PySubdiv: open-source geological modelling and reconstruction by non-manifold subdivision surfaces

Mohammad Moulaeifard[1], Simon Bernard[1], Florian Wellmann[1]

[1]Computational Geoscience and Reservoir Engineering depratment, RWTH Aachen University, 52062 Aachen, Germany

5   *Correspondence to*: Mohammad Moulaeifard (mohammad.moulaeifard@cgre.rwth-aachen.de)
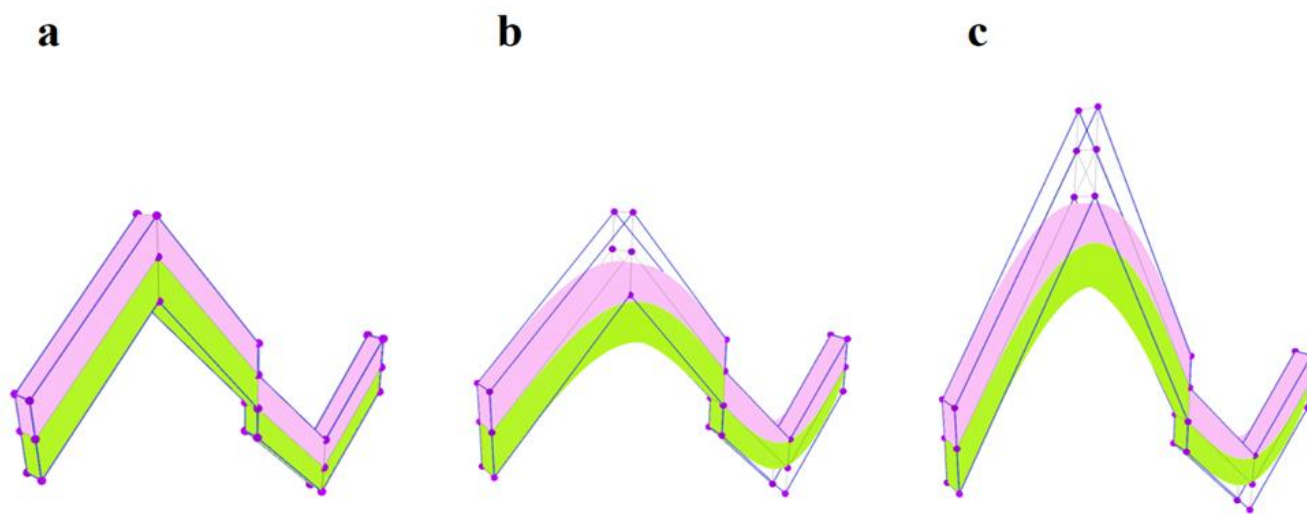
Figure 1: Schematic representation of the procedure to generate the flexible and watertight geological structure with non-manifold topology by PySubdiv; (a) The control mesh with the control points (red) and the edges with different crease sharpness values (blue

10   **and black); (b) Rendered, smooth and watertight final mesh generated by repeatedly applying a non-manifold subdivision surface algorithm; (c) Modified smooth mesh generated by changing the positions of the control points (six red top points).**

Abstract.

Sealed geological models are commonly used as an input to process simulations, for example in hydrogeological or geomechanical studies. Creating these meshes often requires tedious manual work – and it is, therefore, difficult to adjust a

15   once-created model. In this work, we propose a flexible framework to create and interact with geological models using explicit surface representations. The essence of the work lies in the determination of the control mesh and the definition of semi-sharp creases values which, in combination, enable the representation of complex structural settings with a low number of control points. We achieve this flexibility through the adaptation of recent algorithms from the field of computer graphics to the specific requirements of geological modelling, specifically the representation of non-manifold topologies and sharp features.

20 We combine the method with a swarm optimization approach to enable the automatic determination of vertex position and crease sharpness values. The result of this work is implemented in an open-source software (PySubdiv) for reconstructing geological structures while resulting in a model which is (1) sealed/watertight, (2) controllable with a control mesh, and (3) topologically similar to the main geological structure. Also, the reconstructed model may include a fewer number of vertices compared to the input geological structure which results in reducing the cost of modelling and simulation. Furthermore, the

25 proposed method resolves some problems known from spline surfaces. In addition to enabling a manual adjustment of sealed geological models, the algorithm also provides a method for the integration of explicit surface representations in inverse frameworks and the consideration of uncertainties.

1 Introduction

30 Parametric surface-based representation is one of the major approaches to the surface representation of geological objects. Several previous studies considered a surface-based approach in geological modelling since the outstanding features of the structure e.g. heterogeneity are explicitly demonstrated by the surfaces of the boundary (Wellmann et al., 2021; Jacquemyn et al., 2019; Caumon et al., 2009).

NURBS (Non-Uniform Rational B-Splines) and subdivision surfaces are the two major approaches of parametric surface-

35 based representation. In both cases, a 2D parametric domain is employed for the representation of 3D objects (Botsch et al., 2010). Previously published studies of geological modelling have dealt with NURBS (Jacquemyn et al., 2019; Börner et al., 2015; Caumon, 2010; Paluszny et al., 2007). However, NURBS surfaces suffer from two limitations: (1) for generating a model with complex topology, several NURBS patches have to be stitched together which leads to difficulty in modelling and (2) since NURBS surfaces are based on grid structures, a modification of NURBS surfaces is challenging (Botsch et al., 2010;

40 Sederberg et al., 2008; DeRose et al., 1998). In geological modelling, topology alludes to the relationship between different elements of the geological model which is a key constraint for most geological processes, e.g., heat transfer or fluid flow (Thiele et al., 2016).

