees 2, 3, and 4,[2] cover the range of degree of mostly adopted B-spline basis functions and are sufficient to illustrate the influence of the degree on the computational performance of the investigated quadrature schemes. All the problems have been run in one-, two-, and three-dimensional space. The particular jobs are identified as $x$D:$y$-$z$ where $x \in \{1, 2, 3\}$ stands for the spatial dimension of the problem, $y \in \{2, 3, 4\}$ denotes the degree of B-spline basis functions (common for all spatial dimensions), and $z \in \{2, 3, 4, 5\}$ indicates the number of uniform knot spans (also common for all spatial dimensions). For example, 2D:3-4 denotes two-dimensional analysis of degree 3×3 with 4×4 non-zero uniform knot spans (see Figure 1).

[2] The case of degree 1 is not interesting, because then the IGA is identical with FEA and the Gaussian quadrature is optimal in such a case.

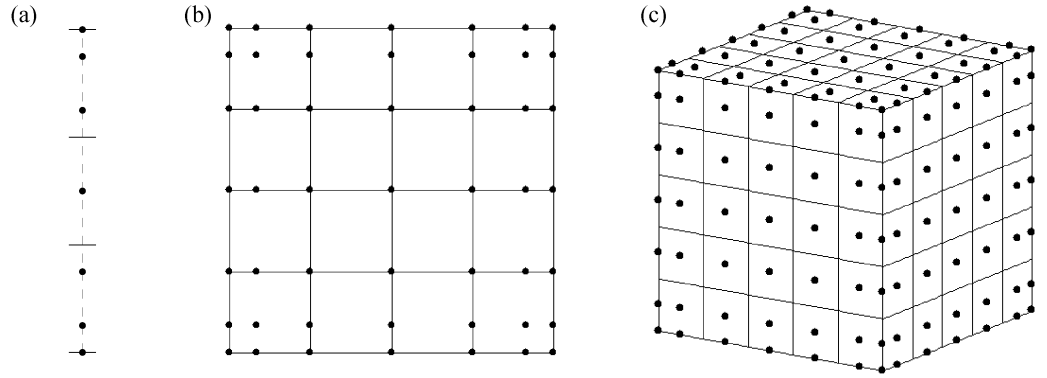


*Fig.1: Examples of geometry of investigated B-spline patches: (a) 1D:4-3, (b) 2D:3-4, (c) 3D:2-5; patch control points are indicated by dots, distinct knots by solid lines; dashed line in 1D is used just to outline the actual geometry*

The investigated quadrature schemes are the following:

- GSR – Gauss Standard Rule,
- GBE – Gauss rule on Bézier Elements,
- GPS – Gauss rule with basis functions Precomputed for all knot Spans,
- EBR – Exact B-spline Rule.

The GPS scheme, considered only to assess the slow-down of the GBE scheme due the application of the extraction operator, is similar to the GBE scheme in that the values of B-spline basis functions and their derivatives are precomputed. However, since the concept of Bézier extraction is not adopted in the GPS scheme, the precomputed values must be stored for all knot spans. Note that due to the tensor product structure of the Gauss rules, only the univariate B-spline functions and their derivatives are stored for individual spatial directions in the GPS as well as in the GBE scheme.

The performance of individual quadrature schemes has been assessed by measuring the time needed for the assembly of complete stiffness matrix (in the symmetric skyline format). In order to make the time measurable, the stiffness matrix has been assembled repeatedly, namely $10^6$ times for 1D problems, $10^4$ times for 2D problems, and $10^2$ times for 3D problems. Recalling that the Jacobian matrix is constant, the integrated terms of the stiffness matrix are univariate polynomials of order equal to $2\,p-2$ in 1D case and multivariate (but of tensor product structure) polynomials of order $2\,p$ (in each variable) in 2D and 3D case, where $p$ denotes the degree of B-spline basis functions. This allows to select the appropriate quadrature rule with minimum number of integration points which still leads to exact results. In the case of Gauss-based schemes, taking into account the basic property of Gauss-Legendre quadrature rules, which states that $n$-point quadrature schemes integrate exactly polynomials of order up to $2\,n-1$, the number of integration points used per span is equal to $p$ for 1D problems and to $p+1$ (in each direction) for 2D and 3D problems. When using the EBR scheme, the number of integration points is dependent not only on the order $q$ of integrated univariate polynomials but also on minimum continuity $k$ of these polynomials at inner knots (meaning that all the derivatives up to $k$ are continuous). The appropriate EBR quadrature scheme then must exactly integrate functions from space $\varphi_{q,k}$ of piecewise polynomials defined over the whole span of the knot vector (see [17] for details). It is not difficult to show that the relevant spaces in the context of the evaluation of the

