HAL open science

Shock capturing computations with stabilized Powell-Sabin elements

Giorgio Giorgiani, Hervé Guillard, Boniface Nkonga

To cite this version:

Giorgio Giorgiani, Hervé Guillard, Boniface Nkonga. Shock capturing computations with stabilized Powell-Sabin elements. ECCOMAS Congress 2016 VII European Congress on Computational Methods in Applied Sciences and Engineering, Jun 2016, Crete, Greece. pp.16. hal-01377909

HAL Id: hal-01377909

https://inria.hal.science/hal-01377909v1

Submitted on 7 Oct 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ECCOMAS Congress 2016
VII European Congress on Computational Methods in Applied Sciences and Engineering
M. Papadrakakis, V. Papadopoulos, G. Stefanou, V. Plevris (eds.)
Crete Island, Greece, 5–10 June 2016

SHOCK CAPTURING COMPUTATIONS WITH STABILIZED POWELL-SABIN ELEMENTS

Giorgio Giorgiani[1], Hervé Guillard[2] and Boniface Nkonga[2]

[1]Aix-Marseille Université
38 rue Joliot-Curie, La Jete, 13451 Marseille
e-mail: giorgio.giorgiani@univ-amu.fr

[2] INRIA Sophia Antipolis
2004 Route des Lucioles, 06902 Valbonne
e-mail: {herve.guillard,boniface.nkonga}@inria.fr

Keywords: Shock capturing, high-order, C1 approximations, splines, CFD, compressible flow.

Abstract. *In the last recent years, thanks to the increasing power of the computational machines, the interest in more and more accurate numerical schemes is growing. Methods based on high-order approximations are nowadays the common trend in the computational research community, in particular for CFD applications.*

This work is focused on Powell-Sabin (PS) finite elements, a finite element method (FEM) based on PS splines. PS splines are piecewise quadratic polynomials with a global C1 continuity, defined on conforming triangulations. Despite its attractive characteristics, so far this scheme hasn't had the attention it deserves. PS splines are adapted to unstructured meshes and, contrary to classical tensor product B-splines, they are particularly suited for local refinement, a crucial aspect in the analysis of highly convective and anisotropic equations. The additional global smoothness of the C1 space has a beneficial stabilization effect in the treatment of advection-dominated equations and leads to a better capturing of thin layers. Finally, unlike most of other typology of high-order finite elements, the numerical unknowns in PS elements are located in the vertices of the triangulation, leading to an easy treatment of the parallel aspects.

Some geometrical issues related to PS elements are discussed here, in particular, the generation of the control triangles and the imposition of the boundary conditions. The PS FEM method is used to solve the compressible Euler equation in supersonic regime. A classical shock-capturing technique is used to reduce the oscillation around the discontinuity, while a variational multiscale formulation is used to introduce numerical diffusion in the streamwise direction. Some typical numerical examples are used to evaluate the performance of the PS discretization.

1 INTRODUCTION

Numerical methods with high degrees of regularity are of interest in many physical applications, such as magnetohydrodynamics, shells analysis and vibrations. In particular, finite-element methods based on spline shape functions have raised a particular interest in the recent years. On one hand, C^1 spline methods allow to discretize higher-order derivatives (that is, derivatives of order superior to the second), and on the other hand, they provide accurate geometrical representation of the computational domain, see for example [1, 3] in the context of Isogeometric Analysis. Moreover, it has also been shown recently that the additional global smoothness of the spline interpolant introduces stability to the numerical solution for highly convective equations, see [1] and [7], and also for turbulence computations, see [2]. Hence, finite element methods based on spline shape functions can be advantageous also in the context of fluid-dynamic problems.

Nowadays, splines are extensively used in the graphical design industry to create smooth surfaces. The success of splines is basically due to the fact that they have very attractive and unique characteristics: they have a compact representation, they are able to represent curves and surfaces with an arbitrary level of regularity, simply increasing the polynomial degree of the basis, and they are efficient in terms of locality, that is, they can be modified locally by moving a control point, without perturbing the rest of the curve. Spline curves are usually represented as a linear combination of basis functions, called *B-splines*. The basis functions of a given order p are defined with a recursive relation starting with piecewise constant basis functions (that is, $p = 0$),

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

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

where the points ξ_i are the knots of the spline, $i = 1, 2, \ldots, n + p + 1$, and n is the number of basis functions. Hence, the spline curve expressed as a combination of B-splines is

$$S(\xi) = \sum_{i=1}^{n} B_{i,p} \mathbf{P}_i, \tag{1}$$

where $\mathbf{P}_i$ are the control points of the spline.

Starting from the one-dimensional case (1), B-splines can be extended to higher dimensions through a tensor product representation. However, tensor product B-splines are restricted to structured rectangular meshes. The refinement procedure with tensor B-splines relies on the insertion of knots, leading hence to a global modification of the domain discretization. Thus, no local refinement is possible. Various solutions are proposed in literature to solve this limitation, for example, the introduction of *T-splines* in [3].

This work is focused on another approach, based on the definition of bivariate splines on irregular triangulations, that is, Powell-Sabin (PS) splines [5, 6, 7, 8]. PS splines are piecewise quadratic polynomial with C^1 continuity, defined on a unstructured triangulation of the domain. The major advantage with respect to tensor product B-splines is that PS splines allow a straightforward adaptive refinement of the mesh, which is a key ingredient in the simulation of anisotropic equations. Another advantage of PS elements is related to the fact that the unknowns are concentrated on the nodes of the triangulation. This means that there are no unknowns on

the element faces or in the interior of the elements. This is not the case for other C^1 interpolation techniques, such as, for example, the Clough-Tocher elements. The consequence is that the stencil is the same for each node of the mesh, which produces a linear system matrix with a regular block shape that facilitates the implementation aspect, enhances the parallel performance and improves the efficiency of the linear system solution.

In this work, a PS finite element method is presented for the solution of the two dimensional (2D) compressible Euler equation in supersonic regime. A SUPG stabilization technique is used to introduce numerical dissipation in the streamwise direction only, and an isotropic diffusion terms is added to avoid oscillation around the discontinuities.

The outline of the paper is as follows. Section 2 introduces the PS splines and their representation. Section 3 deals with the geometrical and mathematical tools to define the PS elements and in particular the generation of the shape functions. A detailed discussion is also given on the imposition of the boundary conditions. Finally, in Section 4 two classical numerical examples are presented.

