GeoPDEs tutorial: solutions

This document contains the solutions of the tutorial, except the **optional** exercises. It is assumed that the user has basic knowledge of Matlab or Octave, and that GeoPDEs (base and elasticity packages) has been already installed. Sometimes I will refer to the article, that can be downloaded from the webpage. For further details about how the functions work, check their help in Matlab.

1. **Understanding h-, p- and k-refinement.**

   (a) Use the NURBS toolbox to define a NURBS parameterization of one quarter of a ring, with internal radius $R_i = 1$ and external radius $R_o = 2$. Use degree elevation to set the degree to 2 in both directions. Plot the geometry and the control points with nrbctrlplot. (Hint: you can use either nrbline and nrbrevolve, or nrbcirc and nrbruled. For degree elevation use nrbdegelev).

   Solution: There are, at least, two alternative ways to create the geometry. The first one defines the line from $(1, 0)$ to $(2, 0)$, and then revolves it along the z-axis:

```
crv = nrbline ([1 0], [2 0]);
srf = nrbrevolve (crv, [0 0 0], [0 0 1], pi/2);
```

   The second one defines two circular arcs (for radius $R_i = 1$ and $R_o = 2$) with NURBS parametrizations, and then creates the ruled surface between them.

```
crv1 = nrbcirc (1, [0 0 0], 0, pi/2);
crv2 = nrbcirc (2, [0 0 0], 0, pi/2);
srf = nrbruled (crv1, crv2);
```

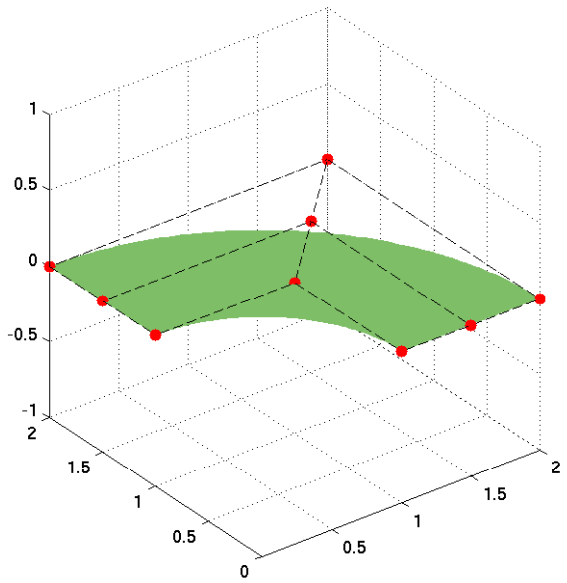
   The information for the NURBS parametrization (order, knots, control points, weights) is stored in the structure srf. For details about these fields, see the Appendix in the article. In particular, the order of the geometry is

```
>> srf.order
ans =
     3     2
```

   Since the order is equal to the degree plus one, we have to elevate the degree by one only in the second parametric direction. The command to do so is

```
srf = nrbdegelev (srf, [0 1]);
```

   Finally, to plot the geometry with its control points, type in Matlab nrbctrlplot(srf). The result should look like this:



(b) Check the knot vector of your geometry. Insert new knots to get a 10×10 mesh. Plot the mesh with nrbkntplot. (Hint: use kntrefine to compute the new knots, and nrbkntins for knot insertion).

Solution: Typing srf in Matlab we can get all the information about the NURBS structure:

```
srf =
      form: 'B-NURBS'
       dim: 4
    number: [3 3]
     coefs: [4x3x3 double]
     knots: {[0 0 0 1 1 1]  [0 0 0 1 1 1]}
     order: [3 3]
```

The surface does not have any internal knots (they are all equal to zero or one), so the mesh consists of one single element.

To refine the mesh we have to perform knot insertion, using the function nrbkntins, and giving as an argument the knots that we want to insert in each parametric direction. The simplest way to compute these knots is to use the Matlab command **linspace**. Since we want a 10×10 mesh we have to add 9 internal knots (11 points minus the two extrema):