stiffness matrix are $\varphi_{2p-2,p-2}$ for 1D case and $\varphi_{2p,p-2}$ for 2D and 3D case. Thus for the investigated degrees of B-spline basis functions 2, 3, and 4, the exact B-spline integrations rules for the quadrature in spaces $\varphi_{2,0}$, $\varphi_{4,0}$, $\varphi_{4,1}$, $\varphi_{6,1}$, $\varphi_{6,2}$, and $\varphi_{8,2}$ are needed. While integration schemes for the first three spaces are available in [17], schemes for the remaining spaces had to be computed and are summarized in Appendix A.

A typical pseudo-code for the assembly of the stiffness matrix (on an abstract level) using the Gaussian quadrature is presented in Table 1. Each knot span is associated with an integration rule (see papers [20, 21]) which stores individual integration points, position of which are defined within that single knot span. Note that at all integration points within the same integration rule, the same basis function attain non-zero value. Such an implementation can be easily adopted for the exact B-spline quadrature schemes. It is just enough to localize the individual integration points, distribution of which is defined over several consecutive knot spans, into individual knot spans and create corresponding integration rules.

```
compute_stiffness_matrix {
   initialize K_global;
   loop over all B-spline patches (Bp) {
      loop over all integration rules (ir) of Bp {
         initialize K_local;
         loop over all integration points (ip) of ir {
            B = compute B_matrix at(ip);
            D = compute D_matrix at(ip);
            J = compute Jacobian at(ip);
            K_local += B^T.D.B.J;
         }
         assemble K_local to K_global;
      }
   }
   return K_global;
}
```

*Tab.1: Pseudo-code for the evaluation of the stiffness matrix*

The results of individual analyses are summarized separately in Tables 2, 3 and 4 for spatial dimension 1, 2, and 3, respectively. Note that the elapsed time[3] does not account neither for precomputing the values of B-spline basis functions and their derivatives (for schemes GBE and GPS) nor for the evaluation of the extraction operator (in the GBE scheme) which is also precomputed and stored. Except the timing, also some additional quantities are provided to complete the information:

- Ctrl pnts – total number of control points describing the B-spline patch,
- G* tip – total number of integration points for GSR/GBE/GPS schemes,
- G* ip/d – number of integration points per number of spatial dimensions on the whole patch for GSR/GBE/GPS schemes,
- EBR $\varphi_{q,k}$ – space of piecewise polynomials corresponding to the EBR scheme,
- EBR tip – total number of integration points for the EBR scheme,
- EBR ip/d – number of integration points per number of spatial dimensions on the whole patch for the EBR scheme.

[3] The simulations have been performed using OOFEM [18] software package on a Dell Precision notebook equipped with Intel dual core 2.53 MHz processor and 4 GB of memory running under Ubuntu 9.04.

| Job id | Ctrl pnts | G* | | GSR | GBE | GPS | EBR | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | tip | ip/d | time | time | time | $\varphi_{q,k}$ | tip | ip/d | time |
| 1D:2-2 | 4 | 4 | 4 | 6.7 | 4.6 | 4.0 | $\varphi_{2,0}$ | 3 | 3 | 5.7 |
| 1D:2-3 | 5 | 6 | 6 | 10.5 | 7.0 | 5.9 | | 4 | 4 | 7.2 |
| 1D:2-4 | 6 | 8 | 8 | 13.4 | 9.3 | 7.6 | | 5 | 5 | 9.8 |
| 1D:2-5 | 7 | 10 | 10 | 17.2 | 11.6 | 9.6 | | 6 | 6 | 11.0 |
| 1D:3-2 | 5 | 6 | 6 | 12.2 | 7.3 | 6.1 | $\varphi_{4,1}$ | 4 | 4 | 8.5 |
| 1D:3-3 | 6 | 9 | 9 | 18.5 | 10.8 | 8.9 | | 6 | 6 | 13.0 |
| 1D:3-4 | 7 | 12 | 12 | 24.8 | 14.0 | 11.9 | | 7 | 7 | 15.3 |
| 1D:3-5 | 8 | 15 | 15 | 31.2 | 18.1 | 15.0 | | 9 | 9 | 19.8 |
| 1D:4-2 | 6 | 8 | 8 | 19.1 | 10.9 | 8.6 | $\varphi_{6,2}$ | 6 | 6 | 14.5 |
| 1D:4-3 | 7 | 12 | 12 | 29.3 | 15.6 | 13.1 | | 8 | 8 | 20.5 |
| 1D:4-4 | 8 | 16 | 16 | 38.8 | 23.3 | 17.3 | | 10 | 10 | 25.8 |
| 1D:4-5 | 9 | 20 | 20 | 50.1 | 29.1 | 22.3 | | 12 | 12 | 31.3 |