2 POWELL-SABIN SPLINES

In this section is introduced the framework and the notation to define spline functions on a triangulation of a polygonal domain. Starting from the definition of bivariate polynomials on a triangle, the goal is to obtain an interpolant of a generic function with C^1 continuity. Considering a single triangle Ω_k of vertices $\mathbf{V}_i = (x_i, y_i)$, with $i = 1, 2, 3$, any bivariate polynomial $p(x, y)$ of degree ≤ 2 in the space $\Pi_2 = \{\sum_{i=0}^{2} \sum_{j=0}^{2-i} a_{i,j} x^i y^j, a_{i,j} \in \mathbb{R}\}$ can be written in the Bernstein-Bézier representation

$$p(x, y) := p(\xi) = \sum_{\substack{i+j+k=2 \\ i,j,k \geq 0}} b_{i,j,k} B^2_{i,j,k}(\xi(x, y)), \tag{2}$$

where $\xi = (\xi_1, \xi_2, \xi_3)$ are the barycentric coordinates of a point $(x, y) \in \mathbb{R}^2$ and $B^2_{i,j,k}(\xi)$ are the Bernstein polynomials of degree 2 in Ω_k, that is

$$B^2_{i,j,k} = \frac{2!}{i!j!k!} \xi_1^i \xi_2^j \xi_3^k. \tag{3}$$

The set of Bernstein polynomials $B^2_{i,j,k}(\xi)$ is a basis for the space of polynomials Π_2, see [4], hence the six coefficients $b_{i,j,k}$ uniquely define the second order polynomial $p(x, y)$ and they are called the Bézier ordinates with respect to the triangle Ω_k[1].

Let $\Omega \subset \mathbb{R}^2$ be a polygonal domain with boundary $\partial\Omega$. Let $\mathcal{T}$ be a conforming triangulation of Ω with vertices $\mathbf{V}_l$, $l = 1, \ldots, N_v$ and elements Ω_e, $e = 1, \ldots, N_e$. Defining $\mathcal{S}^2_1$ as the linear space of piecewise quadratic polynomials on $\mathcal{T}$, the following interpolation problem is considered: given any set of triples (f_l, f_{xl}, f_{yl}), $l = 1, \ldots, Nv$, find $s(x, y) \in \mathcal{S}^2_1$ such that,

$$s(\mathbf{V}_l) = f_l, \quad \frac{\partial s}{\partial x}(\mathbf{V}_l) = f_{xl}, \quad \frac{\partial s}{\partial y}(\mathbf{V}_l) = f_{yl}, \quad l = 1, \ldots, Nv. \tag{4}$$

It is clear that such a problem has no solution in general: in fact, problem (4) requires the imposition of nine parameters to define the second order polynomial on each triangle, while only six coefficients are available, see equation (2).

[1] Note that (2) can be expanded as $p(\xi) = b_{2,0,0}\xi_1^2 + b_{0,2,0}\xi_2^2 + b_{0,0,2}\xi_3^2 + 2b_{1,1,0}\xi_1\xi_2 + 2b_{1,0,1}\xi_1\xi_3 + 2b_{0,1,1}\xi_2\xi_3$.

Thus, to obtain a solution to the interpolation problem (4), one possibility is to subdivide each triangle in sub-triangles and to define the interpolant in the refined triangulation. One of the solutions proposed by Powell and Sabin in [5] is based on the subdivision of each triangle in $\mathcal{T}$ into six smaller triangles (*PS6-split*). Hence, the conditions in (4) are imposed only on the vertices of the original triangulation, while in the other added nodes only C_1 continuity of the interpolating function is imposed. More details can be found in [5].

The so called Powell-Sabin refinement of $\mathcal{T}$ is denoted with $\mathcal{T}^*$ and is obtained dividing each triangle in $\mathcal{T}$ in six sub-triangles. The procedure to define $\mathcal{T}^*$ is described in [7] and reads as follows.

- Select a split point $\mathbf{C}_k$ inside each triangle Ω_k and connect it to the three vertices of Ω_k.
- For each pair of triangles Ω_p and Ω_q, with a common edge, connect the two points $\mathbf{C}_p$ and $\mathbf{C}_q$. If Ω_p is a boundary triangle, connect also $\mathbf{C}_p$ to an arbitrary point on the boundary edge.

The PS refinement defines also a set of points, called *PS points*, associated to each vertex. The PS points associated to the vertex $\mathbf{V}_l$ are defined as the midpoints of all the edges of the PS refinement containing $\mathbf{V}_l$, plus the point $\mathbf{V}_l$ itself. The PS points are fundamental for the definition of the shape functions as explained in the next section.

Having defined the PS refinement $\mathcal{T}^*$, the linear space of piecewise quadratic polynomials with C^1 continuity can be denoted as $S_2^1(\mathcal{T}^*)$. Each element $S_2^1(\mathcal{T}^*)$ is uniquely defined by its values and derivatives at the vertices of $\mathcal{T}$, thus the functional space $S_2^1(\mathcal{T}^*)$ has dimension $3N_v$.

3 POWELL-SABIN FINITE ELEMENTS

In this section PS splines are introduced in a finite-element framework to solve partial differential equations, defining shape functions belonging to the space $S_2^1(\mathcal{T}^*)$. Hence, it is necessary to express each element $s(x,y) \in S_2^1(\mathcal{T}^*)$ as a linear combination of PS spline basis functions, that is

$$s(x,y) = \sum_{l=1}^{N_v} \sum_{r=1}^{3} c_{l,r} B_l^{(r)}(x,y),$$

where the functions $B_l^{(r)}$ are called *Powell-Sabin B-splines* and $c_{l,r}$ are the coefficients of the representation.

Each B-spline $B_l^{(r)}(x,y)$ can be seen as the solution of the interpolation problem (4) with all $(f_i, f_{xi}, f_{yi}) = 0$ except for $(f_l, f_{xl}, f_{yl}) = (\alpha, \beta, \gamma) \neq 0$. The quantity (α, β, γ) is called a *triple* and represents the value of the function and the derivatives with respect the Cartesian axes in a given vertex. Then, it is easy to see that each basis $B_l^{(r)}$, $r = 1, 2, 3$, vanishes outside the molecule M_l of vertex $\mathbf{V}_l$ defined as the union of all triangles Ω_e containing $\mathbf{V}_l$. This guarantees the compactness of the support for PS B-splines.

In principle, it is possible to define the PS B-splines simply choosing, for each vertex $\mathbf{V}_l \in \mathcal{T}$, three linearly independent vectors $(\alpha_{l,r}, \beta_{l,r}, \gamma_{l,r})$, $r = 1, 2, 3$, and solving the interpolation problem (4). This procedure generates three linearly independent basis functions for each vertex. However, in a finite element context, it is interesting to work with *normalized*

B-splines, that is, basis functions that form a partition of unity on Ω, i.e.

$$B_l^{(r)}(x,y) \geq 0 \quad \forall x,y \in \Omega,$$
$$\sum_{l=1}^{N_v}\sum_{r=1}^{3} B_l^{(r)}(x,y) = 1 \quad \forall x,y \in \Omega,$$

which leads to the following constrains on the triples

$$\sum_{r=1}^{3} \alpha_{l,r} = 1, \quad \sum_{r=1}^{3} \beta_{l,r} = 0, \quad \sum_{r=1}^{3} \gamma_{l,r} = 0.$$

This property is also crucial to ensure the completeness of the basis.

