Computational Geosciences manuscript No.
(will be inserted by the editor)

A Volume-conserving Representation of Cell Faces in Corner Point Grids

Per Røe · Ragnar Hauge

Received: date / Accepted: date

Abstract Corner point grids is currently the standard grid representation for use in reservoir simulation. The cell faces in corner point grids are traditionally represented as bilinear surfaces where the edges between the corner points all are straight lines. This representation has the disadvantage that along faults with varying dip the cell faces on either side will not precisely match, giving overlapping cells or gaps between cells. We propose an alternative representation for the cell faces. The four vertical cell faces are still represented as bilinear surfaces, but instead of having linear edges between the cell corners along the top and bottom faces we propose a representation of the vertical cell faces where any horizontal intersection will give a straight line, giving column faces whose shape is independent of the corner point locations of the individual grid cells. This ensures that the grid columns match up and that there are no gaps or overlapping volumes between grid cells. This representation gives a local parameterization for the whole grid column, and the top and bottom grid cell surfaces are modelled as bilinear using this parameterization. A set of local coordinates for the grid cell permits all the common grid operations like volume calculation, area calculation for cell faces and blocking of well traces.

Keywords Corner point grid · Volume calculations · Well blocking · Reservoir Simulation Grid

P. Røe
Norwegian Computing Center
P.O.Box 114, Blindern, N-0314 Oslo, Norway
Tel.: +47 22 85 25 00
Fax: +47 22 69 76 60
E-mail: per.roe@nr.no

R. Hauge
Norwegian Computing Center

Mathematics Subject Classification (2000) MSC 86A60

1 Introduction

Corner point grids [2] have become the standard way of setting up grid models for reservoir simulation, and is supported by all major reservoir modelling software suites and reservoir simulators. These grids are a good compromise between the simplicity of regular grids and the flexibility of unstructured grids. As structural modelling allows a higher number of faults with more complex linkage patterns, more and more complex grids are requested. This requires simple but reliable algorithms for tasks like volume calculations, transmissibility calculations and well blocking.

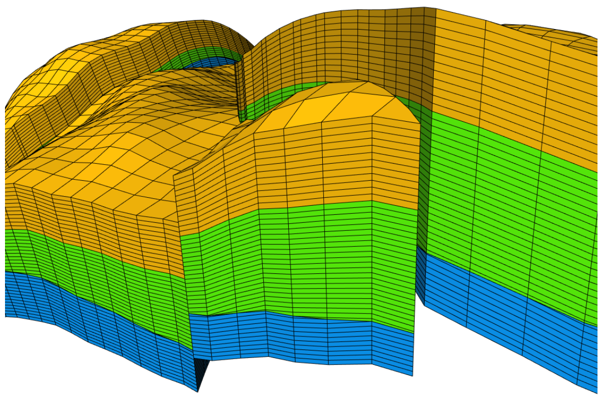

Fig. 1 Simulation grid for a reservoir with three reservoir zones, shown in yellow, green and blue, and with multiple faults.

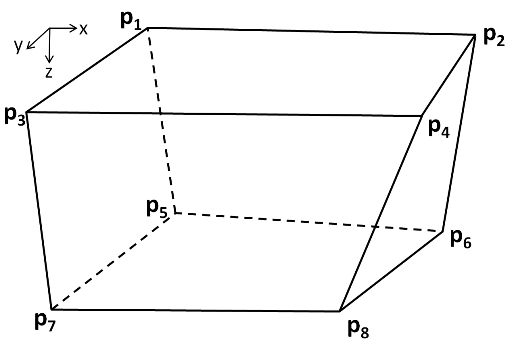

Fig. 2 The eight cell corners for a single grid cell. The points $\mathbf{p}_1$ and $\mathbf{p}_5$ are on the same pillar. The same goes for points $\mathbf{p}_2$ and $\mathbf{p}_6$, $\mathbf{p}_3$ and $\mathbf{p}_7$, and $\mathbf{p}_4$ and $\mathbf{p}_8$.

The grid is usually generated so that the layering of the grid follows the reservoir zonation. This is done to simplify the property modelling. It is also important for the correctness of the fluid simulations that the geometry of the grid cells follow the principal flow directions [1]. Traditionally, faults are gridded so that the grid pillars follow the faults (Fig. 1), but stair-stepped grids are getting more common due to the ability to model more complex fault geometries.