Although extensive research in the field of surface-based geological modelling has been carried out on exploiting NURBS, researchers have not considered subdivision surfaces in much detail. Both methods have their advantages and disadvantages

45 (see (Wellmann et al., 2021)). In this work, we focus on subdivision surface methods as they (1) overcome some of the limitations of NURBS, (2) provide controllable representations of freeform shapes, and (3) release a model from topological constraints (Cashman, 2010). However, to date, there is no practical software to generate and manipulate sealed geological models using subdivision surface approaches. We attempt to close this gap with the approach described in this paper and the implementation in the accompanying software package PySubdiv.

50   The common uses of subdivision surfaces are in computer gaming and animation (Cashman, 2010; Botsch et al., 2010). This approach converts an initial mesh (control mesh) to a smooth mesh by refinements. In this process, modifications are repeated until the final mesh is sufficiently fine and the final smooth mesh can be controlled easily by the control points.

In complex geological and reservoir modelling, non-manifold topologies are broadly observed, e.g., interactions between faults and horizons, channel intersection, and hierarchical layered structures. Representations of non-manifold structures require

55   complex algorithms (Rossignac and Cardoze, 1999). One of the typical examples of non-manifold surfaces in geological modelling is where several faces of the mesh share one edge (Fig. 2). The lack of supporting non-manifold surfaces is one of the limitations of classical subdivision surfaces. (Ying and Zorin, 2001) modified the subdivision surfaces algorithm and made it compatible with the non-manifold topology which is implemented in the core of PySubdiv.
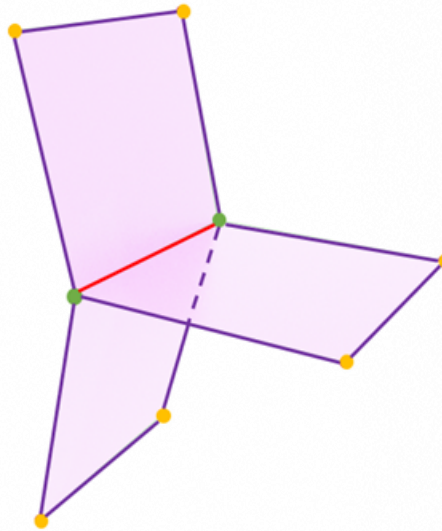
Figure 2: A common example of a non-manifold topology in geological modelling; multiple faces share one edge. Non-manifold vertices (green) and edge (red).

Classic subdivision surface algorithms tend to generate smooth surfaces. However, complex geological structures inherently

65   consist of (semi-)sharp creases and corners which can be generated on the mesh by the resistance of the vertices to the smoothing procedure. Also, the resistance of the vertices can be created by applying a crease sharpness value to the edges consisting of the vertices. The higher the crease sharpness value of the edge, the sharper the creases on the structure. It is worth mentioning that PySubdiv exploits the semi-sharp creases for a better representation of creases of complex geological structures.

70   Subdivision surfaces are used to solve inverse problems by fitting smooth surfaces to a mesh or dense point cloud data (Ma et al., 2015) and since the smooth surfaces are generated based on the control mesh, therefore the generation of a suitable control

mesh is vital in solving the inverse problem (also termed the reconstruction process). The structure of the control mesh is based on two critical variables: (1) the position of the control points and (2) the crease sharpness value of each edge. Therefore, for fitting the smooth mesh to input data, the position of the control points and crease sharpness values can be considered in both

75   *estimation* and *optimization* of the control mesh.

To *estimate* the control mesh, the simplification method is studied by several researchers (Hoppe et al., 1994; Suzuki et al., 1999; Wu et al., 2017). However, this method may raise difficulties in the reconstruction of complex geological structures (Wellmann et al., 2021; Caumon et al., 2004). (Ma et al., 2015) advise exploiting the distinguished features of the input mesh (which is implemented in PySubdiv) instead of the simplification method.

80   Regarding the *optimization* of the control mesh, (Wu et al., 2017) investigated automatic fitting to solve the optimization problem by using the augmented Lagrangian method. They showed that their method provides significant gains over previous works e.g. by (Marinov and Kobbelt, 2005; Hoppe et al., 1994) by generating the reconstructed mesh with fewer control points while consisting of comparable errors. Therefore, for the sake of efficiency, PySubdiv makes use of the automatic reconstruction method proposed by (Wu et al., 2017) which is explained in section 2.4.

85   The final reconstructed structure is controllable by control points, sealed (or watertight), and topologically similar to the input model. This study investigates the advantages and limitations of PySubdiv for modelling and reconstruction of geological and reservoir structures with a case study. Also, PySubdiv can export the final files as the 3D objects based on common object formats e.g. obj, which can be read by most computer graphics and meshing software.

2 Methods

90   The core functionality of PySubdiv consists of four fundamental parts which are investigated in the following section: (1) subdivision surface algorithm, (2) modelling with semi-sharp creases, (3) supporting non-manifold topology, and (4) automatic reconstruction.