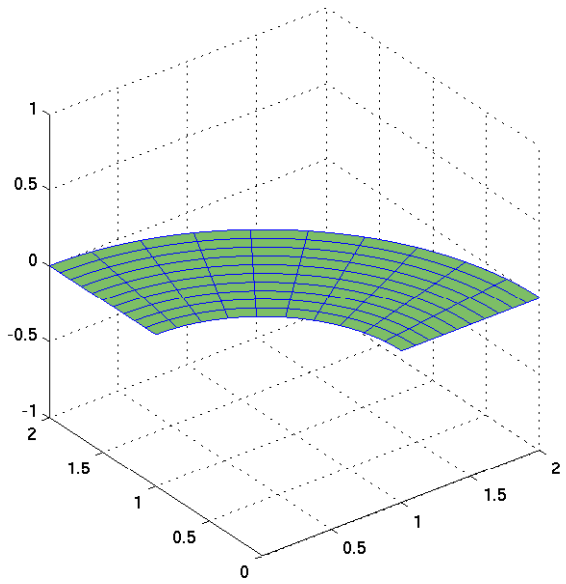
```
new_knots = linspace (0, 1, 11);
new_knots = new_knots(2:end-1);
srf_refined = nrbkntins (srf, {new_knots, new_knots});
```

An alternative way to do this is to use the function kntrefine, which is often used in the examples of GeoPDEs. The advantage of this functions is that it refines uniformly any given knot vector, also the ones with internal knots. The input arguments are the initial knot vector, the number of knots to be inserted on each interval, the degree and the continuity, these last three for each parametric direction. In this case the first two output arguments can be ignored (check the

help of the function for details), the third one is a Matlab cell-array with the new knots to be inserted, already for both parametric directions.

```
[~,~,new_knots] = kntrefine (srf.knots, [9 9], [2 2], [1 1]);
srf_refined = nrbkntins (srf, new_knots);
```

The refined mesh can be plotted with the command nrbkntplot(srf_refined). The figure should look like this:



(c) Perform h-refinement: starting from the geometry of point (a), modify the example ex_article_15lines to solve in a sequence of meshes with 1, 2, 4, 8, 16, 32 and 64 elements in each parametric direction. Check the convergence rate plotting the error vs. the degrees of freedom. Do the same with C^0 continuity, inserting the knots twice, and compare the two curves of convergence.

Solution: The exercise is solved in the file exercise_1c.m. Read the comments in the file to understand what is done. The main ideas are also given here.

To refine the mesh we can use the same commands we have used in point (b), with a different number of subdivisions. Solving the problem for each mesh is simply to repeat the same instructions of the example ex_article_15lines . To plot the convergence rate we must save, for each refinement step, the number of degrees of freedom and the computed error. The curve of convergence can be plotted with the **loglog** command.

Finally, the same can be done for C^0 continuity by simply changing the last argument in the function kntrefine, to insert the knots twice. Check the values in the cell-array nkots in both cases, to see the difference.

(d) Perform p-refinement: starting from the geometry of point (b), modify the example ex_article_15lines to solve with degree from 2 to 8. In this case, the continuity will be C^1 in all the cases. Plot the error vs. the degrees of freedom. (Hint: use nrbdegelev for degree elevation).

Solution: The exercise is solved in the file exercise_1d.m. Read the comments in the file to understand what is done. The modifications are similar to those of the previous exercise, but instead of knot insertion we must use degree elevation.

(e) Perform k-refinement: starting from the geometry of point (a), first raise the degree (from 2 to 8), and then insert new knots to get a 10×10 mesh. Solve the same problem as before. In this case, for degree p, the continuity will be C^{p-1}. Compare the curve of the error with the one in (d).

Solution: The exercise is solved in the file exercise_1e.m. Read the comments in the file to understand what is done. The modifications are similar to those of the two previous exercises. In this example it is important to do the refinement starting always from the parametrization of exercise 1 (a), otherwise we would not have the maximum continuity. For every degree, we must perform degree elevation before knot insertion.

(f) **Optional:** Using a coarse mesh, you can compare the knot vectors, the basis functions and the **msh** and **space** structures for C^0 and C^{p-1} continuity. (Hint: use **msh_precompute** and **sp_precompute**).
Solution: The commands **msh** = **msh_precompute**(**msh**) and **space** = **sp_precompute**(**space**,**msh**) will compute all the files listed in the tables of the article. The idea of this exercise is to understand how the three possibilities of refinement work. You can check the knot vectors (**nurbs**.knots) to see how many repetitions are necessary in each case. With the connectivity array (**space.connectivity**) you can understand what is the support of each single function. It is also interesting to compare the number of nonzero entries of the matrix, with the command **spy**(mat).