*Tab.2: Summary of 1D jobs assembling $10^6$ times stiffness matrix. Timing is in seconds*

| Job id | Ctrl pnts | G* | | GSR | GBE | GPS | EBR | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | tip | ip/d | time | time | time | $\varphi_{q,k}$ | tip | ip/d | time |
| 2D:2-2 | 16 | 36 | 6 | 2.2 | 1.7 | 1.6 | $\varphi_{4,0}$ | 25 | 5 | 1.5 |
| 2D:2-3 | 25 | 81 | 9 | 4.8 | 3.8 | 3.6 | | 49 | 7 | 3.0 |
| 2D:2-4 | 36 | 144 | 12 | 8.5 | 7.2 | 6.5 | | 81 | 9 | 5.0 |
| 2D:2-5 | 49 | 225 | 15 | 13.6 | 10.7 | 10.1 | | 121 | 11 | 8.0 |
| 2D:3-2 | 25 | 64 | 8 | 6.9 | 6.0 | 5.6 | $\varphi_{6,1}$ | 36 | 6 | 4.4 |
| 2D:3-3 | 36 | 144 | 12 | 15.4 | 13.3 | 13.1 | | 81 | 9 | 9.5 |
| 2D:3-4 | 49 | 256 | 16 | 27.9 | 23.5 | 22.6 | | 121 | 11 | 14.3 |
| 2D:3-5 | 64 | 400 | 20 | 43.3 | 37.7 | 35.3 | | 196 | 14 | 23.4 |
| 2D:4-2 | 36 | 100 | 10 | 19.4 | 17.7 | 16.7 | $\varphi_{8,2}$ | 64 | 8 | 13.9 |
| 2D:4-3 | 49 | 225 | 15 | 44.1 | 39.5 | 38.2 | | 121 | 11 | 26.2 |
| 2D:4-4 | 64 | 400 | 20 | 78.5 | 70.2 | 68.0 | | 196 | 14 | 42.9 |
| 2D:4-5 | 81 | 625 | 25 | 121.1 | 109.7 | 105.8 | | 289 | 17 | 63.8 |

*Tab.3: Summary of 2D jobs assembling $10^4$ times stiffness matrix; timing is in seconds*

| Job id | Ctrl pnts | G* | | GSR | GBE | GPS | EBR | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | tip | ip/d | time | time | time | $\varphi_{q,k}$ | tip | ip/d | time |
| 3D:2-2 | 64 | 216 | 6 | 1.3 | 1.3 | 1.3 | $\varphi_{4,0}$ | 125 | 5 | 0.8 |
| 3D:2-3 | 125 | 729 | 9 | 4.5 | 4.3 | 4.3 | | 343 | 7 | 2.3 |
| 3D:2-4 | 216 | 1728 | 12 | 10.6 | 10.2 | 10.1 | | 729 | 9 | 5.0 |
| 3D:2-5 | 343 | 3375 | 15 | 20.8 | 20.1 | 20.0 | | 1331 | 11 | 9.1 |
| 3D:3-2 | 125 | 512 | 8 | 13.7 | 13.6 | 13.6 | $\varphi_{6,1}$ | 216 | 6 | 6.5 |
| 3D:3-3 | 216 | 1728 | 12 | 46.4 | 46.1 | 46.0 | | 729 | 9 | 22.1 |
| 3D:3-4 | 343 | 4096 | 16 | 110.2 | 109.5 | 108.8 | | 1331 | 11 | 40.9 |
| 3D:3-5 | 512 | 8000 | 20 | 217.3 | 215.2 | 214.5 | | 2744 | 14 | 85.3 |
| 3D:4-2 | 216 | 1000 | 10 | 93.0 | 92.9 | 92.6 | $\varphi_{8,2}$ | 512 | 8 | 53.7 |
| 3D:4-3 | 343 | 3375 | 15 | 315.6 | 314.2 | 314.0 | | 1331 | 11 | 140.5 |
| 3D:4-4 | 512 | 8000 | 20 | 747.1 | 744.6 | 745.8 | | 2744 | 14 | 291.1 |
| 3D:4-5 | 729 | 15625 | 25 | 1456.8 | 1458.5 | 1457.3 | | 4913 | 17 | 523.0 |