2.1 Subdivision surface algorithm

95   The subdivision surface algorithm is based on mathematical rules and frequently modified the input mesh (control mesh) to generate the final desirable mesh. Figs. 1 and 3 schematically show the procedure of converting the control mesh (Figs. 1a and Fig. 3a) to smooth mesh (Fig. 1b and Fig. 3b) by using subdivision surfaces. The vertices of the control mesh (red vertices) are the control points which can modify the final mesh (Fig. 1c and Fig. 3c).

Subdivision surfaces follow two steps at each refinement: (1) *splitting* step, which includes implantation of new vertices on

100   the surface and (2) *averaging* step for updating the location of the vertices. There are several subdivision surface schemes based on different criteria e.g. type of input mesh (triangular or quad) and the approach for refinement. Loop Scheme (Loop,

1987) is one of the common subdivision schemes for triangular meshes which is already implemented in PySubdiv. In the following, the Loop algorithm is explained.

105

change mesh color -

add intermediate subdiv. step?
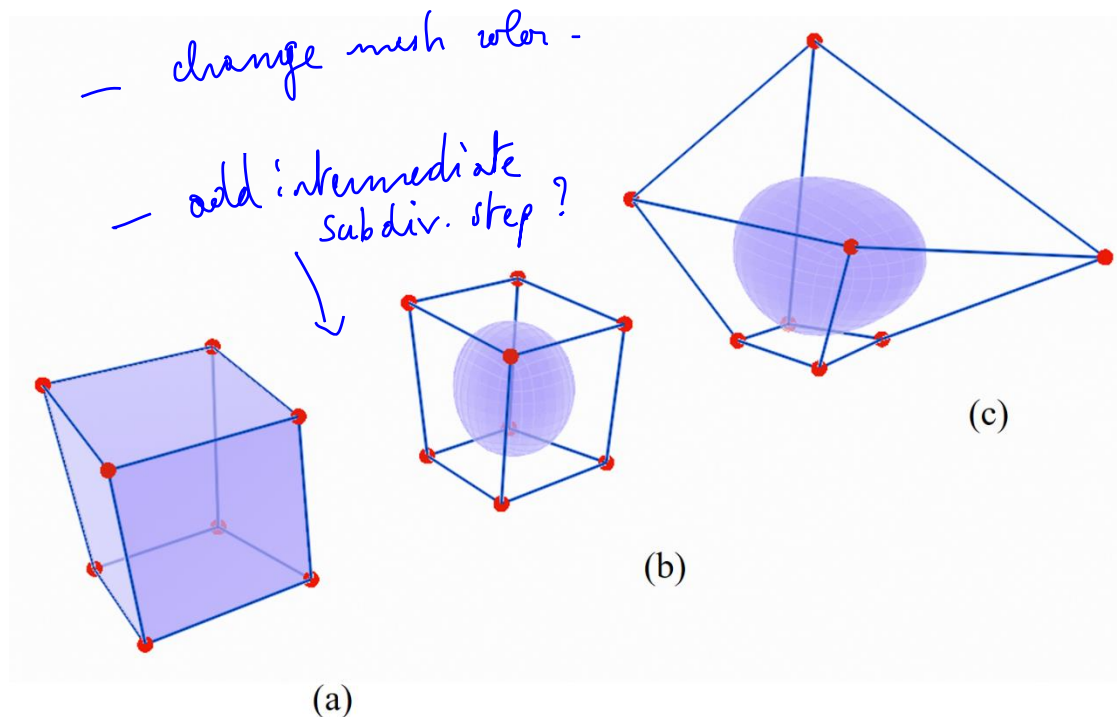
Figure 3: Generation of a smooth mesh by repeatedly applying the subdivision surface algorithm; (a) The input mesh (control mesh); (b) Apply the subdivision surface algorithm to the input mesh three times; (c) Modification of the final mesh by changing the position of the control points.

110

2.1.1 Loop subdivision scheme

Redundant with JTG paper?

(Loop, 1987) defined the loop algorithm to generate smooth surfaces for triangular meshes by using splitting and averaging steps in each refinement stage. In the splitting step, a new vertex is inserted on the midpoint of each edge (blue vertex) which results in the splitting of each triangle of the control mesh into four triangles (Fig. 4).
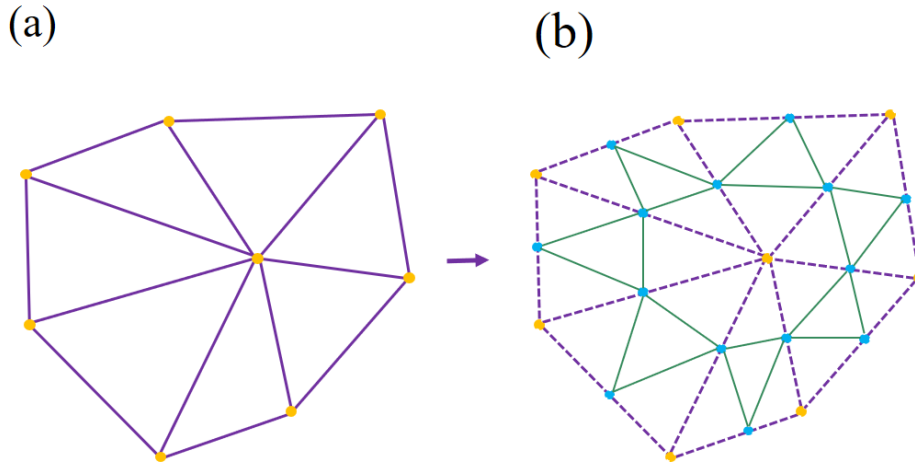
Figure 4: Splitting step of Loop scheme; (a) The control mesh; (b) Splitting each triangle into four by inserting new vertices (red vertices) on the mid of each edge.

