Modeling plate shell structures using pyFormex

Anne BAGGER*, Benedict VERHEGGHE[a], Kristian Dahl HERTZ[b]

* M.Sc., Ph.D. candidate,
Technical University of Denmark, Building 118, 2800 Kgs. Lyngby, Denmark
aeb@byg.dtu.dk

[a] Professor, Ghent University, Belgium
[b] Professor, Technical University of Denmark

Abstract

A shell structure made of glass combines a light-weight structural concept with glass' high permeability to light. If the geometry of the structure is plane-based facetted (plate shell structure), the glass elements will be plane panes, and these glass panes will comprise the primary bearing structure. A plate shell structure is contrary to a triangulated facetted shell structure, where the shell action is concentrated in the edges and vertices of the geometry, thereby resulting in the need for a triangulated lattice structure outlining the edges of the geometry. These two structural principles (plate shells and triangulated lattice shells) may not differ in complexity regarding the topology, but when it comes to the practical generation of the geometry, e.g. in CAD, the plate shell is far more troublesome to handle than the triangulated geometry. The free software tool "pyFormex", developed at Ghent University, has been used to accommodate a parametric generation of plate shell structures. This generation includes the basic facetted shell geometry, joint areas that reproduce given connection characteristics, loads and boundary conditions. From pyFormex the model is exported to the finite element analysis software Abaqus as a Python script, which translates the information to an Abaqus CAE-model. In pyFormex the model has been prepared for applying the meshing in Abaqus, by allocation of edge seeds, and by defining geometry sets for easy handling.

Keywords: glass, pyFormex, plate shells, shell structures, Abaqus

1. Introduction

In order to introduce plane structural elements in a shell structure, without losing the huge advantages of the doubly curved shell shape, *facetted shell structures* are often considered. A facetted shell structure has a piecewise plane geometry, as illustrated in figure 1. Two fundamentally different faceting schemes are illustrated in the figure: point-based faceting (left) and plane-based faceting (right).
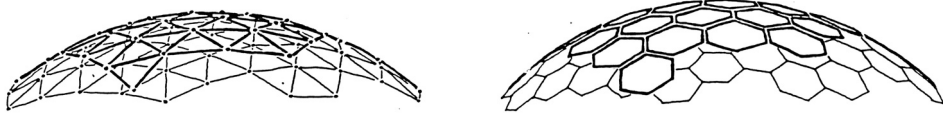
Figure 1: Facetted geometry. Left: point-based (triangulated). Right: plane-based.

In a point-based system, three points define each plane (facet). Lattice shell structures with triangulated truss systems belong to this category, where a geometry of plane triangles approximate a smooth shape.

In a plane-based system, the smooth shape is approximated by a number of planes. In this system, three planes will define a point. As a result, three facets intersect in all vertices (Wester [11]).

The structural behaviour of the two systems corresponding to the two different faceting schemes are very different from each other. In the triangulated system the forces are concentrated in the edges and vertices. In the plane-based system the forces are distributed in the facets and are transferred as in-plane forces between the facets (Bagger *et al.* [4] [3] [2]). The two types of facetted systems can be described as each others dual (Wester [11]).

As a structural principle, a plane-based facetted shell geometry – termed *plate shell structure* – is of special interest, since the surface itself is the primary load bearing structure. There is no need for structural components other than the facets themselves. If the facets are made of glass, this will result in a simple structure, consisting only of the glass facets themselves and their connections, see figure 2.
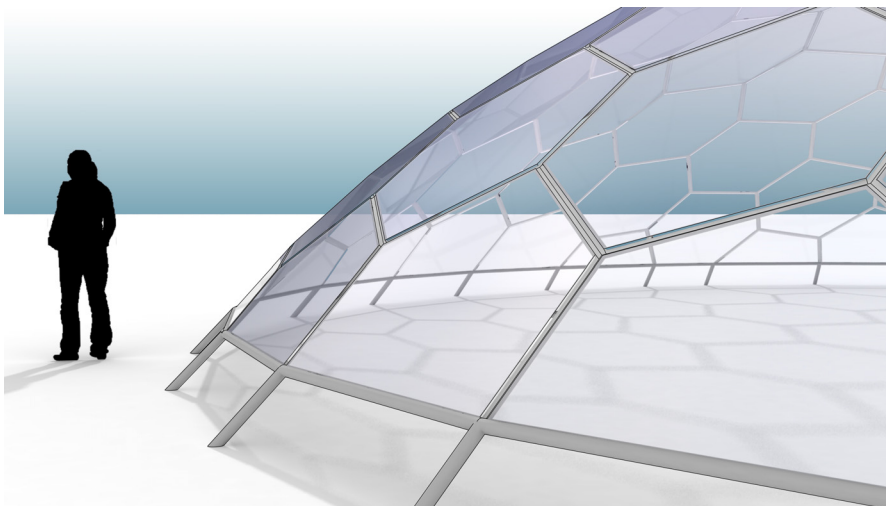
Figure 2: Visualization of plate shell structure by M. Rippman, ILEK, Universität Stuttgart.