Note: In most of the examples of GeoPDEs the refinement is done automatically, giving the degree, the continuity, and the number of subdivisions as data. You can see how the data is given in ex_laplace_iso_ring , for instance, and how the refinement is performed in solve_laplace_2d_iso .

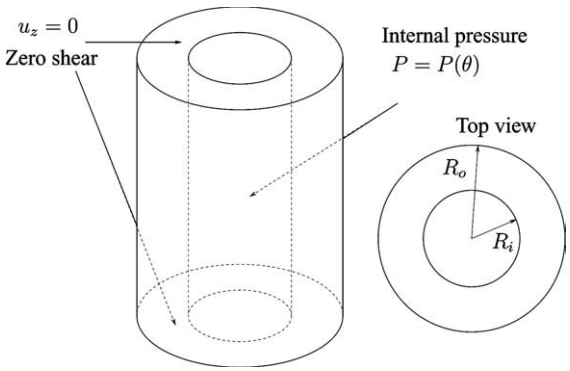


Figure 1: Thick cylinder subjected to internal pressure.

2. **Solid circular cylinder subjected to internal pressure.**

   This is an example from the first paper on IGA [2]. The geometry is a hollow cylinder, subjected to a constant pressure on its internal side. The setting of the problem is given in Fig. 1. The same example is already solved with a plane strain formulation in the files ex_plane_strain_ring and solve_plane_strain_2d. We are going to solve the example in 3D in one eighth of the cylinder, using symmetry conditions.

   (a) Extrude the geometry from exercise 1 (a), with the function nrbextrude. Set the length of the cylinder equal to 5. Save the geometry in a .txt file, with the nrbexport command.

   Solution: The first two lines construct the geometry of exercise 1 (a). Then we extrude it along the vector (0,0,5). Finally, we save the geometry in the file 'cylinder.txt'. Notice that, to be used in GeoPDEs, this file must have the .txt extension.

```
crv = nrbline ([1 0], [2 0]);
srf = nrbrevolve (crv, [0 0 0], [0 0 1], pi/2);
vol = nrbextrude (srf, [0 0 5]);
nrbexport (vol, 'cylinder.txt');
```

   (b) With the help of ex_plane_strain_ring and ex_lin_elast_horseshoe, write an example to solve the linear elasticity problem in the cylinder. Set the degree to 3 and C^2 continuity in the three directions, and refine to have a $5 \times 5 \times 5$ mesh. Apply a constant pressure on the inner boundary, symmetry conditions on the three symmetry boundaries, and homogeneous Neumann conditions on the other boundaries. (Hint: check the numbering of the boundaries, as they may be different from the 2D case).

   Solution: This example is helpful to understand how to apply the boundary conditions. The solution is in the file exercise_2b. The way to set the Lamé parameters, the degree, regularity and the number of elements is explained in the comments of the file, and it is similar to all the examples in GeoPDEs. Here we explain how to apply the boundary conditions.

   We are modelling just one eighth of the cylinder, so our physical domain is $\Omega = \{(x, y, z) : z \in (0, 5),\ 1 < x^2 + y^2 < 4,\ x, y > 0\}$. We will impose symmetry conditions on the boundaries $\{(x, y, z) \in \Omega : x = 0,\ \text{or } y = 0,\ \text{or } z = 0\}$. We will also set the internal pressure on the side $\{(x, y, z) \in \Omega : x^2 + y^2 = 1\}$. On the rest of the boundary we will set homogeneous Neumann conditions.

To impose the boundary conditions we must find which is the number of each face of the boundary. In GeoPDEs there is a numbering for the six faces of the unit cube $\widehat{\Omega} = (0,1)^3$. The numbering of these six faces is:

- Face 1: $\{(\widehat{x},\widehat{y},\widehat{z}) \in \widehat{\Omega} : \widehat{x} = 0\}$. Face 2: $\{(\widehat{x},\widehat{y},\widehat{z}) \in \widehat{\Omega} : \widehat{x} = 1\}$.
- Face 3: $\{(\widehat{x},\widehat{y},\widehat{z}) \in \widehat{\Omega} : \widehat{y} = 0\}$. Face 4: $\{(\widehat{x},\widehat{y},\widehat{z}) \in \widehat{\Omega} : \widehat{y} = 1\}$.
- Face 5: $\{(\widehat{x},\widehat{y},\widehat{z}) \in \widehat{\Omega} : \widehat{z} = 0\}$. Face 6: $\{(\widehat{x},\widehat{y},\widehat{z}) \in \widehat{\Omega} : \widehat{z} = 1\}$.

