Shell finite elements
There are three types of shell finite element; 1) flat elements, 2) elements based on the Sanders-Koiter equations and 3) elements based on reduction of a solid element.

Flat elements are triangles or quadrilaterals. A flat element is based on a simple combination of a disc element (plane stress) and a plate element (bending). Every node has six degrees of freedom (dofs) (Fig. 28, 29, 30). The red dofs do not really contribute to the element accuracy. They are added to make the element fit in a general purpose finite element program. Small elements need to be used due to their low accuracy [1].

Figure 28. Plate membrane element  Figure 29. Plate bending element

Figure 30. Degrees of freedom of flat shell elements
Curved elements can be derived from the Sanders-Koiter equations. A well known element of this type is the semiloof element [2]. It has been derived by Bruce Irons based on discussions with Henk Loof. The element has 3 degrees of freedom in 8 nodes and 1 rotational degree of freedom in 8, so called, Loof nodes (Fig. 31). This thin shell element has high accuracy, however, it is difficult to implement in a finite element program. Therefore, it is not much used.

1 Bruce Irons (1924–1983) was professor at Swansea and Calgary. He was specialised in programming finite elements. He made important contributions to this field and wrote three books on computational analysis. He suffered from multiple sclerosis and committed suicide at the age of 59 [www.wikipedia.org, sept. 2013].

Henk Loof (1929–1988) was professor at Delft University of Technology, Faculty of Civil Engineering. He was very skilful in the mathematics of shell structures. He was not married and lived in the city of Den Haag together with his sister [source Johan Blaauwendraad en Coen Hartsuijker]. The “oo” in Loof is pronounced as the “o” in go.

The development of the semiloof element can be described shortly. Irons met Loof at a conference in Newcastle in 1966. Irons presented a paper on integration rules and Loof presented a paper on shell finite element analysis [3]. In an informal setting they must have talked at length about shell behaviour and shell mathematics. In the years that followed Irons derived a finite element with rotation degrees of freedom in unusual points at the edges. He referred to these points as Loof nodes after his good friend Henk Loof. When he presented his element at a conference in 1974 he modestly called it the SemiLoof element [2]. Surely a better name for the element is the Irons-Loof element but this name change did not take place. The semiloof element is regarded by many specialists a scientific master piece [source …].
Shell elements can also be derived from solid elements. In the process some degrees of freedom are replaced by others and the constitutive equations are simplified (Fig. 32). These elements have 3, 4, 6 or 8 nodes with each 6 degrees of freedom and can be implemented conveniently. The elements with 4 nodes can be twisted. The elements with 6 and 8 nodes can be curved as well (Fig. 33) [3].

![Figure 31. Degrees of freedom of a semiloof element](image)

*Figure 31. Degrees of freedom of a semiloof element*

Element aspect ratio
The aspect ratio of a rectangular shell element is defined as length over width. Many finite element programs have a restriction on the aspect ratio. For example

\[
\frac{1}{20} < \frac{\text{length}}{\text{width}} < 20
\]

The reason for this restriction is that if the element stiffness in two directions is very different the structural stiffness matrix has both very large numbers and almost zero numbers on the main diagonal. As a consequence the computed displacements and stresses may have little accuracy. However, in modern software this is not a problem because high accuracy number representations are used. Sometimes, we need to use aspect ratios of 1000 and this does not need to give accuracy problems.

Mesh refinement
Like all finite elements, a shell element is accurate when it is small. An engineer who is experienced in finite element analysis can just see whether the elements in a model are sufficiently small. However, when in doubt the following procedure is used. 1) Do the analysis with any mesh. 2) Refine the mesh to half the element size. 3) Repeat the analysis. 4)
If the important results do not change significantly, the previous mesh was sufficient. If the important results change significantly, continue at step 2.

For example, the first analysis gives a deflection of 24 mm. The second analysis, with half mesh size, gives a deflection of 26 mm. If you think that 2 mm is sufficient accuracy than you are done. We can estimate the exact result that would be obtained by an extremely fine mesh. For this add the difference to the last result. In this example the exact result is approximately 26 + 2 = 28 mm.

Refining a shell mesh to half element size, requires approximately 4 times as much computer memory and 8 times as much computation time (out-of-core computation).