When it comes to generating/drawing the geometry of a plate shell structure, most existing CAD tools are unsuitable. The topology of a plate shell is no more complex than that of a triangulated shell geometry, but the geometry is far more complex to handle in CAD. Most

CAD tools handle smoothly curved surfaces as triangulated (point-based) facetted surfaces, which serve as approximation to the smoothly curved surface. A plane-based facetted geometry is difficult to handle in such an environment, because the edges/vertices must be determined as intersection lines/points between defined planes. If the location of a vertex is translated (which is what CAD-tools support), the surrounding facets are no longer plane. To generate and manipulate a plane-based facetted geometry, the facets must be treated as separate planes, and the edges and vertices as a consequence of the position of these planes.

A special tool is therefore needed to handle the plane-based geometry of a plate shell structure. To accommodate easy processing of the geometry, parametric alterations and the generation of a finite element model to run in Abaqus [8], a script has been developed for the software tool *pyFormex*.

pyFormex is a powerful free software currently under development [6] by B. Verhegghe *et al*. at Ghent University. pyFormex contains a Python implementation of Formex algebra pioneered by H. Nooshin [9] and was developed for the automated design of spatial structures by means of sequences of mathematical transformations ([6], Verhegghe [10], Belis *et al*. [5]).

This paper will present the above mentioned pyFormex script, partly to show a way to generate and handle an otherwise troublesome structural concept – the plate shell structure – and partly to illustrate the power of pyFormex as a tool for parametric generation of spatial geometry and FE models. Section 2 describes the pyFormex script from the design engineers point of view. Section 3 presents a technical description of the script and the processes involved. The script has been developed by B. Verhegghe in cooperation with A. Bagger.

2. Using the script

2.1 User defined input

As described in section 1, a plane-based facetted geometry is generated by determining the position of each facet's plane in space. In the pyFormex script, this is done by loading a text-file with a list of points, all of which are points on a convex surface (see figure 3, left). Such a file can be generated in CAD by projecting points onto a convex surface, or by determining the location of the points numerically, e.g. by using MATLAB. (If the original shape is not convex, other conditions apply – in the present context only convex shapes are considered.)

For each point in the loaded list, pyFormex generates a facet using the tangent plane to the original convex surface in that point. The relevant intersection lines and points are found, thereby producing a convex plane-based geometry, like the one shown in figure 4.c. This is the basic geometry, which is subsequently manipulated into a FE model of a plate shell structure, using interactively defined user input (see figure 3, right). The possible input options are described in table 1.

Cut plane	The basic geometry is cut by a plane, erasing everything under the plane. This way, a plane boundary for supporting the

	structure is obtained.
Span and height	The basic geometry can be stretched to meet a given size in the ground plane and in the height. This is possible without warping the facets, since points in a plane will remain in a plane after scaling in one or more directions.
Joint geometry	In Abaqus the connections between the facets are modeled by a narrow strip of elements between the facet edges (see e.g. figure 5). The geometric and material characteristics of this joint strip are adjusted, so that the joint strip reproduces the stiffness of a given physical connection detail. (A Ph.D. study in preparation at Technical University of Denmark by A. Bagger is, among other issues, investigating this adjustment and its feasibility.) The pyFormex user specifies the geometric layout of the joint strip.
Mesh	The element size, or the number of elements along a group of edges, is specified by the user as input for the meshing, which is performed in Abaqus. The element type is also specified, chosen among three different shell elements.