*Tab.4: Summary of 3D jobs assembling $10^2$ times stiffness matrix; timing is in seconds*

The inspection of 1D results in Table 2 reveals that the times needed by GSR and EBR schemes are, not surprisingly, approximately in the ratio of the total number of integration points. It also shows that the application of the GBE scheme leads to significant speedup which increases with the growing degree of B-spline basis functions (as the demands for their evaluation are growing). Although the GBE scheme outperforms the EBR scheme for all considered cases, the trend in the results indicates that for a large enough number of knot spans, the EBR scheme is going to start to dominate because the savings due to decreasing the total number of integration points grow faster than the savings of the GBE scheme with respect to the GSR scheme. It is also apparent that the critical number of knot spans will be decreasing with the increasing degree because the costs of the GBE scheme due to the application of the extraction operator are growing with the increasing degree as the size of the extraction operator grows as well. In the case of results of 2D analyses (see Table 3), the situation changes quite considerably. The profit from precomputing the values of basis functions and their derivatives in GBE and GPS schemes is much less pronounced, which is caused by the fact, that the saving is realized (due to the tensor product structure of the patch) only at limited number of integration points corresponding to 1D integration schemes in either direction. On the other hand, the advantage of the EBR scheme due to the smaller number of integration points is accentuated because the total number of integration points is growing with the square (of the number of integration points used in the single direction). This implies that the EBR scheme is the best from all the schemes and its performance (compared to the GSR scheme) improves again not only with the increasing number of knot spans but also with the increasing degree. It is also worth to note that the cost of Bézier extraction for the 2D case is, compared to 1D, diminishing. This is, however, not caused by the improved efficiency of the Bézier extraction in 2D but by the decrease of its participation in the overall computational demands, which are enlarged by two facts. Firstly, the evaluation of the derivatives of basis functions with respect to Cartesian coordinates is more complex and secondly, the size of matrices $B$ and $K_{\text{local}}$ handled in the stiffness matrix assembly algorithm (see Table 1) is growing rapidly with the increasing degree. Assuming that the degree is the same in both spatial directions (which is the considered case), the number of basis functions which are non-zero at a particular integration point is growing with square of the degree. This effect becomes critical in 3D (see Table 4) where there is virtually no difference between individual Gauss based schemes. In this case the size of matrix $K_{\text{local}}$ grows with the cube of the degree of the B-spline basis functions. Thus the costs related to the computation of the product $B^{\text{T}}DBJ$ are dominating and the overall assembly time, for a given degree, is more or less linearly dependent on the total number of integration points. Since the total number of integration points for the EBR scheme is much smaller than the number used by the Gauss based schemes, the EBR scheme in 3D is clearly superior for all degrees and number of knot spans. It is interesting to see, however, that despite the fact that the costs of the numerical quadrature in 2D and 3D are driven predominantly by the evaluation of the product $B^{\text{T}}DBJ$ (not of its components), the times for GSR and EBR schemes are only approximately in the ratio of the total number of integration points. A detailed inspection of the profiling information has uncovered that the time consumed by the function evaluating the product $B^{\text{T}}DBJ$ per integration point is noticeably smaller for the GSR scheme. This could be attributed to the effect of caching $K_{\text{local}}$ because the number of processed integration points per integration rule is generally higher for the GSR scheme compared to the EBR scheme.

## 3. Conclusions

In this paper, a study of computational efficiency of several numerical quadrature schemes available for the IGA has been performed. The performance of the schemes has been assessed on the assembly of the stiffness matrix on such a geometrical arrangement of a B-spline patch that the minimum number of integration points leading to exact results could have been safely determined. The investigation has revealed that the main source of the computational costs of the numerical quadrature is dependent on the spatial dimension qualitatively as well as quantitatively. While in 1D the prevailing costs are related to the expensive evaluation of basis functions and their derivatives and are increasing with the degree and consequently with the complexity of the B-spline basis functions, in 2D and 3D, the dominating costs are associated with the assembly of the contributions to the stiffness matrix at individual integration points, number of which as well as the size of the contributions is also growing with the degree. This implies that in 1D, the algorithms which profit from the precomputed values of the basis functions and their derivatives (such as the GBE scheme) are the most competitive (at least up to a certain crititial number of knot spans). In 2D and 3D, on the other hand, since the critical factor is the total number of integration points, the quadrature rules that benefit from taking into account the continuity between the knot spans (such as the EBR scheme) are always (and in 3D significantly) the better ones.