The procedure proposed in [8] allows to obtain PS B-splines that form a partition of unity and with good characteristic from a numerical point of view. It is based on the fact that each set of three independent B-splines associated to a vertex $\mathbf{V}_l$ uniquely define a *control triangle* $T_l = (\mathbf{Q}_{l,1}, \mathbf{Q}_{l,2}, \mathbf{Q}_{l,3})$ for the PS spline, with vertices $\mathbf{Q}_{l,j} = (X_{l,j}, Y_{l,j})$, $j = 1,2,3$. The control triangle of a PS spline as a role similar to the control polygon for univariate splines in terms of control of the shape of the spline. In fact, in [8] it is shown that the control triangle and the coefficients $c_{l,j}$ define three control points $(X_{l,j}, Y_{l,j}, c_{l,j})$ defining a spatial plane that is always tangent to the spline surface at the vertex $\mathbf{V}_l$. Optimal shape functions from a numerical point of view are associated to a minimal area control triangle [8].

The points $\mathbf{Q}_{l,j}$ are related to the three triples and the vertex $\mathbf{V}_l$ via the linear system

$$\begin{bmatrix} \alpha_{l,1} & \alpha_{l,2} & \alpha_{l,3} \\ \beta_{l,1} & \beta_{l,2} & \beta_{l,3} \\ \gamma_{l,1} & \gamma_{l,2} & \gamma_{l,3} \end{bmatrix} \begin{bmatrix} X_{l,1} & Y_{l,1} & 1 \\ X_{l,2} & Y_{l,2} & 1 \\ X_{l,3} & Y_{l,3} & 1 \end{bmatrix} = \begin{bmatrix} x_l & y_l & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \tag{5}$$

The positivity of the B-splines associated to the vertex $\mathbf{V}_l$ is guaranteed if all the barycentric coordinates with respect the control triangle T_l of all the PS points associated to $\mathbf{V}_l$ are positive. This means geometrically that all the PS points associated to $\mathbf{V}_l$ are contained in the control triangle T_l, see Figure 1.

Hence, the procedure of computing the three shape functions for each vertex is:

- find a suitable control triangle for the vertex $\mathbf{V}_l$ that contains all the PS points associated to $\mathbf{V}_l$;
- compute the triple defining the three shape functions solving the linear system (5).

The control triangles for each vertex are constructed looking for the minimal area triangle that contains all the PS points. This lead to an optimization problem consisting in circumscribing a convex polygon with a minimal area triangle. An optimal algorithm is proposed by O'Rourke in [9] for solving such a problem, with a computational cost $\mathcal{O}(n)$, with n the number of PS points. This is the algorithm used here in the numerical tests. In Figure 1 is depicted an example of control triangle, with the relative PS control points, obtained with the O'Rourke algorithm.

In order to understand the role of the control triangle in the definition of the basis functions, a comparison of the condition number of the linear system matrix as been carried out for three structured computational meshes for a square domain $[0,1]\times[0,1]$. In the first case, the minimal area triangle have been used to compute the basis. In the second case, the minimal area triangle

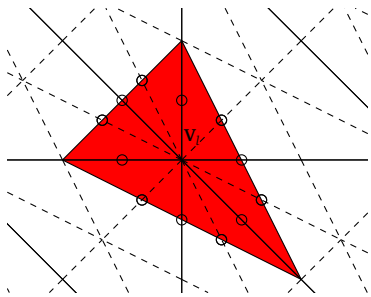


Figure 1: Powell-Sabin control triangle and relative Powell-Sabin points for the vertex $\mathbf{V}_l$.

Table 1: Condition number comparison for the stiffness matrix and the mass matrix obtained with optimal and non-optimal control triangles in a square domain: Ne is the number of triangular elements of the mesh, h is the element size.

		Cond. numb. of the stiffness matrix		Cond. numb. of the mass matrix	
Ne	h	optimal	non optimal	optimal	non optimal
200	70.71E-03	83	389	171	4466
800	35.36E-03	326	453	163	4466
3200	17.68E-03	1305	1585	154	4469

has been expanded moving (to a distance equal to five times the original one) each vertex along the line joining the vertex with the node to which the triangle belongs. Table 1 shows the results: it is possible to notice how increasing the control triangle area degrades the condition number of mass matrix and the stiffness matrix.

4 STABILIZED APPROXIMATION OF THE EULER EQUATIONS WITH POWELL-SABIN ELEMENTS

The 2D inviscid Euler equations of gas dynamics are considered,

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{u}) &= 0 \\ \frac{\partial \rho \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{u} \otimes \boldsymbol{u} + p\boldsymbol{\mathcal{I}}) &= \mathbf{0} \\ \frac{\partial \rho E}{\partial t} + \boldsymbol{\nabla}\cdot((\rho E + p)\boldsymbol{u}) &= 0 \end{aligned} \tag{6}$$

where ρ is the density, $\boldsymbol{u}$ is the velocity vector, E is the total energy, $\boldsymbol{\mathcal{I}}$ is the identity matrix and p is the pressure, which is defined by the equation of state $p = (\gamma_g - 1)\rho\epsilon$, $\gamma_g = 1.4$ is the ratio of specific heats and $\epsilon = E - 1/2\|\boldsymbol{u}\|^2$ is the internal energy.

Problem (6) is recast in a conservative form,

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{\mathcal{F}} = \mathbf{0}, \tag{7}$$

having defined the vector of conservative variables $\boldsymbol{U} = (\rho, \rho u, \rho v, \rho E)^T$ and the flux tensor $\boldsymbol{\mathcal{F}}$ with components, in Cartesian coordinates,

$$\boldsymbol{\mathcal{F}} = \left[\boldsymbol{F}_x \; \boldsymbol{F}_y\right], \quad \boldsymbol{F}_x = \left\{\begin{array}{c} \rho u \\ \rho u^2 + p \\ \rho u v \\ (\rho E + p)u \end{array}\right\}, \quad \boldsymbol{F}_y = \left\{\begin{array}{c} \rho v \\ \rho u v \\ \rho v^2 + p \\ (\rho E + p)v \end{array}\right\},$$

where u and v are the x and y components of the velocity vector.