Table 1: User defined input.

2.2 Output

Based on the user defined input described in section 2.1, the pyFormex script generates a Python script for Abaqus/CAE. When this script is run in Abaqus a CAE-file is generated containing the following information:

→   Plate shell geometry.
→   Material parameters for facets (at present: glass) and joint strips.
→   Element mesh.
→   Defined sets of geometric surfaces and lines for convenient handling of the geometry in Abaqus.
→   Defined steps for analysis: an initial step defining a pinned support of the structure's boundary in the cut plane, and a load step where gravity load is applied to the structure.
→   Defined job, ready to submit for analysis.
→   Defined request for analysis output (stress components, translations, rotations, section forces and section moments).

The resulting model in Abaqus is ready to be submitted for analysis. The user can adjust the joint parameters, adjust the mesh if necessary and apply other loads.
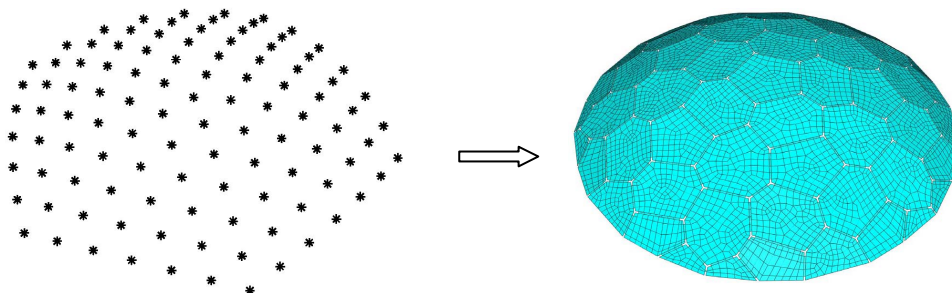
Figure 3: Input to pyFormex script is a cloud of points on a convex surface. Output is a script ready to run in Abaqus, which generates a FE model of a plate shell structure.

3. Procedures of the pyFormex script

This section gives a technical description of the script and the procedures involved.

Step 1. Read the array of points lying on the smooth shell surface. *See figure 4.a.*

Step 2. Construct a convex triangulation through these points. pyFormex uses the external program 'tetgen' [7] by Hang Si to achieve this. 'tetgen' is a quality tetrahedral mesh generator, which as an offspring also generates the triangular boundary mesh. The resulting boundary represents a closed surface, but based on the angles between adjacent triangles, pyFormex can easily distinguish between the shell surface and its closure at the bottom. *See figure 4.b.*

Step 3. Define normal directions at all points. These can either be calculated from the known analytical expression of the surface (e.g. in case of a spherical surface) or be auto-generated from the triangulation (taking the mean normal of all triangles around the point) or be generated by other means.

Step 4. At all points, tangent planes are generated with the respective normal. The intersection of these planes will yield the polygons of the facetted surface. The algorithm basically proceeds as follows. The intersection points of all triplets of planes are the candidate vertices for the polygons. First, at each point the dual polygon is constructed: it results from the intersections with the planes at the points connected in the triangulation, in the same order. A check of the rotation sense of the subsequent polygon edges around the normal reveals the need for swapping vertices with other candidate vertices, or reversing the order of the vertices. In the end, a sound polygonal surface is constructed. *See figure 4.c.*

Step 5. The facetted surface is cut by a horizontal plane at ground level. *See figure 4.d.* As a result, the number of vertices on the cut facets may decrease or increase. The edges that are in the cut plane are marked for applying the support conditions later on.

Step 6. The resulting surface can now be scaled horizontally and/or vertically to achieve given span width or height.