The parameterization $\mathbf{F} : \widehat{\Omega} \longrightarrow \Omega$, which maps the parametric domain to the physical one, also maps the boundary sides from $\partial\widehat{\Omega}$ to $\partial\Omega$. Hence, the boundary sides in $\partial\Omega$ inherit the numbering of the boundary sides in $\partial\widehat{\Omega}$. If you are not sure about how $\mathbf{F}$ was defined, one way to find the numbering is to use the command nrbextract. Given a NURBS geometry, this function extracts the information for the boundary sides, and saves it in an array of structures that can be used in the NURBS toolbox. For instance, the lines

```
bndry = nrbextract (vol);
nrbkntplot (bndry(3));
```

save in the variable bndry the information for the six boundaries of our domain, and then plot the third of these boundaries. Plotting the six boundaries one by one, we find that in our example the numbering is

- Face 1: $\{(x,y,z) \in \Omega : y = 0\}$. Face 2: $\{(x,y,z) \in \Omega : x = 0\}$.
- Face 3: $\{(x,y,z) \in \Omega : x^2 + y^2 = 1\}$. Face 4: $\{(x,y,z) \in \Omega : x^2 + y^2 = 4\}$.
- Face 5: $\{(x,y,z) \in \Omega : z = 0\}$. Face 6: $\{(x,y,z) \in \Omega : z = 5\}$.

Hence, we apply the symmetry condition on the sides 1, 2 and 5, the internal pressure condition on side 3, and the Neumann homogeneous condition on sides 4 and 6 (see exercise_2b).

(c) Using ParaView, compare the horizontal displacements and stress for the 2D and the 3D problems. For the 3D problem, check also that the displacement in the z-direction is null.

Solution: An example on how to export to ParaView is given in exercise_2b. Run this example, and also the 2D example ex_plane_strain_ring to get the ParaView files for both cases. To load the solution in ParaView open the file .vts, and then click on the **Apply** button. To learn the basics about ParaView you can read the second chapter of the ParaView tutorial: http://www.paraview.org/Wiki/The_ParaView_Tutorial.

(d) Solve the same problem removing the symmetry condition on the bottom boundary, and setting a zero z-displacement condition on the top and bottom boundaries. The code as it is only allows to impose Dirichlet conditions for all the components at the same time, but you can impose the condition by modifying the function solve_linear_elasticity . Use the fields in **space.boundary** to impose the condition on the right dofs.

Solution: The exercise is solved in the files exercise_2d and solve_linear_elasticity_3d_drchlt_comp . The file exercise_2d is similar to the one in exercise 2 (b), but we have changed the boundary conditions in the sides 5 and 6, and defined a new boundary condition to set a zero z-displacement. Since this condition was not implemented in GeoPDEs, we have also modified the file solve_linear_elasticity_3d to impose it. The way to do it is to identify the degrees of freedom in the third component, with the field **comp_dofs**, and set the value of the solution for those degrees of freedom to zero.

(e) **Optional:** Write a new function, similar to sp_l2_error and sp_h1_error, to compute the error in L^2-norm for the stress. The exact solution for $\nu = 0$ in cylindrical coordinates is

$$u_r = \frac{PR_i^2}{E(R_o^2 - R_i^2)} \left(r + \frac{R_o^2}{r} \right), \tag{1}$$

$$\sigma_{rr} = \frac{PR_i^2}{R_o^2 - R_i^2} - \frac{PR_i^2 R_o^2}{r^2(R_o^2 - R_i^2)}, \qquad \sigma_{\theta\theta} = \frac{PR_i^2}{R_o^2 - R_i^2} + \frac{PR_i^2 R_o^2}{r^2(R_o^2 - R_i^2)}. \tag{2}$$

Solution: The optional exercises are not solved in this version.

3. **Vibrations of a clamped thin circular plate.**

This is an example taken from [1]. We study the vibrations of a clamped, thin circular plate, modeled as a three-dimensional solid. The difference with the example in the article is that we will use the parameterization of the circle as the one in Fig. 2.

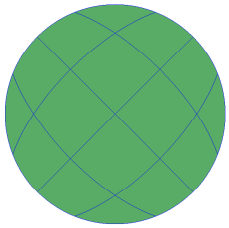


Figure 2: Top view of the parameterization of the thin circular plate.