Updating the position of the *existing* and *midpoint* vertices (yellow and blue vertices) is the averaging step of the Loop scheme (Fig. 5). To determine the new position of the *existing* vertex (q) with m adjunct vertices ($r_1, r_2, r_3, ..., r_k$), Loop scheme proposes Eq. (1) (Fig. 5a) (Loop, 1987):

$$q_{new} = q * (1 - m\beta) + \beta \sum_1^m r_m, \tag{1}$$

Where

$$\beta = \frac{1}{m}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{m}\right)^2\right), \tag{2}$$

Also to compute the new location of the *midpoint* of an edge (h) enclosed by four existing vertices (e_1, e_2, e_3, e_4) the Loop algorithm advises using Eq. (3) (Fig. 5b) (Loop, 1987):

$$h = \frac{1}{8}(e_1 + e_3) + \frac{3}{8}(e_2 + e_4), \tag{3}$$
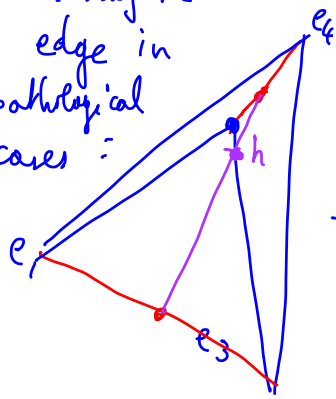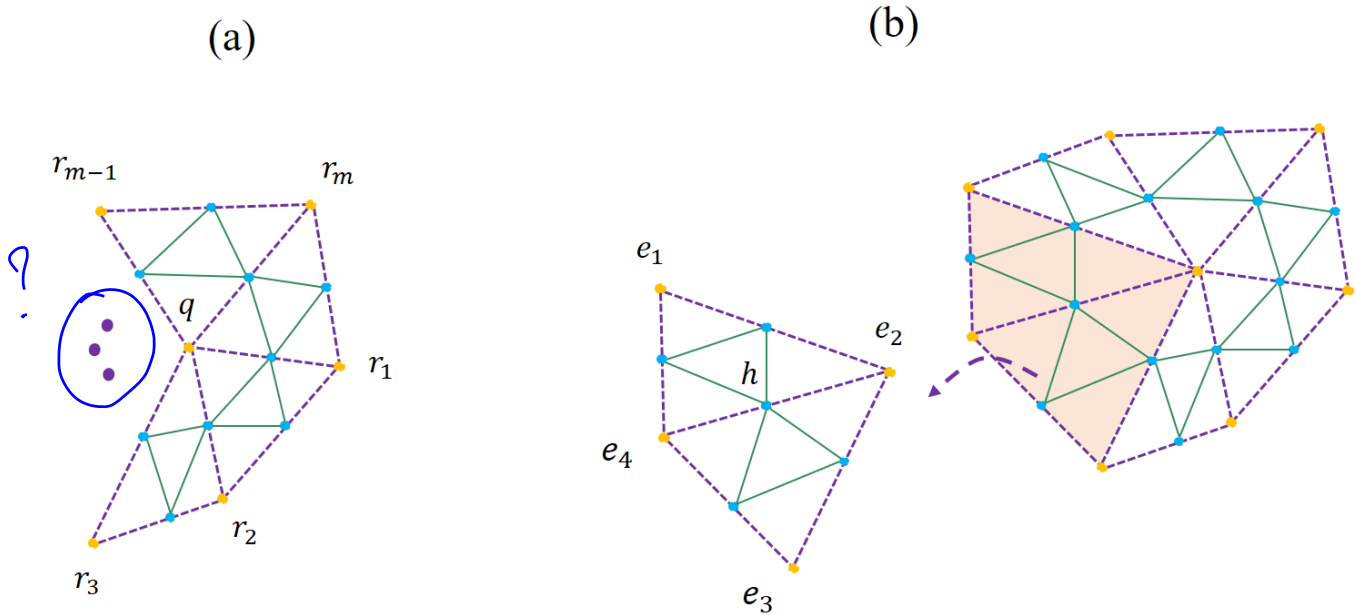
Figure 5: Averaging step of generating the smooth surface by Loop subdivision scheme (a); Updating the position of the existing vertices; (b) Updating the position of the new midpoint vertices.