A corner point grid is defined by a set of linear pillars together with the z-values along these pillars for each of the corner points of the individual grid cells. In this paper we will label the individual corner points $\mathbf{p}_i = (x_i, y_i, z_i)$ as shown in Fig. 2. Traditionally a bilinear representation where also the edges between the pillars (eg. $\mathbf{p}_1 - \mathbf{p}_2$ in Fig. 2) are assumed to be linear is used. As shown in [2], this gives the simple trilinear formula for going from local coordinates $(u, v, w) \in [0, 1]^3$ to global coordinates $\mathbf{p} = (x, y, z)$,

$$\begin{aligned}\mathbf{p} =&(1-w)\big((1-v)((1-u)\mathbf{p}_1 + u\mathbf{p}_2)+\\ &\qquad v((1-u)\mathbf{p}_3 + u\mathbf{p}_4)\big)+\\ &w\big((1-v)((1-u)\mathbf{p}_5 + u\mathbf{p}_6)+\\ &\qquad v((1-u)\mathbf{p}_7 + u\mathbf{p}_8)\big).\end{aligned} \tag{1}$$

This has proven to be a flexible representation, but there are some problems. As noted by [2], "at a fault which is sloping by a changing amount, and along which the fault throw increases, the two bilinear surfaces of connecting grid blocks will not precisely match". This will usually be the case to some extent along fault traces, especially when pillar gridding of the faults is employed. An extreme case of a gap between neighbour grid cells is shown in Fig. 3 **(a)-(b)**. If the layers on either side of the fault tilt the opposite way, the cells would overlap instead as shown in Fig. 3 **(c)-(d)**. These gaps and overlaps give imprecisions in the volume calculations. Also, as discussed later in this paper, finding the cell containing a given point within the grid and finding the corresponding local parameters is non-trivial.

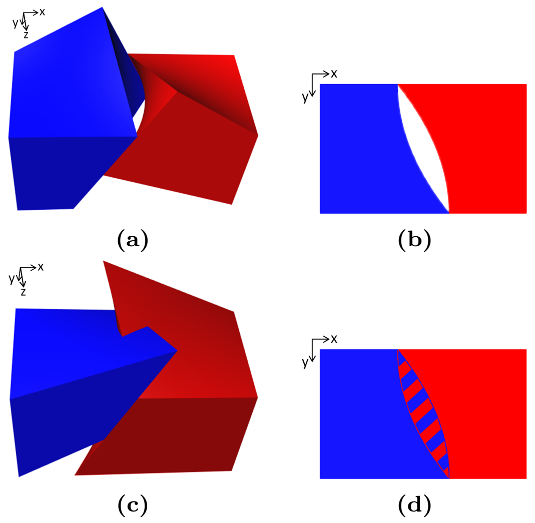

Fig. 3 Two neighbour cells with non-matching layering and a large difference in the direction of the two common grid pillars, seen from **(a)** the side, above and **(b)** through a horizontal intersection. Notice the gap between the cells. In **(c)** and **(d)** two cells are shown with the same common grid pillars but with opposite tilt of the layers, giving overlapping cells.

In this paper we will present an alternative way of representing the grid cell faces in a corner-point grid. The representation is based on the classical parameterization using grid pillars and depths values for grid cell corner points, but with an alternative of way of defining the grid cell faces. No pre-processing of the input data for the grid is needed. The main idea is that instead of using bilinear cell faces where the lateral edges between the cell corners are linear, we ensure that any horizontal intersection through the grid will give linear cell sides. This representation ensures that there are no gaps between neighbour cells and no overlapping cells, and gives well-defined top and bottom surfaces as long as the grid cells are convex. However, due to the more complex representation of the cell faces, the calculations of cell face area and cell volume become more tedious. On the other hand, finding the cell containing a point and the calculations of partial cell volumes or partial cell face areas become much easier. Based on the proposed representation, we will present and discuss key algorithms in reservoir simulation grids: volume calculation, finding the cell containing a given point, well blocking and transmissibility calculations.