The problem, in variational formulation, consists of finding the eigenpairs $(\omega_n, \mathbf{u_n}) \in \mathbb{R} \times (H_0^1(\Omega))^3$, such that

$$\int_\Omega (2\mu\varepsilon(\mathbf{u_n}) : \varepsilon(\mathbf{v}) + \lambda \text{div}\mathbf{u_n}\ \text{div}\mathbf{v}) = \omega_n^2 \int_\Omega \rho \mathbf{u_n} \cdot \mathbf{v}, \quad \forall \mathbf{v} \in (H_0^1(\Omega))^3, \tag{3}$$

where $\varepsilon(\mathbf{u})$ is the symmetric part of $\nabla\mathbf{u}$, ρ is the mass density, and λ, μ are the Lamé parameters. The problem can also be written in the form

$$\mathbf{K}\mathbf{u_n} = \omega_n^2 \mathbf{M}\mathbf{u_n}, \tag{4}$$

where $\mathbf{K}$ and $\mathbf{M}$ are the stiffness and mass matrices, respectively.

(a) Define the 2D geometry with the control points and weights given in Table 1, and use a linear transformation to make it of radius 2 m. Extrude the geometry to create the 3D plate, with a thickness of 0.02 m. (Hint: use nrbmak and nrbtform to define the geometry. Notice that the NURBS toolbox works with weighted control points, defined as $B_{i,j}^w = B_{i,j} w_{i,j}$).

Solution: we start by creating a 2D circle. We use in both directions an open knot vector for degree $p = 2$. The control points and weights are those given in Table 1, and are stored in the variable coefs. For a surface, the size of this variable is $(4, nu, nv)$, where nu and nv are the number of univariate functions in each parametric direction. In our case, $nu = nv = 3$.

```
knots = {[0 0 0 1 1 1], [0 0 0 1 1 1]};
coefs = zeros (4, 3, 3);
coefs(:,1,1) = [1, 0, 0, 1];
coefs(:,2,1) = [1/sqrt(2), -1/sqrt(2), 0, 1/sqrt(2)];
coefs(:,3,1) = [0, -1, 0, 1];
coefs(:,1,2) = [1/sqrt(2), 1/sqrt(2), 0, 1/sqrt(2)];
coefs(:,2,2) = [0, 0, 0, sqrt(2)-1];
coefs(:,3,2) = [-1/sqrt(2), -1/sqrt(2), 0, 1/sqrt(2)];
coefs(:,1,3) = [0, 1, 0, 1];
coefs(:,2,3) = [-1/sqrt(2), 1/sqrt(2), 0, 1/sqrt(2)];
coefs(:,3,3) = [-1, 0, 0, 1];
```

Once that we have set the knots, the control points and the weights, we create the geometry with nrbmak.

```
srf = nrbmak (coefs, knots);
```

Then, we apply a scaling to obtain the circle of radius 2. See the help of nrbtform and vecscale for details.

```
srf = nrbtform (srf, vecscale ([2 2 0]));
```

Finally, we extrude the geometry to obtain the thin plate.

```
vol = nrbextrude (srf, [0 0 0.02]);
```

The geometry can be saved in a .txt file, as in exercise 2.

(b) Create a function, similar to solve_linear_elasticity_3d , to solve problem (3), computing the first six eigenvalues and their corresponding eigenvectors. The function should also return the **geometry**, **msh**, **space** structures (objects). (Hint: use the command eigs to solve the generalized eigenvalue problem for sparse matrices.)

Solution: the solution is in the file solve_linear_elasticity_3d_eig . Read the comments in that file to understand how it was created.

(c) Solve the problem in a $6 \times 6 \times 1$ mesh, with degree 2. Set $E = 30 \cdot 10^6\ KN/m^2$, $\nu = 0.2$ and $\rho = 2.320\ KNs/m^4$. Compare the eigenvalues with the results in [1]. (Hint: plot the eigenvectors to be sure that you are comparing the right eigenvalues).

Solution: the solution is in the file exercise_3c. The geometry is that of point (a).

Since the plate is clamped, we impose homogeneous Dirichlet boundary conditions on the lateral sides (numbers 1 to 4), and homogeneous Neumann conditions on the bottom and top sides (numbers 5 and 6). The numbering of the sides can be checked using the command nrbextract(vol), as in exercise 2 (b).

The Lamé parameters are defined as in the previous examples. Notice that we must also define a parameter for the mass density, that appears in the mass matrix.