2.2 Modeling with semi-sharp creases

The subdivision surface approach can generate smooth mesh associated with creases and corners (DeRose et al., 1998). The creases and corners can be made by considering the crease sharpness values for the edges of the control mesh which specifies the resistance of the vertices of respected edges to the smoothing procedure (Fig. 6). This value can be between zero to one which indicates zero and infinite resistance to the subdivision algorithm, respectively. Modelling different geometric objects become more flexible by regulating crease sharpness. Fig. 6a represents the control mesh with eight control points (red vertices). Also, Fig. 6b shows the final smooth mesh after applying three times subdivision surfaces when all edges of the control mesh have no crease sharpness values. However, Fig. 6c represents the effect of the resistance of three sharp edges to the smoothing procedure (black edges, with crease sharpness values equal to one).

show geol example (kink fold?).

quads or tri?

the
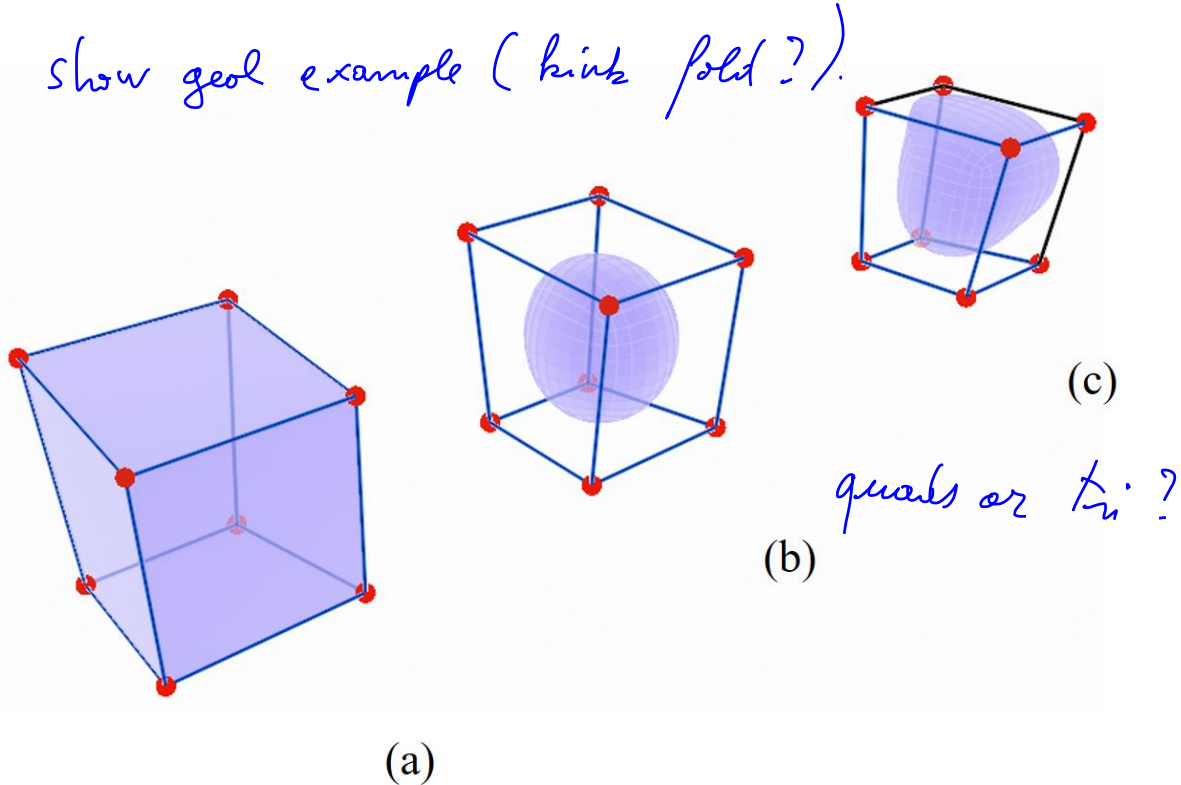
Figure 6: Generating creases on a mesh by applying three times *the* **subdivision surface algorithm; (a) Control mesh (b); All edges of the control mesh are smooth edges (blue edges); (c) Three edges are crease edges (black edges), and nine edges are smooth (blue edges).**

but it has been extended

define.

2.3 Supporting non-manifold topology

but it has been extended

are

Dealing with non-manifold topologies is extensively observed in complex geological and reservoir modelling. The subdivision surfaces cannot support non-manifold topology since there is at least one irregular vertex or/and edge. (Ying and Zorin, 2001) proposed the non-manifold subdivision surface algorithm which supports an extensive variety of non-manifold topology problems. Fig. 2 represents the surface intersections as a common example of non-manifold topology in geological modelling e.g. intersection of different faults or intersections between a horizon and a fault. PySubdiv exploits the non-manifold subdivision surfaces for geological modelling. A full explanation of this method is beyond the scope of this paper. However, (Wellmann et al., 2021) extensively investigated non-manifold subdivision surfaces for geological and reservoir modelling with multiple examples which can be a reference for detailed explanations of this method.

Moulaeifard ?!

could be removed.

2.4 Automatic reconstruction

165   The goal of reconstruction is to generate a suitable control mesh for solving the inverse problem. The reconstructed mesh is
generated by applying the subdivision surface approach to the control mesh and fitting the smooth mesh to the input mesh.
The fitting process can be perfume manually or automatically. (Wellmann et al., 2021) explained step by step manual
reconstruction algorithm by the non-manifold subdivision surface method with the geological examples which is a time-
consuming process.

170   (Wu et al., 2017) investigated the automatic reconstruction by solving the optimization problem. In their method positions of
the control points and crease sharpness value of each edge repeatedly changes until the best fit is achieved. However, they
used the simplification method for the generation of control mesh which brings up challenges in geological reconstruction e.g.
surface intersection (Wellmann et al., 2021; Caumon et al., 2004).