A stabilized formulation based on PS splines is considered to solve problem (7). To simplify the presentation, in an abuse of notation, the same symbol is used for the numerical approximation, belonging to the finite dimensional space, and the exact solution $\boldsymbol{U}$. Hence, the stabilized finite element problem corresponding to (7) reads: find $\boldsymbol{U} \in [S_2^1(\mathcal{T}^*)]^4$ such that

$$\int_\Omega \boldsymbol{V} \cdot \frac{\partial \boldsymbol{U}}{\partial t}\, d\Omega - \int_\Omega \boldsymbol{\nabla V} : \boldsymbol{\mathcal{F}}\, d\Omega + \int_{\partial\Omega} \boldsymbol{\mathcal{F} V} \cdot \boldsymbol{n}\, d\Gamma + \mathcal{D}_{SUPG} + \mathcal{D}_{SC} = 0, \tag{8}$$

for all $\boldsymbol{V} \in [S_2^1(\mathcal{T}^*)]^4$. The terms $\mathcal{D}_{SUPG}$ and $\mathcal{D}_{SC}$ are the stabilization terms and they will be discussed later.

A fully implicit first-order time integration scheme is employed to approximate the time derivatives. The non-linear convective term is linearized using a Newton-Raphson procedure of the type

$$\int_\Omega \boldsymbol{\nabla V} : \boldsymbol{\mathcal{F}}^k\, d\Omega \approx \int_\Omega \boldsymbol{\nabla V} : \mathbb{A}^{k-1}\boldsymbol{U}^k\, d\Omega,$$

where k stands for the current N-R iteration and the third order tensor $\mathbb{A}$ is the Jacobian of the flux,

$$\mathbb{A} = \frac{\partial \boldsymbol{\mathcal{F}}}{\partial \boldsymbol{U}},$$

and can be expressed as a second order tensor for each Cartesian direction, that is

$$\frac{\partial \boldsymbol{F}_x}{\partial \boldsymbol{U}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{\gamma_g-1}{2}\|\boldsymbol{u}\|^2 - u^2 & (3-\gamma_g)u & (1-\gamma_g)v & \gamma_g - 1 \\ -uv & v & u & 0 \\ \frac{\gamma_g-1}{2}u\|\boldsymbol{u}\|^2 - uH & -u^2(\gamma_g-1)+H & (1-\gamma_g)uv & \gamma_g u \end{pmatrix},$$

$$\frac{\partial \boldsymbol{F}_y}{\partial \boldsymbol{U}} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ -uv & v & u & 0 \\ \frac{\gamma_g-1}{2}\|\boldsymbol{u}\|^2 - v^2 & (1-\gamma_g)u & (3-\gamma_g)v & \gamma_g - 1 \\ \frac{\gamma_g-1}{2}v\|\boldsymbol{u}\|^2 - vH & (1-\gamma_g)uv & -v^2(\gamma_g-1)+H & \gamma_g v \end{pmatrix},$$

having defined the entalpy, $H = E + p/\rho$.

4.1 Imposition of the boundary conditions

Solid walls and free-stream boundaries are considered for defining the boundary conditions of system (6). Walls are considered as slip boundary conditions, that is, the normal component of the flux is set to zero on a wall. This kind of condition does not entail particular problems, and it is weakly imposed by the boundary integral in (8). On the contrary, inlet and outlet boundary conditions are imposed setting different components of the unknown vector at the boundary nodes, depending on the subsonic/supersonic regime at the boundary, see [15]. The problem of imposing boundary values deserves particular attention in the framework of PS finite elements. Since these kind of conditions are typically set on straight boundaries, the interest is focused on imposing boundary values with PS elements on polygonal boundaries.

A careful choice of the control triangle associated to the boundary node $\mathbf{V}_l$ simplify the task of imposing boundary values. Two different cases must be considered: the boundary node belongs to two edges forming an angle different to π, or equal to π. From equation (5), the

following relations are derived

$$\alpha_{l,1}\mathbf{Q}_{l,1} + \alpha_{l,2}\mathbf{Q}_{l,2} + \alpha_{l,3}\mathbf{Q}_{l,3} = \mathbf{V}_l, \tag{9}$$

$$\beta_{l,1}\mathbf{Q}_{l,1} + \beta_{l,2}\mathbf{Q}_{l,2} + \beta_{l,3}\mathbf{Q}_{l,3} = \mathbf{e}_x, \tag{10}$$

$$\gamma_{l,1}\mathbf{Q}_{l,1} + \gamma_{l,2}\mathbf{Q}_{l,2} + \gamma_{l,3}\mathbf{Q}_{l,3} = \mathbf{e}_y, \tag{11}$$

where $\mathbf{e}_x$ and $\mathbf{e}_y$ are respectively the unity vector in the x and y directions. Equation (9) states that $(\alpha_{l,1}, \alpha_{l,2}, \alpha_{l,3})$ are the barycentric coordinates of the vertex $\mathbf{V}_l$ with respect to the control triangle, while (10) and (11) mean that $(\beta_{l,1}, \beta_{l,3}, \beta_{l,3})$ and $(\gamma_{l,1}, \gamma_{l,2}, \gamma_{l,3})$ are respectively the barycentric coordinates of the vectors $\mathbf{e}_x$ and $\mathbf{e}_y$ with respect to the control triangle.

In the case $\mathbf{V}_l$ forms an angle different from π, if the control triangle has two sides aligned with the two boundary edges as in Figure 2 (left), from equation (9) derives that only one shape function has value different from zero in $\mathbf{V}_l$. This allows to directly set the boundary value to the unknown associated to the non-zero shape function. Moreover, from (10) and (11) derives that the other two shape functions have zero derivative in the directions the two edges concurring in $\mathbf{V}_l$. In particular, using the notation depicted in Figure 2 (left) where triangle vertices are now called $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$, it results

$$\begin{aligned} \alpha_{l,B} = \alpha_{l,C} = 0, (\Rightarrow \alpha_{l,A} = 1), \\ \begin{Bmatrix} \beta_{l,B} \\ \gamma_{l,B} \end{Bmatrix} \cdot \boldsymbol{t} = 0, \\ \begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} \cdot \boldsymbol{r} = 0. \end{aligned}$$