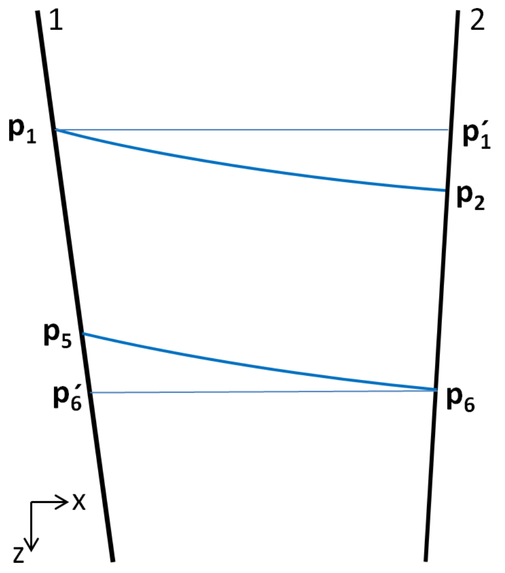

Fig. 4 A single vertical cell face shown using the proposed representation, with $\mathbf{p}_1 - \mathbf{p}_2$ and $\mathbf{p}_5 - \mathbf{p}_6$ represented as parabolas and with the projection points $\mathbf{p}_1'$, and $\mathbf{p}_6'$ used in the definition of the cell face representation.

2 Proposed representation

2.1 Column sides

The main idea behind the proposed new grid cell face representation is that all horizontal intersections through the four vertical cell faces should be linear. Traditionally, the arcs $\mathbf{p}_1 - \mathbf{p}_2$ and $\mathbf{p}_5 - \mathbf{p}_6$ in Fig. 4 have been represented as straight lines. We now allow these arcs to be non-linear, and instead we require that the horizontal arcs $\mathbf{p}_1 - \mathbf{p}_1'$ and $\mathbf{p}_6' - \mathbf{p}_6$ are linear, where the point $\mathbf{p}_1'$ on pillar 2 is at the same depth z_1 as $\mathbf{p}_1$, and $\mathbf{p}_6'$ on pillar 1 is at the same depth z_6 as $\mathbf{p}_6$

A point $\mathbf{p}(u,\tilde{w})$ on the bilinear surface defined by the points $\mathbf{p}_1, \mathbf{p}_1', \mathbf{p}_6'$ and $\mathbf{p}_6$, with $(u,\tilde{w}) \in [0,1]^2$ as the bilinear weighting of these points, is then given as

$$\mathbf{p}(u,\tilde{w}) = (1-u)(\mathbf{p}_1 + \tilde{w}(\mathbf{p}_6' - \mathbf{p}_1)) + u(\mathbf{p}_1' + \tilde{w}(\mathbf{p}_6 - \mathbf{p}_1')), \tag{2}$$

where the z-component of $\mathbf{p}(u,\tilde{w})$ is

$$z(u,\tilde{w}) = z_1 + \tilde{w}(z_6 - z_1). \tag{3}$$

Notice that this means that z is independent of u, meaning that the straight line obtained when keeping $\tilde{w}$ constant will be horizontal for any $\tilde{w}$. Furthermore, notice that the surface defined by the four points in Eq. 2 will, where overlapping, be equal to any surface defined by a set of four points, pair-wise on the same depth on the two pillars. The shape of the cell sides is only dependent on the pillars, not on the corner points of the actual cells. This ensures that the sides of two neighbour grid columns will always match, independently of the gridding within the individual columns.

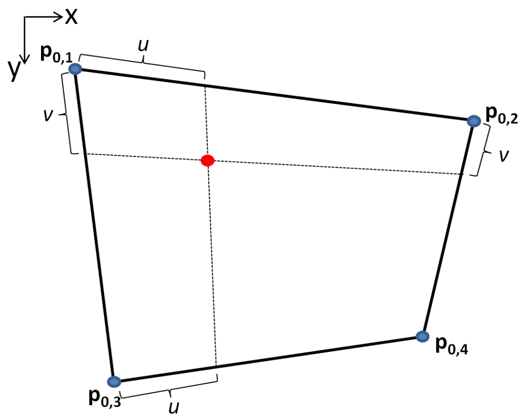

Fig. 5 Definition of local lateral relative coordinates u and v within a grid column.