In both manual and automatic methods, the first step is the estimation of the control points (c) with outstanding features of the

175   input geological mesh e.g. local maxima, and minima. PySubdiv proposes some critical points of the input mesh as suitable
candidates for the generation of control mesh. However, it is always possible for the user to select the control points arbitrary.
Then PySubdiv used the Delaunay triangulation approach for generating the initial control mesh.

The second step is the reconstruction through optimizing the location of control points and crease sharpness value for each
edge of the control mesh simultaneously, to find the best fit of the reconstructed mesh to the input mesh. Given the input

180   geological mesh consists of t vertices $q = \{q_1, q_2, q_3, .., q_t\}$, the control mesh consists of n vertices $c = \{c_1, c_2, c_3, .., c_n\}$ and
m edges with the crease sharpness values $h = \{h_1, h_2, h_3, .., h_m\}$. Hoppe et al, (1994) mentioned that each of the vertices of
subdivided mesh can be written as an *affine combination* of control points(c). Therefore the positions of the vertices of the
subdivided mesh can be written as $f(h)\, c$ and the non-linear optimization problem can be represented by Eq. (4):

$$\min\{\|q - f(h)\, c\|\},\qquad(4)$$

185   Wu et al., (2017) proposed to turn Eq. (4) into a constrained optimization problem and then solved it by using the augmented
Lagrangian approach. This approach is applied in PySubdiv and explained in the following.
With considering $k = q - f(h)\, c$, Eq. (4) converts to a constrained problem. Therefore, the optimization problem will be:

$$\min\{\|k\|\},\qquad(5)$$

Subject to $k - (q - f(h)\, c) = 0,$

190   (Wu et al., 2017) offered the following augmented Lagrangian function for solving the Eq. (5):

$$L(c, h, k; \lambda) = \|k\| + \, <\lambda, (k - (q - f(h)\, c) > \, + \frac{b}{2}\|k - (q - f(h)\, c\|^2,\qquad(6)$$

Where λ is the Lagrangian multiplier, $<,>$ is the vector dot product operator and the b is a positive number. It should be
mentioned that the last term of Eq. (6), is the penalty term.

Based on the augmented Lagrangian approach, we have to solve the Eq. (6) by considering $\nabla^L_{c,h,k} = 0$ which means $\nabla^L_c = 0$, $\nabla^L_h =$

195   0, and $\nabla^L_k = 0$. Therefore, the problem converted to three subproblems respecting c, h, k.

This is the main contrib. Stren it.

(1) Subproblem with respect to c: the position of the control points c can be calculated in each iteration of optimization by solving the Eq. (7) which is the linear problem:

$$\min{}_p < \lambda, (k - (q - f(h)\,c) > + \tfrac{b}{2}\|k - (q - f(h)\,c\|^2, \tag{7}$$

200

(2) Subproblem with respect to h: the crease sharpness h in each iteration can be captured by solving Eq. (8) which cannot be converted to the linear system and requires particle swarm optimization (James V. Miranda, 2018; Slowik, 2011):

$$\min{}_h < \lambda, (k - (q - f(h)\,c) > + \tfrac{b}{2}\|k - (q - f(h)\,c\|^2, \tag{8}$$

Not mandatory.

(3) Finally, the Subproblem with respect to the parameter k: the parameter k can be calculated in each iteration by solving Eq. (9):

205

$$\min{}_z \|k\| + < \lambda, (k - (q - f(h)\,c) > + \tfrac{b}{2}\|k - (q - f(h)\,c\|^2, \tag{9}$$

Equation (9) has a closed-form, as well:

$$k = \max\{0, 1 - \tfrac{1}{b\|y\|}\}y, \tag{10}$$

Where the $y = q - f(h)\,c - \tfrac{\lambda}{b},$

210 **3 CORE of PySubdiv** *) Merge algo with 2.4.*

As mentioned in section 2, the core functionality of PySubdiv includes (1) subdivision surface algorithm, (2) modelling with semi-sharp creases, (3) supporting non-manifold topology, and (4) automatic reconstruction. PySubdiv is written in the object-oriented approach by using Python programming language (Rossum and Boer, 1991). Also, PySubdiv exploits different varieties of open-source external libraries which are integrated into the core. Table (1) represented the main external libraries

215 implemented in PySubdiv.

Table 1

The main external libraries implemented in PySubdiv.

	Library	Explanation
1	PyVista	Interactive 3D graphics application programming interface (Sullivan and Kaszynski, 2019)
2	Numpy	Well-organize numerical computations (Van Der Walt et al., 2011)
3	PySwarms	Swarm optimization (James V. Miranda, 2018)
4	Scipy	Scientific computing (Virtanen et al., 2021)

220 The following section explains the workflow for the reconstruction of geological and reservoir structures by PySubdiv and Fig. 7 shows the graph of the stages.

Pre-reconstruction: preparing the input mesh: PySubdiv accepts the structures with triangle mesh while the input mesh can be generated by any arbitrary method (e.g. implicit or explicit method). Also, the input mesh can be either watertight or non-watertight. If the input mesh is not triangulated, PySubdiv can convert the input mesh to the triangle mesh by using the triangulate function of the PyVista library (Sullivan and Kaszynski, 2019).