Denoting with U the component of $\boldsymbol{U}$ to be imposed, and $\hat{u}$ the $\mathcal{L}^2$ projection of the prescribed values on the space of univariate quadratic splines on $\partial\Omega$, the resulting equations relative to the unknowns of the vertex $\mathbf{V}_l$ are

$$\begin{aligned} U_{l,A} = \hat{u}(\mathbf{V}_l), \\ U_{l,A} \begin{Bmatrix} \beta_{l,A} \\ \gamma_{l,A} \end{Bmatrix} \cdot \boldsymbol{r} + U_{l,B} \begin{Bmatrix} \beta_{l,B} \\ \gamma_{l,B} \end{Bmatrix} \cdot \boldsymbol{r} = \boldsymbol{\nabla}\hat{u}(\mathbf{V}_l) \cdot \boldsymbol{r}, \\ U_{l,A} \begin{Bmatrix} \beta_{l,A} \\ \gamma_{l,A} \end{Bmatrix} \cdot \boldsymbol{t} + U_{l,C} \begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} \cdot \boldsymbol{t} = \boldsymbol{\nabla}\hat{u}(\mathbf{V}_l) \cdot \boldsymbol{t}, \end{aligned}$$

that is

$$\begin{aligned} U_{l,A} &= \hat{u}(\mathbf{V}_l), \\ U_{l,B} &= \frac{\boldsymbol{\nabla}\hat{u}(\mathbf{V}_l) \cdot \boldsymbol{r} - \hat{u}(\mathbf{V}_l) \begin{Bmatrix} \beta_{l,A} \\ \gamma_{l,A} \end{Bmatrix} \cdot \boldsymbol{r}}{\begin{Bmatrix} \beta_{l,B} \\ \gamma_{l,B} \end{Bmatrix} \cdot \boldsymbol{r}}, \\ U_{l,C} &= \frac{\boldsymbol{\nabla}\hat{u}(\mathbf{V}_l) \cdot \boldsymbol{t} - \hat{u}(\mathbf{V}_l) \begin{Bmatrix} \beta_{l,A} \\ \gamma_{l,A} \end{Bmatrix} \cdot \boldsymbol{t}}{\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} \cdot \boldsymbol{t}}, \end{aligned}$$

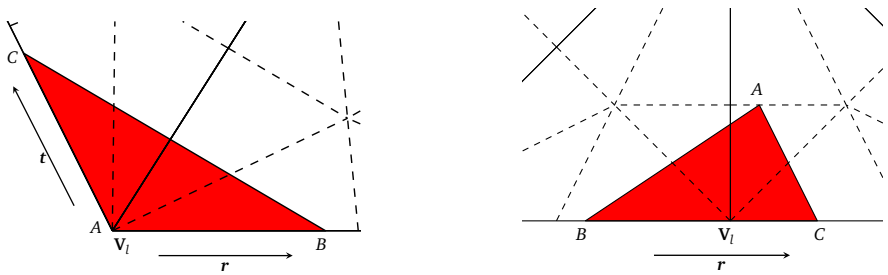


Figure 2: Control triangles for imposing boundary values: on an angle different from π (left) and equal to π (right).

where $U_{l,A}, U_{l,B}, U_{l,C}$ are the unknowns related to the shape functions with respectively triples $(\alpha_{l,A}, \beta_{l,A}, \gamma_{l,A})$, $(\alpha_{l,B}, \beta_{l,B}, \gamma_{l,B})$ and $(\alpha_{l,C}, \beta_{l,C}, \gamma_{l,C})$.

In the case $\mathbf{V}_l$ belongs to a plane angle, if the control triangle is aligned with the boundary as in Figure 2 (right), there is one shape function with zero value on $\mathbf{V}_l$, see equation (9). Hence, the unknown related to this shape function is not affected by the boundary condition. Moreover, again equations (10) and (11) guarantee that the same shape function has zero derivative in the direction parallel to the boundary. Thus, with reference to Figure 2 (right), results

$$\alpha_{l,A} = 0,\ \alpha_{l,B} \neq 0,\ \alpha_{l,C} \neq 0,$$
$$\begin{Bmatrix} \beta_{l,A} \\ \gamma_{l,A} \end{Bmatrix} \cdot \boldsymbol{r} = 0,$$

leading to the following relations for the unknowns in the vertex $\boldsymbol{V}_l$

$$U_{l,B}\alpha_{l,B} + U_{l,C}\alpha_{l,C} = \hat{u}(\mathbf{V}_l),$$
$$U_{l,B}\begin{Bmatrix} \beta_{l,B} \\ \gamma_{l,B} \end{Bmatrix} \cdot \boldsymbol{r} + U_{l,C}\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} \cdot \boldsymbol{r} = \boldsymbol{\nabla}\hat{u}(\mathbf{V}_l) \cdot \boldsymbol{r},$$

that is

$$U_{l,B} = \frac{\left(\hat{u}(\mathbf{V}_l)\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} - \alpha_{l,C}\boldsymbol{\nabla}\hat{u}(\mathbf{V}_l)\right) \cdot \boldsymbol{r}}{\left(\alpha_{l,B}\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} - \alpha_{l,C}\begin{Bmatrix} \beta_{l,B} \\ \gamma_{l,B} \end{Bmatrix}\right) \cdot \boldsymbol{r}},$$
$$U_{l,C} = \frac{\left(\hat{u}(\mathbf{V}_l)\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} - \alpha_{l,C}\boldsymbol{\nabla}\hat{u}(\mathbf{V}_l)\right) \cdot \boldsymbol{r}}{\left(\alpha_{l,B}\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix} - \alpha_{l,B}\begin{Bmatrix} \beta_{l,C} \\ \gamma_{l,C} \end{Bmatrix}\right) \cdot \boldsymbol{r}}.$$

4.2 Stabilization by artificial viscosity

Stabilized formulations are obtained adding extra terms in the Galerkin weak form, with the goal of reducing the instabilities with the introduction of artificial viscosity. Here, two stabilization terms are considered.

The first term is the streamline upwind Petrov-Galerkin (SUPG) term, used to produce a stable upwind discretization without introducing excessive numerical dissipation. The SUPG method was introduced by T. Hughes in [11] and it is one of the most established stabilized formulations in finite element flow computations. The SUPG method introduces a certain amount of artificial viscosity in the streamline direction only.

The second stabilization term is a shock capturing term, and it is used to prevent oscillations around the discontinuities arising in the solution in supersonic regimes. These spurious oscillations might lead to severe accuracy loss or stability problems. Differently from the SUPG stabilization term, the shock-capturing stabilization term introduce an isotropic artificial diffusion, but only in a sharp zone surrounding the discontinuities.

Since a first order time discretization is employed, a simplified version of the SUPG stabilization term is used

$$\mathcal{D}_{SUPG} = \sum_{e=1}^{Ne} \int_{\Omega_e} \mathbb{A}^T \boldsymbol{\nabla V} : \left(\tau_{\text{SUPG}} \mathbb{A} \boldsymbol{\nabla U}\right) d\Omega_e, \tag{12}$$

which does not take into account the complete residual of the original equation, but only the convective term.