For a given horizontal intersection through a grid cell at depth z_0 with corresponding points on the grid pillars $\mathbf{p}_{0,i} = (x_{0,i}, y_{0,i}, z_0)$ (Fig. 5), a point $\mathbf{p}(u,v) = (x(u,v), y(u,v), z_0)$ within the cell can be given as a bilinear weighting of the pillar points by $(u,v) \in [0,1]^2$ as

$$\begin{aligned} x(u,v) &= (1-v)((1-u)x_{0,1} + ux_{0,2}) + \\ &\qquad v((1-u)x_{0,3} + ux_{0,4}) \quad \text{and} \\ y(u,v) &= (1-v)((1-u)y_{0,1} + uy_{0,2}) + \\ &\qquad v((1-u)y_{0,3} + uy_{0,4}). \end{aligned} \tag{4}$$

This defines a local coordinate system where any point at depth z_0 can be represented by the local coordinates (u,v). This (u,v) parameterization is only dependent on the grid pillars, not the individual grid cells, and hence common for all the grid cells within the grid column defined by these four grid pillars. This parameterization is well-defined as long as the pillar intersection points $\mathbf{p}_{0,i}$ span a convex quadrilateral for all depths z_0. Concave quadrilaterals are discussed in Sec. 4.1

Let $\mathbf{d}_i$ be the z-normalized direction of the pillars so that a general point at depth z on pillar i is given as $\mathbf{p}_{0,i} + (z - z_0)\mathbf{d}_i$. A point within the column at depth z and local lateral coordinates (u,v) will then be given as

$$\begin{aligned}\mathbf{p}(u,v,z) = {} & (1-v)((1-u)\mathbf{p}_{0,1}+u\mathbf{p}_{0,2})+ \\ & v((1-u)\mathbf{p}_{0,3}+u\mathbf{p}_{0,4})+ \\ & \big((1-v)((1-u)\mathbf{d}_1+u\mathbf{d}_2)+ \\ & v((1-u)\mathbf{d}_3+u\mathbf{d}_4)\big)(z-z_0),\end{aligned} \tag{5}$$

which is a straight line going through

$$\begin{aligned}\mathbf{p}_0(u,v) = {} & (1-v)((1-u)\mathbf{p}_{0,1}+u\mathbf{p}_{0,2})+ \\ & v((1-u)\mathbf{p}_{0,3}+u\mathbf{p}_{0,4})\end{aligned} \tag{6}$$

with z-normalized direction

$$\begin{aligned}\mathbf{d}(u,v) = {} & (1-v)((1-u)\mathbf{d}_1+u\mathbf{d}_2)+ \\ & v((1-u)\mathbf{d}_3+u\mathbf{d}_4).\end{aligned} \tag{7}$$

2.2 Top and bottom surfaces

In general any top and bottom surface can be used, as long as there for each surface is a single intersection point between the surface and the pillar given by a pair of local coordinates (u,v) for any $(u,v) \in [0,1]^2$. The local parameterization $z^{\text{top}}(u,v)$ and $z^{\text{bot}}(u,v)$ is then defined from these intersection points. A local depth coordinate w can then be defined as the relative position between the top and bottom surfaces

$$w(z,u,v) = \frac{z - z^{\text{top}}(u,v)}{z^{\text{bot}}(u,v) - z^{\text{top}}(u,v)}. \tag{8}$$

We propose to use a representation of the top and bottom surfaces where the surfaces are bilinear in the (u,v)-domain. For a grid cell where the z coordinates of the corner points are z_1-z_8 (see Fig. 2), the grid cell top and bottom surfaces z^{top} and z^{bot} are given by

$$\begin{aligned}z^{\text{top}}(u,v) &= (1-v)((1-u)z_1+uz_2)+ \\ &\quad v((1-u)z_3+uz_4) \\ z^{\text{bot}}(u,v) &= (1-v)((1-u)z_5+uz_6)+ \\ &\quad v((1-u)z_7+uz_8).\end{aligned} \tag{9}$$

This gives a simple representation in the (u,v) domain, ensuring the simplicity of the ensuing algorithms. We also see that for a well-defined grid cell where $z_5 \geq z_1$, $z_6 \geq z_2$, $z_7 \geq z_3$ and $z_8 \geq z_4$, the top and bottom surfaces will be well-ordered, $z^{\text{bot}}(u,v) \geq z^{\text{top}}(u,v)$. This representation is also equivalent with the traditional representation in the common case when all the grid pillars are vertical.

Instead of using the simple local bilinear representation given in Eq. 9 which in general is not smooth across grid columns, one could for example use globally interpolated surfaces generated using B-splines as top and bottom surfaces. The disadvantage with this is that now $z^{\text{top}}(u,v)$ and $z^{\text{bot}}(u,v)$ in general only can be found by solving the corresponding equation systems, meaning that the volume and area of the cell faces would have to be calculated numerically.