In the current implementation of two- and three-dimensional GSR, GBE as well as EBR schemes, there is still some space for savings. For example, the evaluation of particular components of the stiffness matrix could be accelerated if they are computed on the level of integration rule rather than on the level of integration point, because the locally precomputed univariate quantities (on the level of integration rule) can be repeatedly reused (due to the tensor product structure) for all integration points within the same integration rule. The preliminary results reveal, however, that this effect is of only a little significance in 2D and completely negligible in 3D.

An important issue is related to the fact that, in reality, the integrated functions are only rarely polynomials. More commonly, the integrated terms are of rational character as the consequence of non-constant Jacobian (does not matter whether due to the location of control points[4] of a B-spline geometry or because of using non-uniform weights in a NURBS geometry). In such a case, the common practice to select the quadrature rule under the assumption that the Jacobian is constant may lead to significant error. This aspect can be demonstrated on a simple example of one-dimensional truss modelled by a quadratic B-spline patch with two uniform knots with control points located at $x_1 = 0$, $x_2 = 0.2$, $x_3 = 0.6$, and $x_4 = 1$. The entry $[2, 3]$ of stiffness matrix computed analytically yields $-0.112943611$. When applying 2-point (per span) Gaussian integration, the obtained value is $-0.09615$, which corresponds to approximately 15 % error. The 2-span exact B-spline rule with three integration points for integration in space $\varphi_{2,0}$ gives value $-0.25$ which is wrong by far more than 100 %. Although this is rather pathological case, effect of which is reduced by the fact that the above entry corresponds approximately to 3.4 % of the maximum stiffness matrix entry, it should not be generally neglected. Thus the over-integration when using the Gaussian quadrature rules may play also a positive role. Moreover, taking into account the fact that the derivation of exact B-spline quadrature rule for non-uniform knot

[4] Note that while in the FEA the Jacobian (more precisely, its variation) could be reasonably controlled by the quality of the finite element mesh, in the IGA, the analyst does not have usually such a flexibility as he/she is stuck with the geometry.

spans is computationally prohibitive, especially if large number of spans and high degree of basis functions is considered, and that refining the knot vector to (at least piecewise) uniform knot vector leads to the increase of both the number of control points (and thus also problem unknowns) and the number of integration points, the use of the standard Gaussian quadrature per non-zero knot span still remains reasonable alternative (in a general case). This, however, implies that the question of numerical quadrature in the IGA remains open and that there is a strong need to further search for efficient quadrature rules.

**Acknowledgments**

This work was supported by the Grant Agency of the Czech Republic – Project No. 103/09/2009. Its financial assistance is gratefully acknowledged.

**Appendix A.**

In this Appendix, the quadrature schemes for the exact integration in $\varphi_{6,1}$, $\varphi_{6,2}$, and $\varphi_{8,2}$ on the interval $[0,1]$ with 2, 3, 4, and 5 uniform knot spans are provided. Coordinates and weights of quadrature points, summarized in Tables 5, 6, and 7, have been computed by the numerical procedure outlined in [17] which is based on the solution of a system of non-linear equations stating the exactness of the quadrature rule in the form

$$\int_{\Omega} B_j^p(\xi)\,\mathrm{d}\xi = \sum_{i=1}^{nip} H\, w_i\, B_j^p(\xi_i)\ , \quad j = 1,\ldots,ndof\ , \tag{1}$$

where $B_j^p(\xi)$ denotes the $j$-th univariate B-spline basis function of degree $p$, $ndof$ represents the overall number of basis functions (equal to the total number of control points on