The shock capturing stabilization term is

$$\mathcal{D}_{SC} = \sum_{e=1}^{Ne} \int_{\Omega_e} \boldsymbol{\nabla V} : \left(\tau_{\text{SC}} \boldsymbol{\nabla U}\right) d\Omega_e. \tag{13}$$

The terms τ_{SUPG} and τ_{SC} are stabilization matrices. Various options to compute these terms can be found in literature. In this work, for the SUPG stabilization term, two options have been retained: the first one is a simple diagonal matrix

$$\tau_{\text{SUPG}} \equiv \tau_{\text{SP}} = \tau \mathbf{I},$$

and a constant value τ is used in the whole mesh, $\tau = dt/2$, being dt the time step. The second choice, denoted τ_{Tz}, is inspired by the work of Tezduyar [13]

$$\tau_{Tz} = \left((1/2dt)^{(-2)} + \sum_{r=1}^{3}\sum_{q=1}^{3} \left|\boldsymbol{u} \cdot \boldsymbol{\nabla} B_r^{(q)}\right|\right)^{(-1/2)}.$$

For the shock-capturing operator, two choices described in [13, 12] are retained, and are described using four sensors. The first sensor is based on the relative density variation and it is defined, in each element, as

$$[\widetilde{\boldsymbol{\nabla}\rho}]_e = \sum_{r=1}^{3}\sum_{q=1}^{3} \left| \frac{\boldsymbol{\nabla}\rho \cdot \boldsymbol{\nabla} B_r^{(q)}}{|\boldsymbol{\nabla}\rho|} \right|.$$

The second sensor is based on the relative gradient according to the principle axes and is written

$$[\widetilde{\boldsymbol{\nabla U}}]_e = \left(\sum_{d=1}^{2} \left\| \frac{\partial \boldsymbol{U}}{\partial x_d} \right\|^2 \right)^{\frac{1}{2}},$$

while the third sensor scales the with the values of the unknowns

$$[\widetilde{\boldsymbol{U}}]_e = \|\boldsymbol{U}\|.$$

Finally the fourth sensor takes into account the local relative convective flux

$$[\widetilde{\mathbb{A}\boldsymbol{\nabla U}}]_e = \|\mathbb{A}\boldsymbol{\nabla U}\|.$$

Hence, the first shock capturing artificial viscosity is defined as a combination of the sensors with a managing parameter β as follows

$$\tau_{\text{SC1}}(\beta) = [\widetilde{\boldsymbol{\nabla}\rho}]_e [\widetilde{\boldsymbol{\nabla}\boldsymbol{U}}]_e^{-(2-\beta)} [\mathbb{A}\widetilde{\boldsymbol{\nabla}\boldsymbol{U}}]_e^{-\beta},$$

and the second

$$\tau_{\text{SC2}}(\beta) = [\widetilde{\boldsymbol{\nabla}\rho}]_e [\widetilde{\boldsymbol{\nabla}\boldsymbol{U}}]_e^{-(2-\beta)} [\widetilde{\boldsymbol{U}}]_e^{(1-\beta)} [\mathbb{A}\widetilde{\boldsymbol{\nabla}\boldsymbol{U}}]_e^{-\beta}.$$

In [12], it is suggested to use $\beta = 1$ is for smooth shocks and $\beta = 2$ for strong shocks. A compromise between the two definitions is proposed,

$$\tau_{\text{SC}} = \frac{\tau_{\text{SC}}(\beta = 1) + \tau_{\text{SC}}(\beta = 2)}{2},$$

and it is the choice used in this work.

5 NUMERICAL TESTS

5.1 Numerical simulation of a supersonic flow past a forward facing step

This is classical test case for 2D numerical codes for supersonic flows and a detailed description can be found in [14]. It consists in a supersonic flow entering an infinite long wind tunnel with a step. The interaction of the supersonic flow with the step and the tunnel walls creates a typical patter of shock reflections.

The numerical set up of the simulation is as follows (see Figure 3): the computational domain has dimensions 3×1 length units, and the step is 0.2 length units high and it is located at 0.6 length units from the left boundary. The flow is entering from the left boundary with a uniform horizontal Mach 3 velocity and pressure $p = 1$. The tunnel walls are non-slip boundaries, while the right boundary is a free exit. The simulation is started setting the variables at the inlet boundary values. No particular treatment is done for the corner of the step, which represent a singular point and it is known for introducing numerical errors, and no positivity correction was needed for the test cases here illustrated. However, with strong refinement in the corner, a negative pressure was developed in a point few elements after the corner and would require a special treatment. This point is still under investigation.

For the SUPG stabilization, the simple τ_{SM} is used. For the shock capturing term, the form $\tilde{\tau}_{\text{SC1}}$ is retained. The time step dt is linearly incremented at the beginning of the simulation, until reaching a final CFL=1. In order to compare the results with [16], four computational meshes are employed with decreasing element size $h = 1/40, 1/80, 1/160, 1/320$. The results for a simulation time $t = 4$ are depicted in Figure 4 for the density variable. A correct shock reflection pattern is obtained and the results are globally similar to the reference [16]. While the Mach steam is almost completely disappeared in the finest mesh, as expected, there is no Kelvin-Helmholtz instabilities near the top of the domain. This could be due to the extra diffusion introduced by the triangular discretization, with respect the quadrangular used in [16], and it is still not fully understood.

In Figure 5 is also shown the shock profile on a horizontal line at $y = 0.1$, for the four meshes, with a comparison with the element size. It is possible to notice that there are no overshoots in the solution across the discontinuity and the shock is spread across about four element widths, which is acceptable.

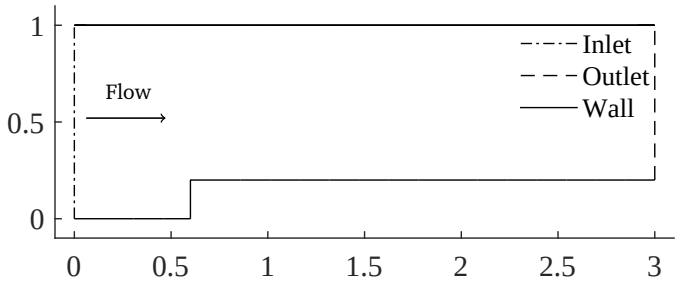


Figure 3: Forward facing step case: computational domain and boundary conditions.