**Element accuracy**

Element accuracy cannot be expressed as a percentage. This is because the accuracy depends on the situation in which an element is used. What we do know is the smaller the element, the smaller the error. For example, the element deformation can have an error of $O(h)$. (Pronounce “order h”). This means that the error is proportional to the element size $h$. It is the smallest finite element accuracy possible. Other errors are $O(h^2)$ and $O(h^3)$. The table below gives the errors of shell finite elements $[4][5][6].$

<table>
<thead>
<tr>
<th>Element type</th>
<th>deflection</th>
<th>membrane forces</th>
<th>moments</th>
<th>shear forces</th>
</tr>
</thead>
<tbody>
<tr>
<td>flat element</td>
<td>$O(h^2)$</td>
<td>$O(h)$</td>
<td>$O(h^2)$</td>
<td>$O(h)$</td>
</tr>
<tr>
<td>semi-Loof element […]</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>reduced solid element without midside nodes</td>
<td>$O(h^2)$</td>
<td>$O(h^2)$</td>
<td>$O(h^2)$</td>
<td>$O(h^2)$</td>
</tr>
<tr>
<td>reduced solid with midside nodes</td>
<td>$O(h^2)$</td>
<td>$O(h^2)$</td>
<td>$O(h^2)$</td>
<td>$O(h^2)$</td>
</tr>
</tbody>
</table>

**Model accuracy**

In the example on mesh refinement it is assumed that the deformation has an error $O(h)$. The table below shows more formulas for estimating the exact result from two computation results.

<table>
<thead>
<tr>
<th>$h_2$</th>
<th>$u = u_2 + (u_2 - u_1)$</th>
<th>$u = u_2 + (u_2 - u_1)/3$</th>
<th>$u = u_2 + (u_2 - u_1)/7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.500</td>
<td>$h_1$</td>
<td>$u = u_2 + 2.41(u_2 - u_1)$</td>
<td>$u = u_2 + 0.547(u_2 - u_1)$</td>
</tr>
<tr>
<td>0.707</td>
<td>$h_1$</td>
<td>$u = u_2 + 3.85(u_2 - u_1)$</td>
<td>$u = u_2 + 1.70(u_2 - u_1)$</td>
</tr>
<tr>
<td>0.794</td>
<td>$h_1$</td>
<td>$u = u_2 + 3.85(u_2 - u_1)$</td>
<td>$u = u_2 + 1.70(u_2 - u_1)$</td>
</tr>
</tbody>
</table>

The table results have been obtained from two equations. For example

```plaintext
> eq1 := u = u1 + C*h^2;
> eq2 := u = u2 + C*(0.500*h)^2;
> solve({eq1, eq2}, {u, C});
```

**Selecting the element type**

Suppose we can choose to do an analysis with $O(h)$ elements or $O(h^2)$ elements. Which type is best? Of course, we want accurate results and a fast computation. Figure 33b shows some computation result as a function of the number of nodes. This graph is typical for a

---

2 Element accuracy can be determined by performing three analysis; the second with half element size and the third with one-fourth element size. This gives three equations

```plaintext
> eq1 := u = u1 + C*h^b;
> eq2 := u = u2 + C*(h/2)^b;
> eq3 := u = u3 + C*(h/4)^b;
> solve({eq1, eq2, eq3}, {b, u, C});
```

from which the order of the error can be solved $b = \frac{2 \log u_2 - u_1}{u_3 - u_2}$. 67
complicated structure [5]. If we are satisfied with an error of 10% or larger then \(O(h)\) elements require the least number of nodes and the least computation time. If we need a smaller error then the \(O(h^2)\) elements need the least computation time. From this we can conclude,

Choose the most accurate element that is available unless you are just testing.

Also the shape of the elements is important. Quadrilaterals are more accurate than triangles of the same order.

![Typical convergence of a finite element result for \(O(h)\) and \(O(h^2)\) elements](image)

**Integration points**

In all finite elements the material behaviour (stresses, stains, yielding, cracks, etc.) is computed in a number of points (Fig. 33). These points are called integration points or Gauss points. The stresses etc. in other points of the element are computed by interpolation and extrapolation.