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.046212737218262 | 0.115024181444685 |
| 2 | 0.213797850599990 | 0.203072613437640 |
| 3 | 0.413962200649013 | 0.181903205117487 |
| 4 | 0.586037799350987 | 0.181903205117487 |
| 5 | 0.786202149400010 | 0.203072613437640 |
| 6 | 0.953787262781738 | 0.115024181444685 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.018520173765736 | 0.046097561282814 |
| 2 | 0.085687841697670 | 0.081402101437263 |
| 3 | 0.166020807756211 | 0.073397453079548 |
| 4 | 0.237178093583745 | 0.077039783118035 |
| 5 | 0.322462519002690 | 0.086822232209338 |
| 6 | 0.397484374599436 | 0.062648276280257 |
| 7 | 0.462203552699062 | 0.072592592592556 |
| 8 | 0.537796447300938 | 0.072592592592556 |
| 9 | 0.602515625400564 | 0.062648276280257 |
| 10 | 0.677537480997310 | 0.086822232209338 |
| 11 | 0.762821906416255 | 0.077039783118035 |
| 12 | 0.833979192243789 | 0.073397453079548 |
| 13 | 0.914312158302330 | 0.081402101437263 |
| 14 | 0.981479826234264 | 0.046097561282814 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.030642376608000 | 0.076266745374326 |
| 2 | 0.141734846651573 | 0.134567907647719 |
| 3 | 0.273971127286385 | 0.118618440802605 |
| 4 | 0.379779139633981 | 0.106063492775587 |
| 5 | 0.500000000000000 | 0.128966826799148 |
| 6 | 0.620220860366019 | 0.106063492775587 |
| 7 | 0.726028872713615 | 0.118618440802605 |
| 8 | 0.858265153348427 | 0.134567907647719 |
| 9 | 0.969357623392000 | 0.076266745374326 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.023151919684118 | 0.057626217478808 |
| 2 | 0.107117994402021 | 0.101761040444035 |
| 3 | 0.207547338857538 | 0.091778791186793 |
| 4 | 0.296610452114193 | 0.096512828661717 |
| 5 | 0.403475006137249 | 0.108804883034682 |
| 6 | 0.500000000000000 | 0.087032478387552 |
| 7 | 0.596524993862751 | 0.108804883034682 |
| 8 | 0.703389547885807 | 0.096512828661717 |
| 9 | 0.792452661142462 | 0.091778791186793 |
| 10 | 0.892882005597979 | 0.101761040444035 |
| 11 | 0.976848080315882 | 0.057626217478808 |

*Tab.5: Coordinates and weights of quadrature points for exact quadrature in $\varphi_{6,1}$ on the interval $[0,1]$ with 2 (top left), 3 (top right), 4 (bottom right), and 5 (bottom left) uniform knot spans*

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.046212737218262 | 0.115024181444685 |
| 2 | 0.213797850599990 | 0.203072613437639 |
| 3 | 0.413962200649013 | 0.181903205117487 |
| 4 | 0.586037799350987 | 0.181903205117487 |
| 5 | 0.786202149400010 | 0.203072613437639 |
| 6 | 0.953787262781738 | 0.115024181444685 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.019796220904769 | 0.049291039978820 |
| 2 | 0.091787579286588 | 0.087594474874824 |
| 3 | 0.181541216020813 | 0.089204170945075 |
| 4 | 0.274797052289712 | 0.098993104267671 |
| 5 | 0.371943037927442 | 0.090194507546738 |
| 6 | 0.456885245389865 | 0.084722702386682 |
| 7 | 0.543114754610135 | 0.084722702386682 |
| 8 | 0.628056962072558 | 0.090194507546738 |
| 9 | 0.725202947710288 | 0.098993104267671 |
| 10 | 0.818458783979187 | 0.089204170945075 |
| 11 | 0.908212420713412 | 0.087594474874824 |
| 12 | 0.980203779095231 | 0.049291039978820 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.032509212332349 | 0.080935503234357 |
| 2 | 0.150617986025149 | 0.143497669038710 |
| 3 | 0.294914900084714 | 0.137136269815659 |
| 4 | 0.429193875888619 | 0.138430557911085 |
| 5 | 0.570806124111381 | 0.138430557911085 |
| 6 | 0.705085099915286 | 0.137136269815659 |
| 7 | 0.849382013974851 | 0.143497669038710 |
| 8 | 0.967490787667651 | 0.080935503234357 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.024674350061536 | 0.061435702093662 |
| 2 | 0.114388066516845 | 0.109125580973635 |
| 3 | 0.225696248333798 | 0.109440065519943 |
| 4 | 0.338034400421348 | 0.117618543966031 |
| 5 | 0.451622850188181 | 0.102380107446540 |
| 6 | 0.548377149811819 | 0.102380107446540 |
| 7 | 0.661965599578652 | 0.117618543966031 |
| 8 | 0.774303751666202 | 0.109440065519943 |
| 9 | 0.885611933483155 | 0.109125580973635 |
| 10 | 0.975325649938464 | 0.061435702093662 |

*Tab.6: Coordinates and weights of quadrature points for exact quadrature in $\varphi_{6,2}$ on the interval $[0, 1]$ with 2 (top left), 3 (top right), 4 (bottom right), and 5 (bottom left) uniform knot spans*