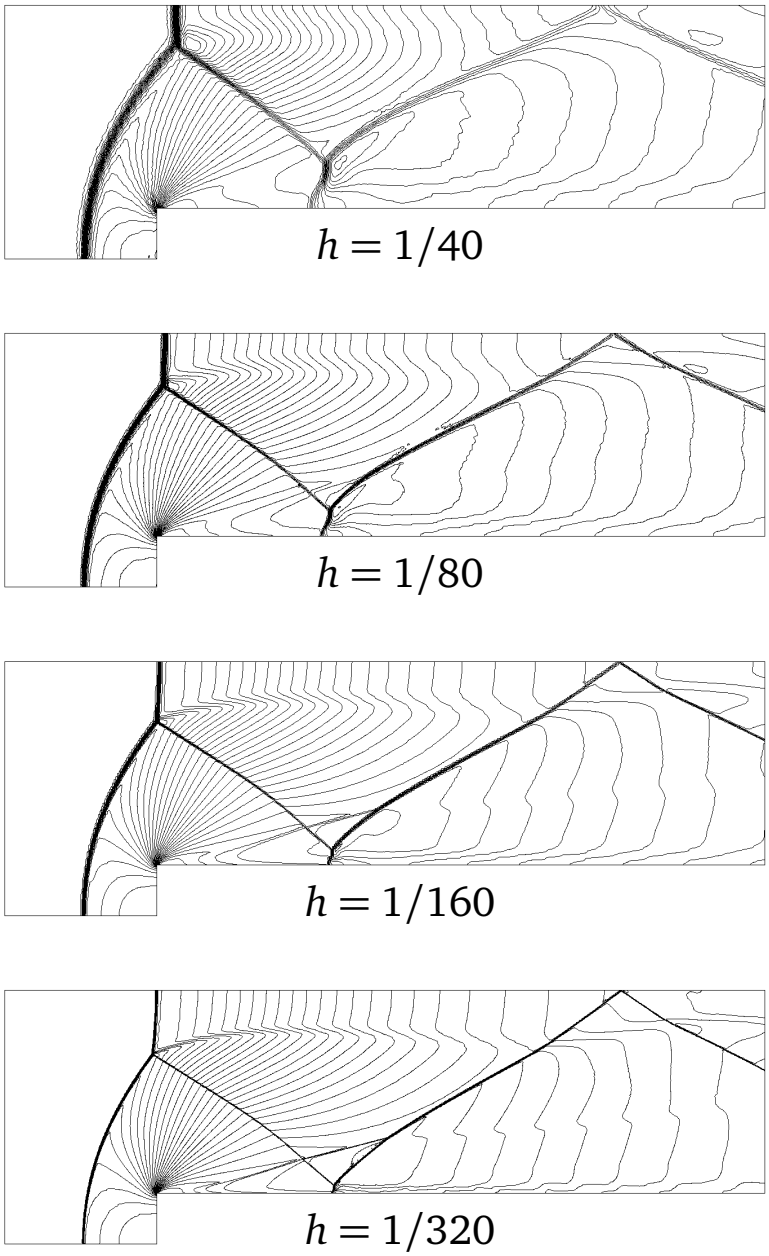


Figure 4: Forward facing step problem: density ρ, 30 equally spaced contour lines at $t = 4$.

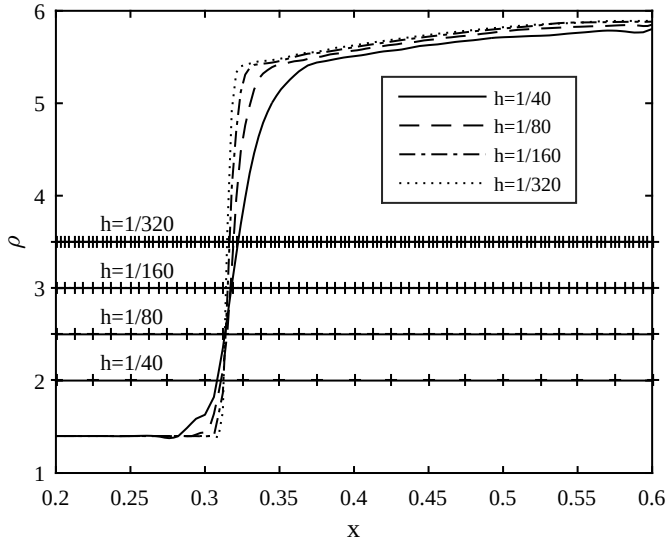


Figure 5: Forward facing step case: shock profile at a section $y = 0.1$ for the four meshes. The discretization is also displayed.

5.2 The Double Mach reflection test

The second test case considered is the Double Mach reflection test, also introduced in [14]. It represents the impinging of a strong shock wave, at Mach 10, against an oblique wall, which produce a typical reflected shock with a curved shape, and a bubble of denser gas. The set up of the test is the following, see Figure 6 : the computational domain is a box of dimensions 3×1, and a wall boundary condition is set in the lower boundary starting from $x = 1/6$. The right boundary and the remaining of the lower boundary (from $x = 0$ to $x = 1/6$) is set to an outflow boundary, while the left and the top boundaries are inlet boundaries. In the upper boundary, a time variable condition is set, simulating the traveling of the shock wave to the right. In the initial condition, the shock wave creates an angle of $60°$ from the x axis and impinges the wall at $x = 1/6$. Post-shock conditions are set to the right of the wave, pre-shock to the left.

In order to avoid spurious oscillation of the solution in the first steps of the simulation, the fields are initialized with a smooth shock profile in the domain, with the discontinuity smeared over four elements. Moreover, the time step is increased with a cubic ramp until reaching a CFL=1 in 100 time steps.

For the SUPG stabilization, similar results are obtained for τ_{SM} and τ_{Tz}. The shock-capturing stabilization however resulted more critical in this case: the parameter $\tilde{\tau}_{\text{SC1}}$ used in the previous case resulted too diffusive, hence the form $\tilde{\tau}_{\text{SC2}}$ is used instead. As in [16], four meshes are considered for comparison purpose, with element size $h = 1/60, 1/120, 1/240, 1/480$. Again, a stable solution is obtained with a sharp shock resolution. However, the shock patterns inside the bubble is not obtained, and further investigation are needed. A positivity correction was used in this case: the value of the internal energy is artificially set to 10^{-5} in those Gauss points where a negative value is obtained.

6 CONCLUSIONS

In this paper is presented a finite element method based on Powell-Sabin (PS) splines, for the solution of the 2D compressible Euler equations in supersonic regime. The mathematical and geometrical tools to define the PS shape functions are presented, in particular, the subdivision

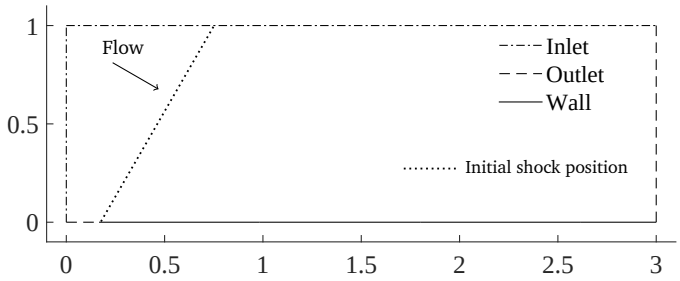


Figure 6: Double Mach reflection case: computational domain and boundary conditions.

of the elements that allows to define the PS splines as piecewise second order polynomials in each sub-triangle, the definition of the PS points and the control triangles.