Step 7. The final surface needs to be further processed to introduce joints between the individual polygonal facets. Polygon edges having contact with another facet are offset over a distance equal to half the desired joint width, in the direction of the polygon's center and new vertices are calculated from the offset lines. This procedure may result in slightly unequal edge lengths for the two meeting faces and in reduction of the number of edges of the facet. *See figure 4.e.*

Step 8. The resulting space between two adjacent facets is filled with a rectangular strip representing the joint. The joint is usually taken somewhat shorter than the full length of the facet edges. *See figure 4.f.*

Step 9. The facet edges connected to a joint are split up in three parts, according to the actual joint length and the offsets at the ends. This enables setting matching mesh seeds on facets and joints.

Step 10. To finalize the geometrical model, extremely small facets and joints may be discarded.

Step 11. Material properties, boundary conditions and loads are defined, as well as a number of geometry sets in order to facilitate the input to Abaqus and the processing of the results of the simulation.

Step 12. Finally, the pyFormex script exports all data, including mesh seeds and element type, in the format of an Abaqus/CAE Python script. The facets and joints are exported as individual plane shell parts. Each part is built from a sketch with the 2D coordinates of the part's vertices in its plane. The parts are then assembled giving them the proper translation and rotation to position them correctly in 3D space. The mesh seeds on adjacent facet and joint edges are set equal, to enable compatible meshing.

Step 13. Running the exported script in Abaqus will create the element mesh for all parts and prepare the analysis job for submission.
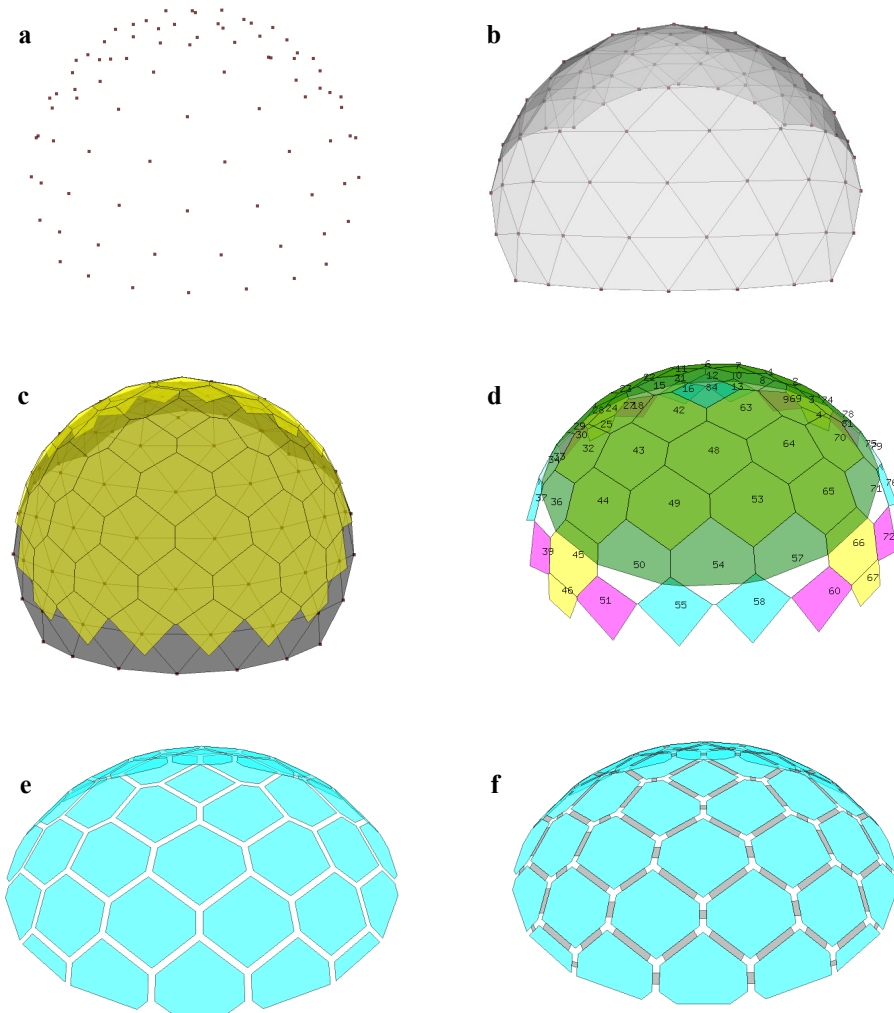
Figure 4: Illustration of steps in the pyFormex script.