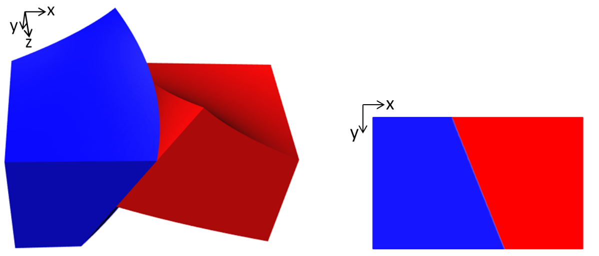

Fig. 6 The same two cells as in Fig. 3 **(a)-(b)**, visualized using the proposed new cell face representation.

Figure 6 shows the two neighbour cells from Fig. 3 **(a)-(b)** visualized using the proposed cell face representation. As we see, the sides of the two neighbour cells match, even though the two shared grid pillars have opposite dips. This will always be true with the new representation; the lateral grid column faces for two neighbour columns, and hence also the cell faces for the individual grid cells, will always match. Similarly, the top surface of a cell will always match the bottom surface of the cell above as long as the cell corner points match.

3 Algorithms

3.1 Grid cell volume

The volume of a single grid cell can be calculated as

$$V = \int_0^1\int_0^1\int_0^1 J\,du\,dv\,dw \tag{10}$$

where J is the Jacobian

$$J = \begin{vmatrix} f_u(u,v,w) & f_v(u,v,w) & f_w(u,v,w) \\ g_u(u,v,w) & g_v(u,v,w) & g_w(u,v,w) \\ h_u(u,v,w) & h_v(u,v,w) & h_w(u,v,w)\end{vmatrix}. \tag{11}$$

The functions $f(u,v,w)$, $g(u,v,w)$ and $h(u,v,w)$ for going from local coordinates (u,v,w) to global coordinates (x,y,z) are given by Eqs. 20-22 in the appendix. The calculations are a bit more tedious than the similar volume calculations done using the traditional grid cell parameterization [2,4] due to more complex terms, but they are still straightforward. With the new representation it is also possible to calculate the volume for a whole grid column at the time in the same manner.

The results from this calculation is exactly the same as the sum of the volume of the individual grid cells in the column as long as the top surfaces of all cells match the bottom surfaces of the cells above. Also note that in contrast with the traditional representation, the volume is dense; the sum of the volume of individual grid cells is equal to the volume between the top and bottom surfaces for the whole grid.

The cell volume above a given surface, for example a fluid contact, can be found in similar manner. If $z^{\text{surf}}(u,v)$ is the representation of the surface in local coordinates, with the special case $z^{\text{surf}}(u,v) = z$ for a horizontal surface, then the integral

$$V = \int_0^1 \int_0^1 \int_{w^{\text{surf}}}^1 J \, dw \, dv \, du, \tag{12}$$

where

$$w^{\text{surf}}(u,v) = \frac{z^{\text{surf}}(u,v) - z^{\text{top}}(u,v)}{z^{\text{bot}}(u,v) - z^{\text{top}}(u,v)} \tag{13}$$

can be used to calculate the volume, either for the volume above the surface within a single cell, or for the volume above the surface for a part of or the whole grid column. Note that finding the volume above a surface is much simpler using the proposed representation than using the traditional representation due the need of representing the surface in local coordinates.

3.2 Algorithm for finding grid cell containing a given point

Using the traditional algorithm, it is non-trivial to find the grid cell containing a given point. Candidate cells can be found using a fast rejection algorithm by examining boxes enclosing the individual cells or group of cells. However, the only way to find the actual cell is by checking whether the point is on the inside of all the cell faces. This is non-trivial since all the cell faces are bilinear surfaces. Also, due to the fact that the cells might intersect or there might be a gap between cells no unique solution is guaranteed; the point might be within two cells, or between cells.

Using the proposed representation, it is simple to find the grid cell containing a given point using the following algorithm:

The first step is to find the grid column containing this point given by its global coordinates (x, y, z). This is done by taking the horizontal intersection through the grid at depth z. As previously noted, this will give a set of quadrilaterals, and finding the quadrilateral containing the point in two dimensions is straightforward.