A stabilized formulation is adopted, with SUPG and shock-capturing term. A simple isotropic artificial diffusion technique is adopted for the shock capturing term. Two classical examples are shown: the Woodward and Colella forward-facing step with Mach 3 flow and the Double Mach reflection test. Satisfactory results are obtained in terms of stabilization of the solution and resolution of the discontinuities. Further investigations are needed to understand the differences with the results with the reference solution and to set up an efficient positivity correction procedure.

REFERENCES

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

[2] Y. Bazilevs, V.M. Calo, J. A. Cottrell, T. J. R. Hughes, A. Reali, G. Scovazzi, Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows, Comput. Methods Appl. Mech. Eng. 197 (1-4) (2007) 173–201.

[3] Y. Bazilevs, V.M. Calo, J. A. Cottrell, J. A. Evans, T. J. R. Hughes, S. Lipton and M. A. Scott and T. W. Sederberg, Isogeometric analysis using T-splines, Comput. Methods Appl. Mech. Eng. 199 (5-8) (2010) 229–263.

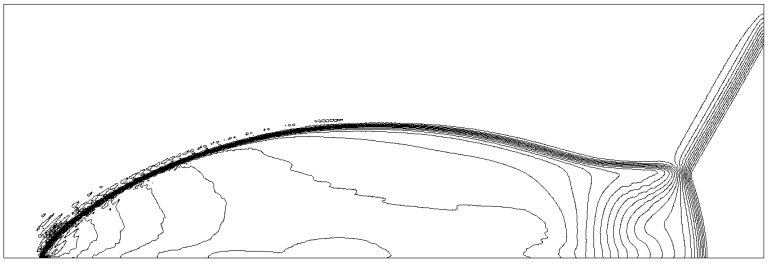
[4] M. J. Lai, L. L. Schumaker, Spline Functions on triangulations, Cambridge University Press, 2007.

[5] M. J. D. Powell, M. A. Sabin, Piecewise quadratic approximations on triangles, ACM Trans. Math. Softw. 3 (4) (1977) 316–325.

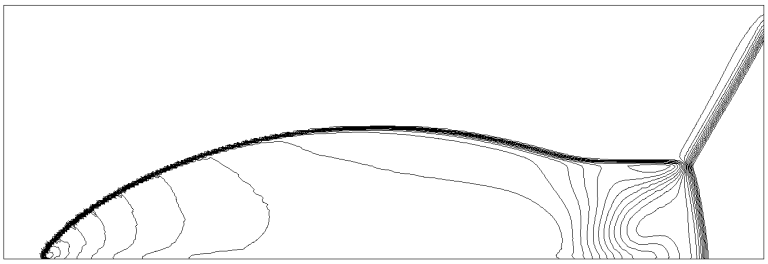
[6] N. Jaxon , X. Qian, Isogeometric analysis on triangulations, Computer-Aided Design 46 (2014) 45–57.

[7] H. Speleers, C. Manni, F. Pelosi, M. L. Sampoli, Isogeometric analysis with Powell-Sabin splines for advection-diffusion-reaction problems, Comput. Methods Appl. Mech. Eng., 221-222 (2012) 132–148.

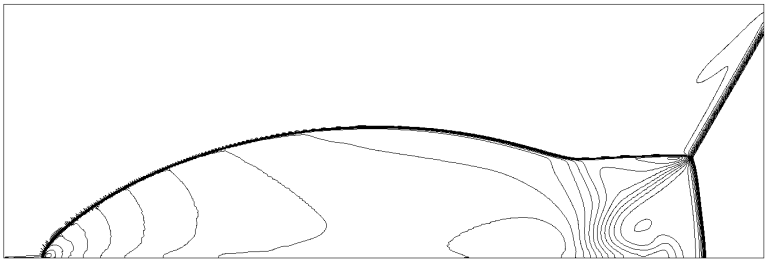
[8] P.Dierckx, On calculating normalized Powell-Sabin B-splines, Comput. Aided Geom. D., 15 (1) (1997) 61–78.



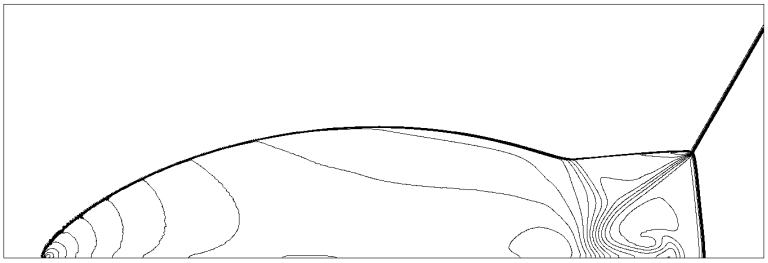
$h = 1/60$



$h = 1/120$



$h = 1/240$



$h = 1/480$

Figure 7: Double Mach reflection problem: density ρ, 30 equally spaced contour lines at $t = 0.2$.

[9] J. O'Rourke, A. Aggarwal, S. Maddila, M. Baldwin, An optimal algorithm for finding minimal enclosing triangles, J. Algorithm., 7 (2) (1986) 258–269.

[10] H. Speleers, P. Dierckx, S. Vandewalle, Powell-Sabin splines with boundary conditions for polygonal and non-polygonal domains, J. Comput. Appl. Math., 206 (1) (2007) 55–72.

[11] A. N. Brooks and T. J. R. Hughes, Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Eng., 32 (1990) 199–259.

[12] C. Wervaecke, H. Beaugendre, B. Nkonga, A fully coupled RANS Spalart-Allmaras SUPG formulation for turbulent compressible flows on stretched-unstructured grids, Comput. Methods Appl. Mech. Eng., 233-236 (2012) 109–122.

[13] T. E. Tezduyar, M. Senga, Stabilization and shock-capturing parameters in SUPG formulation of compressible flows, Comput. Methods Appl. Mech. Eng., 195 (2006) 1621–1632.

[14] P. Woodward, P. Colella, The numerical simulation of two-dimensional fluid flow with strong shocks, J. Comput. Phys., 54 (1) (1984) 115–173.

[15] J. Donea, A. Huerta, Finite element methods for flow problems, John Wiley & Sons, Chichester (UK), 2003.

[16] B. Cockburn, C. W. Shu, The Runge-Kutta discontinuous Galerkin method for conservation laws V, J. Comput. Phys., 141 (1998) 199–224.