4. Examples of use

In this section, the use of the pyFormex script is illustrated by some examples. Table 2 gives an overview of the examples.

Structure	Model	Illustrating figure	
Section of a sphere Span 11.5m, height 3.47m	A	Fig. 5	Geometry, surface stress plot (maximum principal stress, upper surface of facets), meshing.
		Fig. 6	Surface stress plot, area with refined mesh.
		Fig. 7	In-plane stress plot, area with refined mesh.
Paraboloid of revolution Span 11.5m, height 1.92m Span 11.5m, height 3.84m	B C	Fig. 8 Fig. 9	Geometry, surface stress plot, meshing. Geometry, surface stress plot, meshing.
		Fig. 10	Surface stress plot, area with refined mesh.

Table 2: Overview of structures in the illustrated examples.

All calculations are geometrically linear, and the materials are linear elastic. The facets are modeled as glass, with an E-modulus of $70*10^3$ N/mm^2, a poisson's ratio of 0.22, and a density of 25kN/m^3. The glass thickness is 15mm. The joints are modeled by 20mm wide strips of elements with an E-modulus of $8.14*10^3$ N/mm^2, a poisson's ratio of 0.0, a density of 10kN/m^3 and a thickness of 6.15mm. These element strips represent a connection stiffness of 16kNm/m against rotation of the plate edges, and 5kN/mm^2 against movements of the facets perpendicular to their edge (in their own plane). The issue of connection details and representation of their stiffness parameters will be addressed in the Ph.D. dissertation finished later this year by A. Bagger.

The first structure, model A, is a facetted approximation to a section of a sphere with a span of 11.5m and a height of 3.47m. The structure is generated using the pyFormex script, followed by some mesh adjustments and submission of analysis in Abaqus/CAE. The distance between the facet corners and the end of the joint strips is 150mm. (The joint strips are shorter than the plate edges because stress concentrations occur in both the facets and their connections if the facet corners are connected to each other. This paper will not go into detail with this effect.)

The second structure, models B and C, is a facetted approximation to a paraboloid of revolution. The span is like that of the first structure, 11.5m. The height of model B is 1.92m. Model C has been scaled in the vertical direction by a factor 2 compared to model B, making the height 3.84m. The distance between the facet corners and the end of the joint strips is 80mm in both models.

All three models are pinned along the facet edges in the horizontal cut plane, and loaded by a distributed vertical load of $3.7kN/m^2$ (load per plate surface area). A relatively coarse mesh (6-10 elements per facet edge) yields accurate results for the structure's deformations, but poor precision for the stresses. Therefore, if the user knows where the design stresses in the structure will be, the mesh can be refined in that specific area, and a coarse mesh can be applied to the rest of the structure. This will reduce the calculation time significantly. Test for stress convergence at locations of large stress variations can also be carried out by refining the mesh in that area.

Selected results are illustrated in the following figures. Table 2 shows an overview of these figures. All stresses are in N/m^2.
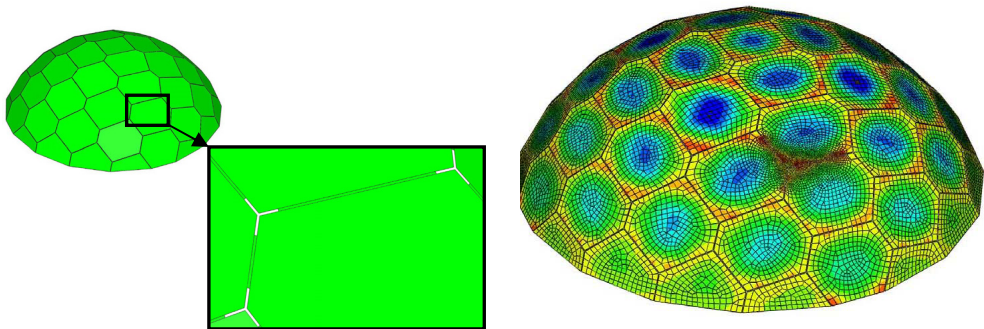
Figure 5: Plate shell structure, model A (see table 2). Left: geometry and detail at facet corner. Right: surface stress plot (maximum principal stress in N/m^2, upper surface of the facets). The mesh is refined in the darker area.
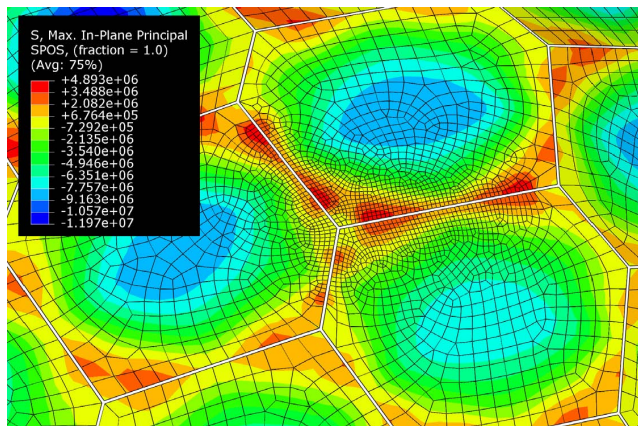