To solve the problem we call the function that we created in point (b). The solution of the eigenvalue problem gives us the values ω_n^2, so we must take its square root to compare with the values in [1].

The first six computed natural frequencies are

1.3202e+02, 4.7615e+02, 4.7615e+02, 7.2229e+02, 1.5565e+03, 1.5623e+03.

To compare with the values in [1], we first plot the modes of vibration (the eigenfunctions) in ParaView. It can be seen that the modes that appear in the article correspond to the first, the second and the fifth computed modes (see Figure 3).

The computed values are still very far from the exact solution, as can be seen in the following table. However this seems to be normal, since the values computed in [1] for degree 2 in a coarse mesh are also far away from the exact solution.

	ω_{01} [rad/s]	ω_{11} [rad/s]	ω_{02} [rad/s]
Computed	132.02	476.15	1556.5
Exact	53.863	112.670	210.597

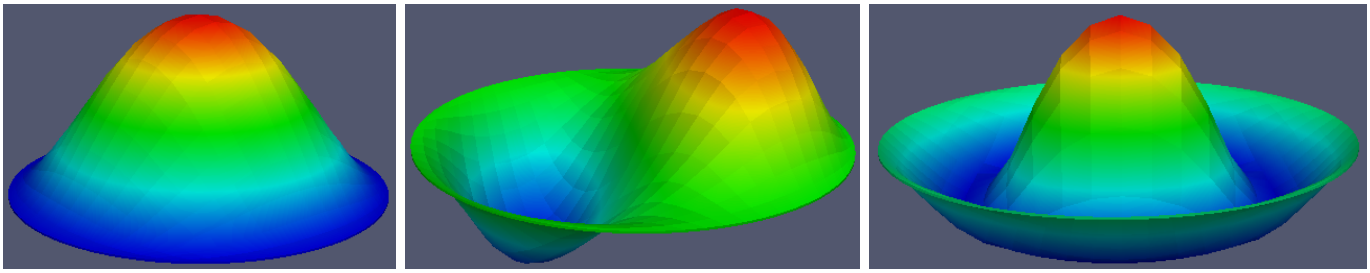


Figure 3: First, second and fifth computed eigenmodes, for degree 2 in a $6 \times 6 \times 1$ mesh.

(d) Raise the degree (up to 5) in the first two directions, and compare your results with the ones in [1]. Do the same using a coarser mesh.

Solution: the solution is in the file exercise_3d. The file is very similar to that of point (c), with the difference that we have inserted a loop to solve with degrees from 2 to 5.

The values of the computed frequencies are listed in the following table. Notice that for degree 3 and above, the computed eigenvalues that correspond to the ones in [1] are the first, the second and the sixth ones.

Degree	ω_{01} [rad/s]	ω_{11} [rad/s]	ω_{02} [rad/s]
2	132.02	476.15	1556.5
3	59.375	147.18	425.82
4	54.762	119.58	261.23
5	54.239	114.17	225.45
Exact	53.863	112.670	210.597

(e) **Optional:** solve the same problem with the parameterization of [1], which you can easily construct by revolution. Notice that in this case there are repeated control points, and it is necessary to identify the dofs associated to these control points as one single dof.

Solution: The optional exercises are not solved in this version.

i	$B^w_{i,1}$	$B^w_{i,2}$	$B^w_{i,3}$	$w_{i,1}$	$w_{i,2}$	$w_{i,3}$
1	$(1,\ 0)$	$(1/\sqrt{2},\ 1/\sqrt{2})$	$(0,\ 1)$	1	$1/\sqrt{2}$	1
2	$(1/\sqrt{2},\ -1/\sqrt{2})$	$(0,\ 0)$	$(-1/\sqrt{2},\ 1/\sqrt{2})$	$1/\sqrt{2}$	$\sqrt{2}-1$	$1/\sqrt{2}$
3	$(0,\ -1)$	$(-1/\sqrt{2},\ -1/\sqrt{2})$	$(-1,\ 0)$	1	$1/\sqrt{2}$	1

Table 1: Control points and weights for the circle of radius 1.

References

[1] J. A. Cottrell, A. Reali, Y. Bazilevs, and T. J. R. Hughes. Isogeometric analysis of structural vibrations. *Comput. Methods Appl. Mech. Engrg.*, 195(41-43):5257–5296, 2006.

[2] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comput. Methods Appl. Mech. Engrg.*, 194(39-41):4135–4195, 2005.