Stage 1: estimation of the control points: the estimation of the position and number of control points is the key stage in reconstruction which leads to **setting up** the watertight modelling by providing the control points at surface intersections. PySubdiv offers some candidates to the user based on the features of the input geological structure e.g. minima, maxima, and boundaries. However, it is always possible for the user to select or generate the control points based on the requirements for the interpretation or topological limitations. For example, the geological models generated based on the real data may be associated with uncertainties (Wellmann and Caumon, 2018). Therefore, the user can consider some control points at the **suspicious locations** regardless the locations are on the boundary or body part of the layer.

Stage 2: assigning the crease-sharpness value: the assignment of the crease sharpness value to each edge of the control mesh can be done either manually by the user or automatically. PySubdiv automatically proposes the value for each edge based on the angle between the normals of the faces sharing the edge.

Stage 3: assuming the initial values for Lagrangian multiplayer and the parameter k: the initial values for the Lagrangian multiplayer (λ^0) and parameter k (k^0) should be specified by the user which can be zero at the first (before updating).

Stage 4: calculation of new control points, new crease sharpness values, and new parameter k: in this stage the new control points (c^*), new crease sharpness values (h^*), and new parameter k (k^*) computed by using the augmented lagrangian method (equation 7, 8, and 10, respectively). Since the calculation of h^* is not a linear problem and requires particle swarm optimization, PySubdiv exploits the PySwarms external library (James V. Miranda, 2018) for the sake of efficiency.

Stage 5: checking the satisfaction criteria: after calculation of c^*, h^*, and k^* the algorithm should satisfy the following criteria based on the desired error selected by the user (ϵ) (Wu et al., 2017).

$$\epsilon < \|c^* - c^{*-1}\|^2, \tag{11}$$

Where the c^* is the new calculated control points and c^{*-1} is the control point of the previous step. If Eq. (11) is satisfied, therefore the **final control mesh** which consists of c^*, h^* (as the control points and crease sharpness) will be exported. Otherwise, the lagrangian multiplayer (λ^*) should be updated based on Eq. (12).

$$\lambda^* = \lambda^{*-1} + b * (k - (q - f(h)\,c)), \tag{12}$$

Afterwards, new control points (c^*), new crease sharpness values (h^*), and new parameter k (k^*) should be calculated based on the new lagrangian multiplayer (stage 4). Finally, the reconstructed mesh can be built by applying the subdivision surface algorithm to the final control mesh.
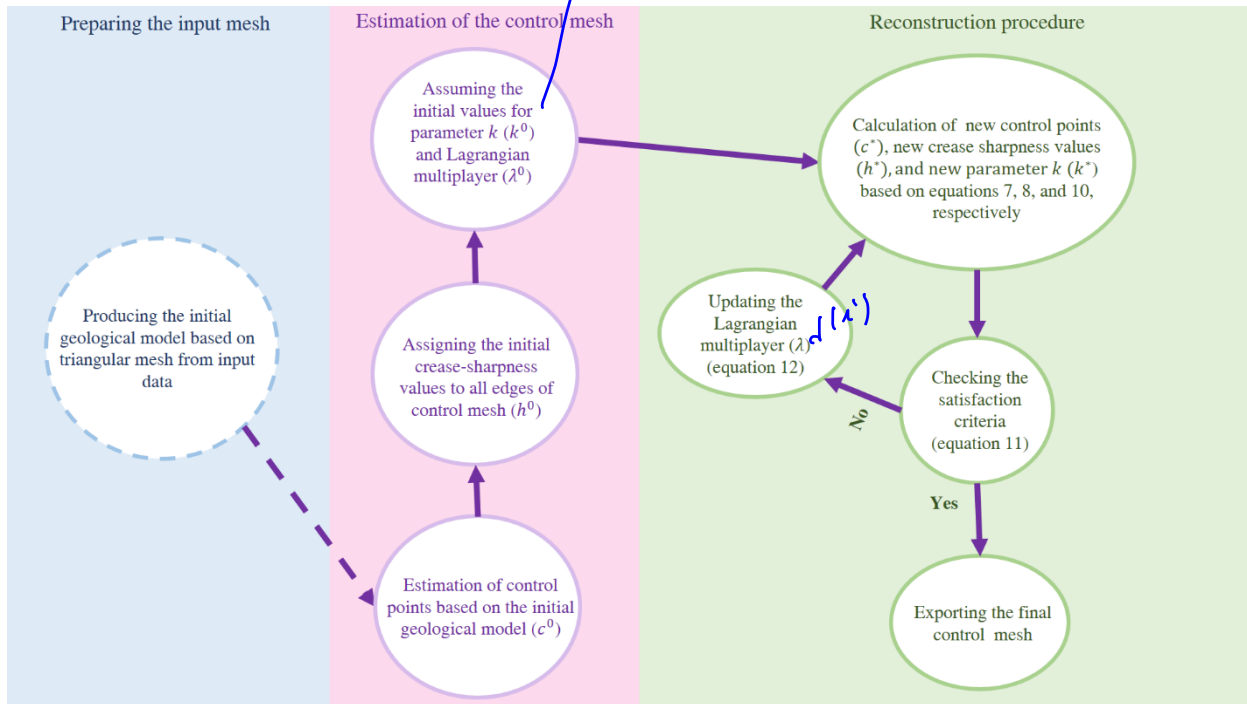
Figure 7: The graph of the stages of PySubdiv for the reconstruction of geological structures.