The second step is to find the local coordinates (u, v) for the point. This is done by solving the bilinear system of equations

$$\begin{aligned} x &= (1-v)\big((1-u)x_{z,1} + ux_{z,2}\big) + \\ &\qquad v\big((1-u)x_{z,3} + ux_{z,4}\big) + \\ y &= (1-v)\big((1-u)y_{z,1} + uy_{z,2}\big) + \\ &\qquad v\big((1-u)y_{z,3} + uy_{z,4}\big), \end{aligned} \tag{14}$$

where $(x_{z,i}, y_{z,i}, z)$ is the point on pillar i at depth z. The cell containing the point within the column is then found by comparing the z-coordinate of the point with the z-coordinate for the top surfaces for the cells in the column at (u, v). If the local depth coordinate w is needed, it can be calculated using Eq. 8.

3.3 Cell face area calculations

The area of a cell face is found by setting the appropriate local coordinate to 0 or 1 in Eqs. 20-22 and integrating the remainder of the coordinates. For example, for the vertical cell face given by $u = 0$, the mapping from local to global coordinates is

$$\begin{aligned} x = f(v,w) &= (1-v)\big(x_1 + \frac{h(v,w) - z_1}{z_5 - z_1}(x_5 - x_1)\big) + \\ &\qquad v\big(x_3 + \frac{h(v,w) - z_3}{z_7 - z_3}(x_7 - x_3)\big) \\ y = g(v,w) &= (1-v)\big(y_1 + \frac{h(v,w) - z_1}{z_5 - z_1}(y_5 - y_1)\big) + \\ &\qquad v\big(y_3 + \frac{h(v,w) - z_3}{z_7 - z_3}(y_7 - y_3)\big) \\ z = h(v,w) &= (1-w)((1-v)z_1 + vz_3)) + \\ &\qquad w((1-v)z_5 + vz_7)), \end{aligned} \tag{15}$$

and the integral for calculating the area becomes

$$A = \int_0^1 \int_0^1 \left\| \frac{\partial \mathbf{x}}{\partial v} \times \frac{\partial \mathbf{x}}{\partial w} \right\| dv \, dw, \tag{16}$$

where $\mathbf{x}(v,w) = (x,y,z) = (f(v,w), g(v,w), h(v,w))$.

Similar to the volume calculations, the area of parts of the cell face can be found in the same manner as the area of the whole cell face, and the calculated area of the different parts of the cell face will sum up to the total area. This is also true for the area of the intersection between two faulted cells, which is not even properly defined using the traditional representation.

3.4 Transmissibility calculations

The calculation both of cell face area projections and the projected area of cell face intersections are important parts of the calculation of the transmissibilities used in fluid simulation. The contribution from a single cell to the transmissibilities for an interface is given by

$$T = K|\mathbf{A} \cdot \mathbf{d}|/d^2 \tag{17}$$

where K is the permeability, $\mathbf{A} = (A_x, A_y, A_z)$ is the vector of cell face area projections, and d is the length of $\mathbf{d}$, the vector going from the centre of the cell to the centre of the cell face contributing to $\mathbf{A}$, see [2].

When using the traditional representation, the projections of the intersection become quadrilaterals, making it trivial to calculate the projected area. With the proposed representation the edges of the top and bottom surfaces are no longer linear. In this case the area is calculated by integrating in the local projected coordinate system. For example, the x-projection A_x of the cell face corresponding to $u = 0$ is calculated as

$$A_x = \int_0^1 \int_0^1 J dv\, dw, \tag{18}$$

where J is the Jacobian

$$J = \begin{vmatrix} g_v(v,w) & g_w(v,w) \\ h_v(v,w) & h_w(v,w) \end{vmatrix}. \tag{19}$$

For a faulted interface (Fig. 7), the projected area of the intersection between two neighbour cell faces can be calculated as in [2] by finding the intersection points between the top and bottom surfaces for the cells on either side of the interface, and then summing up the area between the individual intersection points. Finding the intersection points in the (u, v) domain is straightforward since the top and bottom surfaces are bilinear in this domain and the intersection lines with the vertical cell faces are hence linear. A common w is needed, and in Fig. 7 we have suggested using the horizontal lines corresponding to the highest and lowest z value for the individual patches (z_0 and z_1 in the figure), giving $w = (z - z_0)/(z_1 - z_0)$.