![Possible locations of integration points in triangular elements](image)

**Locking and hourglass modes**

In some element types the out of plane bending stiffness is too large. This is called shear locking. Some element types are too stiff for extensional deformations. This is called membrane locking. These locking problems can be solved in three ways; 1) very fine mesh 2) different elements or 3) reduced integrations. In reduced integrations specific integration points that are needed for exact computation of the element stiffness are omitted. This can be a very effective trick to improve the element accuracy. Most finite element programs use reduced integration. It can be switched off but it is not wise to do so.

Due to reduced integration the elements may have no stiffness at all for particular deformations. Consequently, the elements can deform in a pattern that looks like hourglasses (Fig. 35). This deformation is called an hourglass mode or a zero energy mode. Clearly, this is not what we want and all handbooks give warnings for the phenomenon. However, an hourglass mode can only occur in a perfectly regular mesh with special boundary conditions. In a practical finite element model these hourglass modes are extremely rare. The author and
his colleagues have never observed one despite many years of experience. If you would ever see an hourglass mode in a finite element model please make a picture of the screen and send it to me.

![hourglass shape](image)

*Figure 35. Deformation of square elements into an hourglass mode*

**The largest model that your PC can process**

Modern computer programs for numerical analysis use numbers with double precision. This means that each number is stored in 8 bytes of memory. One byte is equal to 8 bits. A bit is represented by an electrical switch with can assume two voltage levels.

The most important operation that a finite element program performs is solving a very large system of equations that is represented in a matrix. This matrix has a length and width equal to the number of degrees of freedom (dofs) of the finite element model. This matrix needs to be stored in the memory of the PC. For example, if a model with 15000 dofs is analysed the computer needs $15000 \times 15000 \times 8 = 1.8 \times 10^9$ bytes of memory. This is 1.8 GB (gigabyte). A new PC has approximately 3 GB memory of which about 1 GB is used by Windows (2010). Therefore, the model of this example can be analysed in memory. The linear elastic computation can be performed within a minute.

If the matrix does not fit in memory then the software can move most of the matrix to the hard disk. This computation is called out-of-core. For example, if a model has $10^5$ dofs the required hard disk space is $10^7 \times 10^5 \times 8 = 80$ GB. A partition on a hard disk might have 150 GB (2010), of which 90 GB might be free for performing the analysis. This is sufficient for analysing this example. The linear elastic computation time can be half an hour or more. If you listen carefully, you can hear the hard disk becoming active. Then you know that the computation will take more time than a minute.

Many finite element programs use smart methods to optimise the computation. 1) The matrix is often symmetrical so only half of it needs be stored. 2) Most of the numbers in the matrix are just zero. The non-zero numbers occur around the matrix diagonal. Therefore, only the numbers within some distance from the diagonal need be stored. 3) This distance is called band width. The band width can be reduced by sorting the node numbers of the finite element model. 4) Some programs have an iterative solver that does not need any matrix for solving the system of equations. Therefore, the largest model that can be analysed depends strongly on efforts of the software engineers. For example, the finite element program Ansys can analyse a model of $10^6$ dofs in half an hour on a normal PC (2007). The largest model also depends on the analyses choices that the software user makes, for example, yes or no node sorting.

**Moore’s law**

Moore’s law is [6]

*Computation power doubles every two years.*
This law describes accurately the development of computation power since 1971. It is expected to be valid in the near future too. So, if your current PC cannot analyse a particular model, it is not difficult to calculate when your future PC can do this job.

**Arithmetic accuracy**

A double precession number has a precision of about 16 significant digits and a magnitude range of approximately $10^{-308}$ to $10^{308}$. Some precision is lost in every addition, subtraction, multiplication and division. This is inevitable. After solving a large matrix the result can have just 3 significant digits. This is sufficient for most applications. The software should give a warning if the arithmetic is not accurate but some programs do not.

**Finite element benchmarks**

Shell elements need to be tested to determine their accuracy. Three tests are often applied; a cylinder (Fig. 36), a hemisphere (Fig. 37) and a hemisphere with an opening (Fig. 38). The cylinder is closed on both ends by a diaphragm, therefore, the edge nodes are fixed in the $x$ and $y$ directions. Note that due to symmetry just part of the shells needs to be modelled. A finite element program can be checked by comparing the displacement under the forces with the results of others. The reference displacement of the cylinder directly under the force is 1.8248 mm [9]. The displayed mesh is too course for most applications. Approximately 1000 elements will be needed to obtain 1% accuracy. The reference displacement of the hemisphere is 0.0924 m directly under the forces [9]. Approximately 200 elements will be needed to obtain 1% accuracy. The reference displacement of the hemisphere with an opening is 0.0935 m directly under the forces [9]. Approximately 100 elements will be needed to obtain 1% accuracy.