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.029317365558605 | 0.073785388617378 |
| 2 | 0.142157704774954 | 0.144519421377153 |
| 3 | 0.299856825427078 | 0.160564975301295 |
| 4 | 0.443811666373756 | 0.121130214704163 |
| 5 | 0.556188333626244 | 0.121130214704163 |
| 6 | 0.700143174572922 | 0.160564975301295 |
| 7 | 0.857842295225046 | 0.144519421377153 |
| 8 | 0.970682634441395 | 0.073785388617378 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.012217141892880 | 0.030750255302139 |
| 2 | 0.059263644659815 | 0.060288691569683 |
| 3 | 0.125182489234921 | 0.067371141771344 |
| 4 | 0.187554035144786 | 0.057210267660312 |
| 5 | 0.247257333632995 | 0.066005863127736 |
| 6 | 0.318054522422214 | 0.071759255045056 |
| 7 | 0.383299962490317 | 0.057049551398649 |
| 8 | 0.437792437671208 | 0.056976406190641 |
| 9 | 0.500000000000000 | 0.065177135868859 |
| 10 | 0.562207562328792 | 0.056976406190641 |
| 11 | 0.616700037509683 | 0.057049551398649 |
| 12 | 0.681945477577786 | 0.071759255045056 |
| 13 | 0.752742666367005 | 0.066005863127736 |
| 14 | 0.812445964855214 | 0.057210267660312 |
| 15 | 0.874817510765079 | 0.067371141771344 |
| 16 | 0.940736355340185 | 0.060288691569683 |
| 17 | 0.987782858107121 | 0.030750255302139 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.020216037277298 | 0.050882211074448 |
| 2 | 0.098054432048896 | 0.099731887928612 |
| 3 | 0.207036969428665 | 0.111258635867919 |
| 4 | 0.308710240572963 | 0.089942766919391 |
| 5 | 0.396857836748817 | 0.094098649790251 |
| 6 | 0.500000000000000 | 0.108171696838738 |
| 7 | 0.603142163251183 | 0.094098649790251 |
| 8 | 0.691289759427037 | 0.089942766919391 |
| 9 | 0.792963030571335 | 0.111258635867919 |
| 10 | 0.901945567951104 | 0.099731887928612 |
| 11 | 0.979783962722702 | 0.050882211074448 |

| # | Coordinate | Weight |
|---|---|---|
| 1 | 0.015253923946387 | 0.038393631636945 |
| 2 | 0.073993328835077 | 0.075270862300191 |
| 3 | 0.156285641994598 | 0.084089673438105 |
| 4 | 0.233943313351680 | 0.070747659625745 |
| 5 | 0.306871944085494 | 0.079958531122863 |
| 6 | 0.392416628402800 | 0.086471694873232 |
| 7 | 0.469822629830853 | 0.065067947002908 |
| 8 | 0.530177370169147 | 0.065067947002908 |
| 9 | 0.607583371597200 | 0.086471694873232 |
| 10 | 0.693128055914506 | 0.079958531122863 |
| 11 | 0.766056686648320 | 0.070747659625745 |
| 12 | 0.843714358005402 | 0.084089673438105 |
| 13 | 0.926006671164923 | 0.075270862300191 |
| 14 | 0.984746076053613 | 0.038393631636945 |

*Tab.7: Coordinates and weights of quadrature points for exact quadrature in $\varphi_{8,2}$ on the interval $[0, 1]$ with 2 (top left), 3 (top right), 4 (bottom right), and 5 (bottom left) uniform knot spans*

1D patch), $H$ stands for the size of the parametric space $\Omega$ ($H = 1$ was used as the unit interval $[0, 1]$ is considered as $\Omega$), $nip$ is the number of integrations points and $\xi_i$ and $w_i$ are their unknown coordinates and weights. For even $ndof$, the minimum number of integration points allowing to fulfil Eqs (1) is given by $nip = ndof/2$. For odd $ndof$, the minimum required number of integration points turns out to be $nip = (ndof + 1)/2$, which yields one less equations than is the number of unknowns. To ensure the unique solution in such a case, the system of equations (1) is supplemented by a symmetry condition representing the fact that, for the investigated case of uniform knot vector, the $(nip+1)/2$-th integration point is in the middle of $\Omega$. The left-hand side of Eqs (1), corresponding to the exact values of integrals of individual B-spline basis functions, can be evaluated using, for example, the standard Gauss quadrature rule using $(p + 1)/2$ and $(p + 2)/2$ Gauss integration points on each non-zero knot span for odd and even $p$, respectively. The system of equations (1), extended by the symmetry condition for odd $ndof$, has been solved using the MATLAB *fsolve* routine with tolerance $10^{-12}$.