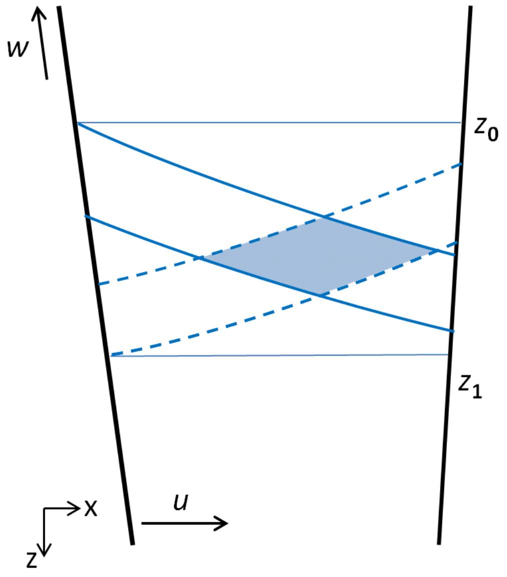

Fig. 7 The projected interface between two neighbour cells on either side of a fault. The solid lines are the intersection lines for the top and bottom faces of the cell on the hanging wall side, while the dotted lines are the corresponding lines for the cell on the footwall side. The intersection of the two cell faces is shown as shaded.

3.5 Well blocking

Another common task related to gridding is well blocking, i.e. the process of upscaling well data to the grid scale. The upscaled well path is the sequence of grid cells that the well path passes through. To find this upscaled well path, the cells containing the points along the well path has to be identified, and as seen in Sec. 3.2 this is becomes much easier using the new representation. The blocked well is for each cell populated with an upscaled measure of the log values within the cell. To be able to do the upscaling, all log values have to be assigned to the cell containing the point associated with the log value. The length of the well segment can also be used for weighting the log values.

To be able to find the length of the well path segment going through a cell, the intersection point between the cell face and the well path needs to be found. When finding the intersection point, we must differentiate between the vertical and horizontal cell faces due to the differences in representation. The vertical cell faces are bilinear surfaces, and the intersection point can be found analytically as described in [3]. Note that the calculations can be numerically unstable, especially if the well path is close to parallel with the cell face, but that this to some extent can be alleviated since we can do the calculations for the whole grid column face, instead of the individual cell faces. To calculate the intersection point between the well path and the top or bottom cell face analytically, we would need to represent the well path in local coordinates. This is possible, but in general a numerical algorithm based on a linear approximation of the cell face and binary search might be just as convenient.

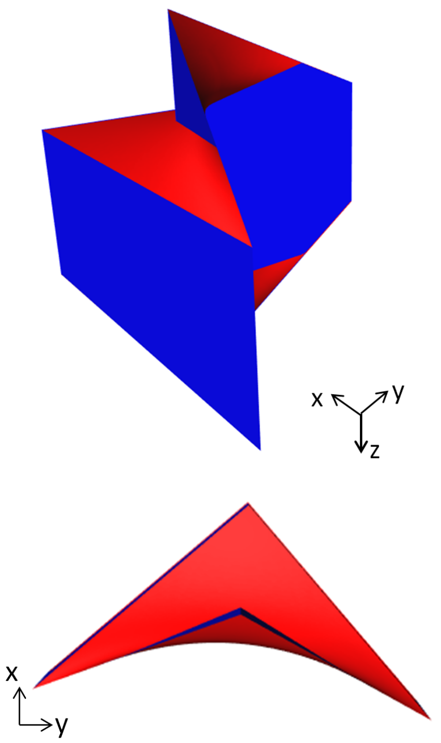

Fig. 8 A single concave grid cell shown with the proposed cell representation seen from the side and above. The top and bottom faces are coloured red, while the lateral cell faces are coloured blue.

4 Discussions

4.1 Nondegeneracy of the grid cells

The conditions for nondegeneracy for a grid cell is discussed in [4]. With the proposed representation it is sufficient that all the horizontal intersections yield convex quadrilaterals, and that the corner points of the cell are well-ordered, as discussed in Sec. 2.2. This is also in general true for the traditional representation, and all cell geometries that are degenerated using the proposed representation will also be degenerated using the traditional representation.