![Figure 36. Cylinder loaded by opposite forces](image)

**Figure 36. Cylinder loaded by opposite forces**

- $E = 3 \times 10^6$ N/mm$^2$
- $\nu = 0.3$
- $a = 300$ mm
- $t = 3$ mm

![Figure 37. Hemisphere loaded by opposite forces](image)

**Figure 37. Hemisphere loaded by opposite forces**

- $E = 6.825 \times 10^7$ N/m$^2$
- $\nu = 0.3$
- $a = 10$ m
- $t = 0.04$ m
Thick shells
In a thick shell the shear deformation can be important compared to bending deformation. Shear deformation is included in Mindlin-Reissner elements. These elements can be necessary to obtain sufficient accuracy.

In a very thick shell the stress distribution is not linear (normal stresses) or parabolic (shear stresses) over the thickness. Volume elements can be necessary to compute the stresses accurately. The element mesh needs to have several volume elements over the shell thickness. Volume elements are also called solids, bricks or tets. The last is short for tetrahedrons.

Averaging over nodes
In the finite element method all elements in the model are in equilibrium. However, the stresses etc. on either side of element edges can be different. This is a result of approximations in the element formulation. Many programs can average the computation results in the nodes to make very smooth contour plots (Fig. 39). It needs to be kept in mind that this also removes any real jumps in the results. For example, a real jump in the stresses occurs when adjacent shell elements have different thicknesses.

Stresses in the ultimate limit state
The ultimate limit state (ULS) is defined as a situation in which the structure is collapsing. This can be computed with a finite element program using nonlinear analysis. In this analysis the load is increased in small steps until collapse.

The load that the structure needs to carry is defined in codes of practice, for example the eurocode. Usually the code specifies not just one load but multiple load combinations. Engineers calculate the stresses due to these load combinations. These are referred to as “stresses in the ultimate limit state”.

\[ E = 6.825 \times 10^7 \text{ N/m}^2 \]
\[ \nu = 0.3 \]
\[ a = 10 \text{ m} \]
\[ t = 0.04 \text{ m} \]
Clearly, this choice of words is incorrect because the structure is not collapsing (in general). It
would be better to refer to these stresses as

“stresses due to a load combination that the structure needs to be able to resist before or
when the ultimate limit state occurs”.

The latter choice of words is very impractical and therefore the shorter version is often used.
However, it is very important to understand their real meaning.

Types of analysis
Most finite element programs can do many types of structural analysis. The following types
can be performed by any well educated structural engineer.
– linear analysis
– buckling analysis
– natural frequency analysis
– temperature distribution analysis
– second order analysis (geometrical nonlinear up to the serviceability limit state)

The following types of analysis are advanced. Therefore, you need to do a course on the
subject before you can perform the analysis and understand the results.
– geometrical nonlinear analysis
– physical nonlinear analysis
– transient analysis

What type of analysis to use?
Clearly, we choose the simplest analysis type that does the job. Following is a list of “jobs”
and the analysis that can be used.

<table>
<thead>
<tr>
<th>Job</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>stresses in the ultimate limit state</td>
<td>linear analysis *</td>
</tr>
<tr>
<td>stresses due to support settlements</td>
<td>linear analysis *</td>
</tr>
<tr>
<td>displacements in the serviceability limit state</td>
<td>linear analysis or second order analysis **</td>
</tr>
<tr>
<td>buckling critical load factors</td>
<td>linear buckling analysis</td>
</tr>
<tr>
<td>concrete crack widths (SLS)</td>
<td>hand calculation or physical nonlinear analysis</td>
</tr>
<tr>
<td>load factor at collapse</td>
<td>geometrical and physical nonlinear analysis including creep and realistic imperfections</td>
</tr>
</tbody>
</table>

* If the structure performs well the displacements will be small and nonlinear effects are negligible. If the structure
does not perform well we can use the too large stresses to design a better shape and dimensions.
** The deformations of most well designed shells are very small and second order analysis is not needed.