Figure 6: Plate shell structure, model A (see table 2). Maximum principle stress (N/m^2), upper surface of the facets, at area with refined mesh.
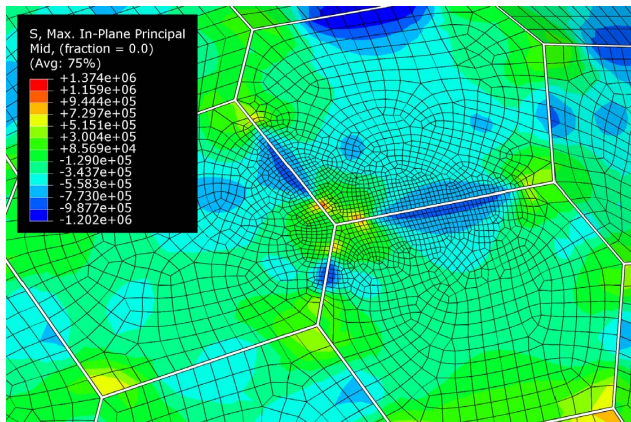
Figure 7: Plate shell structure, model A (see table 2). Maximum principle stress (N/m^2), middle surface of the facets, at area with refined mesh.
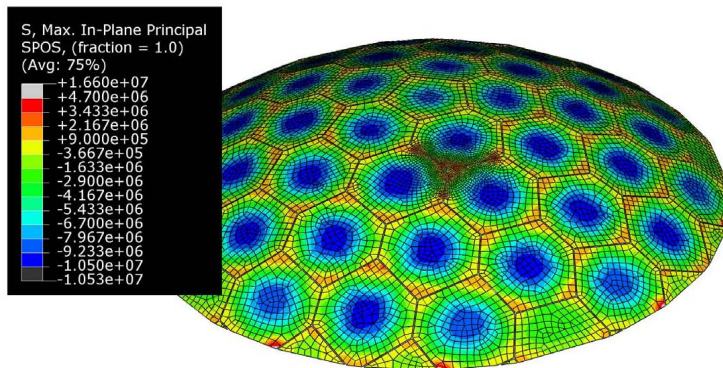