## References


[1] Hughes T.J.R., Cottrell J.A., Bazilevs Y.: Isogeometric Analysis: CAD, Finite Elements, NURBS, Exact Geometry and Mesh Refinement, Computer Methods in Applied Mechanics and Engineering, 194 (2005), pp. 4135–4195

[2] Cottrell J.A., Hughes T.J.R., Bazilevs Y.: Isogeometric Analysis: Toward Integration of CAD and FEA, John Wiley & Sons 2009

[3] Rogers D.F.: An Introduction to NURBS: With Historical Perspective, Morgan Kaufmann 2000

[4] Piegl L., Tiller W.: The NURBS Book, Springer-Verlag 1997

[5] Sederberg T.W., Zheng J., Bakenov A., Nasri A.: T-splines and T-NURCCs, ACM Transactions on Graphics (SIGGRAPH 2003), 22(3) (2003), pp. 477–484

[6] Bazilevs Y., Calo V.M., Cottrell J.A., Evans J.A., Hughes T.J.R., Lipton S., Scott M.A., Sederberg T.W.: Isogeometric Analysis Using T-splines, Computer Methods in Applied Mechanics and Engineering, 199(5–8) (2010) pp. 229–263

[7] Sederberg T.W., Gardon D., Finnigan G., North N., Zheng J., Lyche T.: T-spline Simplification and Local Refinement, ACM Transactions on Graphics (SIGGRAPH 2004), 23(3) (2004), pp. 276–283

[8] Dörfel M.R., Jüttler B., Simeon B.: Adaptive Isogeometric Analysis by Local h-refinement with T-splines, Computer Methods in Applied Mechanics and Engineering, 199(5-8) (2010), pp. 264–275

[9] Li X., Zheng J., Sederberg T.W., Hughes T.J.R., Scott M.A.: On Linear Independence of T-splines Blending Functions, Computer Aided Geometric Design 29(1) (2012), pp. 63–76

[10] Vuong A.-V., Giannelli C., Jüttler B., Simeon B.: A Hierarchical Approach to Adaptive Local Refinement in Isogeometric Analysis, Computer Methods in Applied Mechanics and Engineering, 200(49–52) (2011), pp. 3554–3567

[11] Cottrell J.A., Reali A., Bazilevs Y., Hughes T.J.R.: Isogeometric Analysis of Structural Vibrations, Computer Methods in Applied Mechanics and Engineering, 195 (2006), pp. 5257–5296

[12] Cottrell J.A., Hughes T.J.R., Reali A.: Studies of Refinement and Continuity in Isogeometric Structural Analysis, Computer Methods in Applied Mechanics and Engineering, 196 (2007), pp. 4160–4183

[13] Auricchio F., da Veiga L.B., Buffa A., Lovadina C., Reali A., Sangalli G.: A Fully "Locking-free" Isogeometric Approach for Plane Linear Elasticity Problems: A Stream Function Formulation, Computer Methods in Applied Mechanics and Engineering, 197 (2007), pp. 160–172

[14] Lipton S., Evans J.A., Bazilevs Y., Elguedj T., Hughes T.J.R.: Robustness of Isogeometric Structural Discretizations under Severe Mesh Distortion, Computer Methods in Applied Mechanics and Engineering, 199 (2010), pp. 357–373

[15] Borden M.J., Scott M.A., Evans J.A., Hughes T.J.R.: Isogeometric Finite Element Data Structures Based on Bézier Extraction of NURBS, International Journal for Numerical Methods in Engineering, 87(1–5) (2011), pp. 15–47
[16] Scott M.A., Borden M.J., Verhoosel C.V., Sederberg T.W., Hughes T.J.R.: Isogeometric Finite Element Data Structures Based on Bézier Extraction of T-splines, International Journal for Numerical Methods in Engineering, 88(2) (2011), pp. 126–156
[17] Hughes T.J.R., Reali A., Sangalli G.: Efficient Quadrature for NURBS-based Isogeometric Analysis, Computer Methods in Applied Mechanics and Engineering, 199(5–8) (2010), pp. 301–313
[18] Patzák B.: OOFEM project home page, http://www.oofem.org, 2012
[19] Greville T.N.E.: Spline Functions, Interpolation, and Numerical Quadrature, Mathematical Methods for Digital Computers, II, (1967), Ralston A., Wilf H.S. (eds.), Wiley, New York, pp. 156–168
[20] Rypl D., Patzák B.: From the Finite Element Analysis to the Isogeometric Analysis in an Object Oriented Computing Environment, Advances in Engineering Software, 44(1) (2012), pp. 116–125
[21] Rypl D., Patzák B.: Object Oriented Implementation of the T-spline Based Isogeometric Analysis, Advances in Engineering Software 50 (2012), pp. 137–149

*Received in editor's office*: April 13, 2012
*Approved for publishing*: June 26, 2012