The intersections need to be convex since due to the bilinear representation used for the local u and v some u's and v's will give points that are outside the cell for a concave grid cell. The resulting top and bottom surfaces of the cell will then go outside the column defined by the grid pillars as seen in Fig. 8. For very thin concave grid cells the calculated volume might become negative, both when using the classical representation and the proposed new representation. The best way to prevent these problems is to make sure that only convex grid columns are generated when building the simulation grid, for example by creating regularized near-orthogonal grids. However, often grids where the cell faces closely follow the fault traces are preferred, something that can result in grids with numerous concave cells.

5 Conclusions

We have presented a novel way of representing the cell faces in a corner point grid. Although the representation is slightly more complex than the purely bilinear representation traditionally used, it has significant benefits. The main property of the presented cell face representation is that any horizontal intersection through the grid will give a lattice of quadrilaterals. This ensures that there are no gaps between cells or overlapping cells and hence that the volume of the entire grid is equal to the sum of the volumes of the individual grid cells. The quadrilateral lattices also make it much easier and faster to find the grid cell containing a given point, a common operation used in well blocking. Cell volumes and transmissibilities are found using the same approach as with the traditional representation, using a set of local coordinates. The use of a single parameterization for a whole grid column also makes it possible to do the volume calculations for a whole grid column at a time. The proposed parameterization could, if implemented in a reservoir simulator, give a slight improvement in the consistency of the results of the simulations, especially considering that the simulation grid would become dense with regard to the volumes. The downside is that the calculation of grid cell volumes and transmissibilities would take more time, but these calculations are done only once for each simulation run. For consistency, we recommend that the same grid cell representation is used throughout the whole workflow, both while populating the grid and in the flow simulation.

6 Acknowledgements

The writing of this paper was funded by the Research Council of Norway. The authors want to thank Tor Barkve at Roxar for valuable input to the paper. The authors also want to thank Erik Makino Bakken for help with calculations and testing of alternative representations for the top and bottom cell surfaces.

A Appendix

The mapping from local coordinates $(u, v, w) \in [0, 1]^3$ to global coordinates is given by

$$\begin{aligned}
x = f(u,v,w) = (1-v)\big((1-u)(x_1 + \frac{h(u,v,w)-z_1}{z_5-z_1}(x_5-x_1)) + u(x_2 + \frac{h(u,v,w)-z_2}{z_6-z_2}(x_6-x_2))\big) + \\
v\big((1-u)(x_3 + \frac{h(u,v,w)-z_3}{z_7-z_3}(x_7-x_3)) + u(x_4 + \frac{h(u,v,w)-z_4}{z_8-z_4}(x_8-x_4))\big) \qquad (20)
\end{aligned}$$

$$\begin{aligned}
y = g(u,v,w) = (1-v)\big((1-u)(y_1 + \frac{h(u,v,w)-z_1}{z_5-z_1}(y_5-y_1)) + u(y_2 + \frac{h(u,v,w)-z_2}{z_6-z_2}(y_6-y_2))\big) + \\
v\big((1-u)(y_3 + \frac{h(u,v,w)-z_3}{z_7-z_3}(y_7-y_3)) + u(y_4 + \frac{h(u,v,w)-z_4}{z_8-z_4}(y_8-y_4))\big) \qquad (21)
\end{aligned}$$

$$\begin{aligned}
z = h(u,v,w) = (1-w)\big((1-v)((1-u)z_1 + u\,z_2) + v((1-u)z_3 + u\,z_4)\big) + \\
w\big((1-v)((1-u)z_5 + u\,z_6) + v((1-u)z_7 + u\,z_8)\big). \qquad (22)
\end{aligned}$$

References

1. Aavatsmark, I.: An introduction to multipoint flux approximations for quadrilateral grids. Computational Geosciences **6**(3-4), 405–432 (2002). DOI 10.1023/A:1021291114475
2. Ponting, D.K.: Corner point geometry in reservoir simulation. In: P.R. King (ed.) The Mathematics of Oil Recovery, pp. 45–65. Oxford University Press, USA (1992)
3. Ramsey, S.D., Potter, K., Hansen, C.: Ray bilinear patch intersections. Journal of Graphics Tools **9**(3), 41–47 (2004). DOI 10.1080/10867651.2004.10504896
4. Ushakova, O.: Conditions of nondegeneracy of three-dimensional cells. a formula of a volume of cells. SIAM J. Sci. Comput. **23**(4), 1274–1290 (2001). DOI 10.1137/S1064827500380702