Figure 8: Plate shell structure, model B (see table 2). Maximum principle stress (N/m^2), upper surface of the facets. The mesh is refined in the darker area.
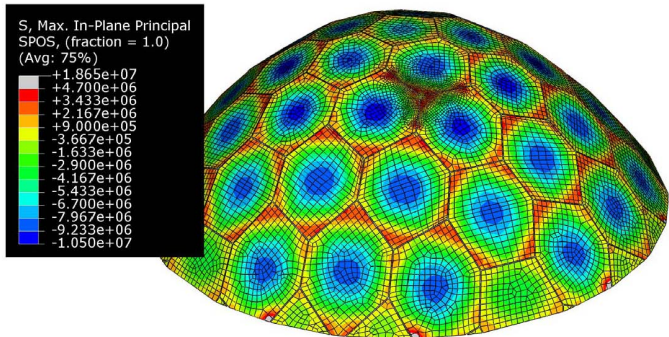
Figure 9: Plate shell structure, model C (see table 2). Maximum principle stress (N/m^2), upper surface of the facets. The mesh is refined in the darker area.
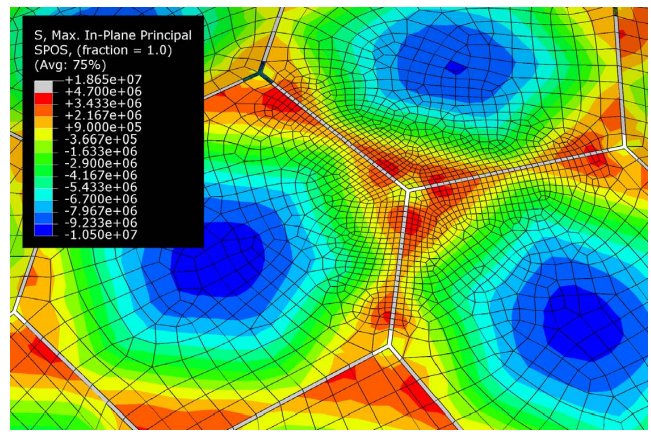

Figure 10: Plate shell structure, model C (see table 2). Maximum principle stress (N/m^2), upper surface of the facets, at area with refined mesh.

5. Conclusion and future development

The pyFormex script has proven extremely useful in the generation of plate shell models. Previously, the generation of a plate shell geometry and FE model has taken 2-3 weeks with existing software tools (CAD and Abaqus/CAE). With the pyFormex script, the generation of a full FE model can be handled in a matter of minutes, and parametric adjustments can easily be done.

Many useful additions can be made to the script. For instance, relevant loads can be auto-generated in pyFormex. Snow load on a facet depends on the angle between the facet and the horizontal plane, and since this information is readily available in the script, the application of snow load to the structure can easily be implemented. Wind load can be defined for each element as a basic pressure value, multiplied with a factor which varies with the element's position in the horizontal plane.

To model the finite stiffness of the supports, an element strip – with a lower stiffness than the glass – can be added between the supported glass edge and the support.

The pyFormex script could be extended to export the geometry in other formats (e.g. DXF) for visualization purposes, or create detailed part lists for manufacturing and construction purposes.

Geometric imperfections could be implemented by the pyFormex script, as input for geometrically non-linear FE studies.

Acknowledgements

References

[1]   Almegaard H., *Skalkonstruktioner – metoder til afklaring af sammenhængene mellem form, stabilitet, stivhed og understøtninger*, Ph.D. dissertation, Royal Danish Academy of Fine Arts – School of Architecture, 2003.

[2]   Bagger A., Jönsson J., Almegaard H., Bending stresses in facetted glass shells, *Challenging Glass, Conference on Architectural and Structural Applications of Glass, Conference Proceedings*, Delft, 2008.

[3]   Bagger A., Jönsson J., Almegaard H., Wester T., Facetted Shell Structure of Glass, *Glass Processing Days, Conference Proceedings*, Tampere, 2007.

[4]   Bagger A., Jönsson J., Wester T., Investigation of Stresses in Facetted Glass Shell Structures, *International Association of Shell and Spatial Structures (IASS) Symposium, Conference Proceedings*, Venice, 2007.

[5]   Belis J., Verhegghe B., De Beule M., Van Impe R., Evaluation of Glass Domes using pyFormex, *International Association of Shell and Spatial Structures (IASS) Symposium, Conference Proceedings*, Venice, 2007

[6]   http://pyformex.org

[7]   http://tetgen.berlios.de

[8]   http://www.simulia.com

[9]   Nooshin H., "Algebraic Representation and Processing of Structural Configurations", *International Journal of Computers and Structures*, Vol.5, Issues 2-3, 1975.

[10] Verhegghe B., *pyFormex Manual*, Release 0.8-a1, Ghent, 2009.

[11] Wester T., *Structural Order in Space – the plate-lattice dualism*, The Royal Danish Academy of Fine Arts, School of Architecture, Smed Grafik, 1984.