255

4. Case study

To demonstrate the workflow, part of Upper Rhine Graben (URG) is reconstructed by PySubdiv. URG is a long geological structure in the central part of the European Cenozoic rift system that contains geothermal energy resources. The data set of
260   the URG consist of several different geological units (grid nodes) published by (Freymark et al., 2020) which consists of 616464 individual nodes (Fig. 8). The dimensions of the original model are 292 km in the x-direction, 525 km in the y-direction and 130 km deep (z-direction).

As mentioned in section 3, the input data of PySubdiv should consist of the triangular mesh which can be generated by any arbitrary method or software. In this case study, the PyVista (Sullivan and Kaszynski, 2019) is used to generate the triangular
265   mesh from the individual grid nodes which results in 18 different surfaces and 10 volumes.

| Rhenohercynian | Northern Phyllite Zone | Mid-German Crystalline High | Odenwald | Saxothuringian |
| Alps | Moldanubian | Lower Crystalline Crust | Lithospheric Mantle 1 | Lithospheric Mantle 2 |
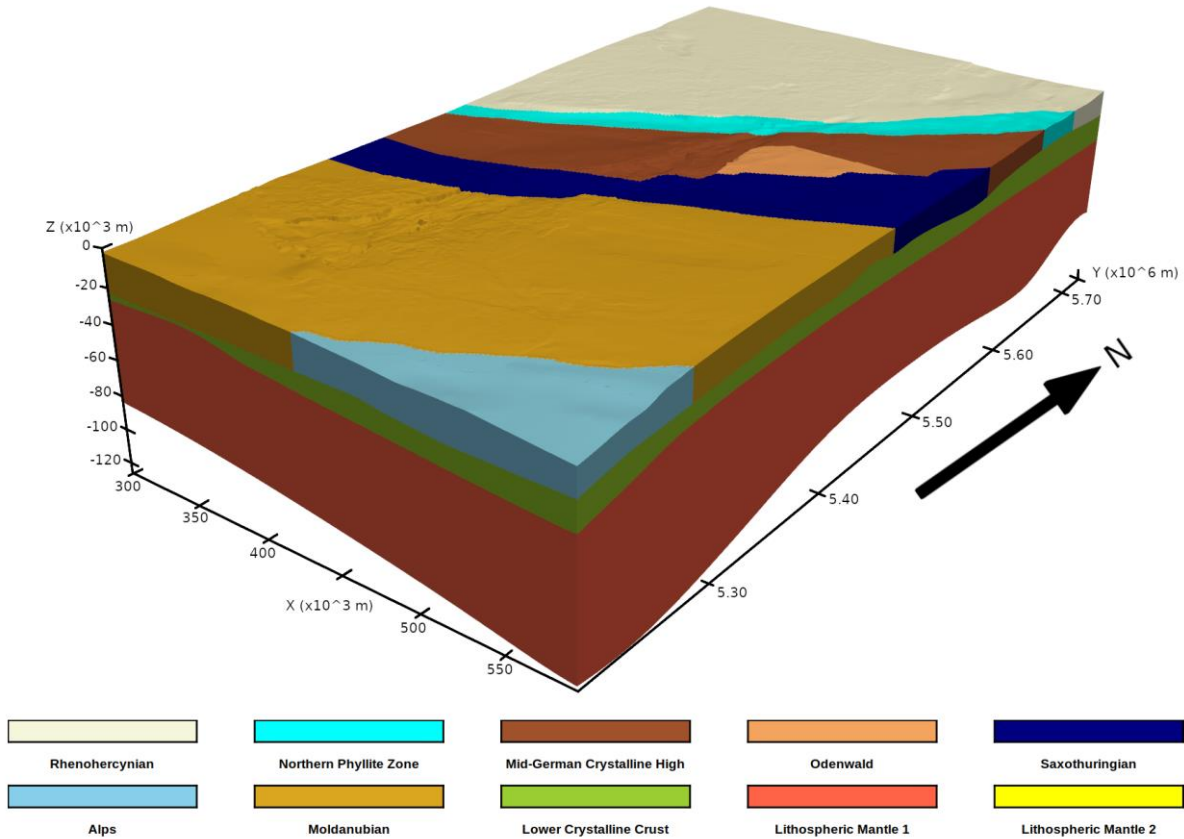
Figure 8: Grid-based representation of part of URG model which contains 616464 individual nodes and 10 different volumes based on the data of Freymark et al., (2020).

270

but units seem oblique to the graben ?

4.1 Estimation of the control mesh

The initial watertight control mesh is prepared based on the prominent features of the input mesh such as (1) minimal and maximal points, (2) distinct points on the actual graben geometry and (3) surface intersections of different layers to ensure that the final mesh is watertight. Fig. 9 represents the control mesh consisting of 832 control points distributed over 18 individual

275   surfaces.

Also, the edges of the control mesh which consist of a threshold angle of 80° (between normal of adjacent faces), are considered sharp and given the crease sharpness values equal to one. Other edges are considered smooth and assigned crease sharpness values equal to zero. From our experience, exploiting the angle of 80° as the threshold angle in this case study leads to acceptable results. However, this angle can be different depending on the complexity of the model.
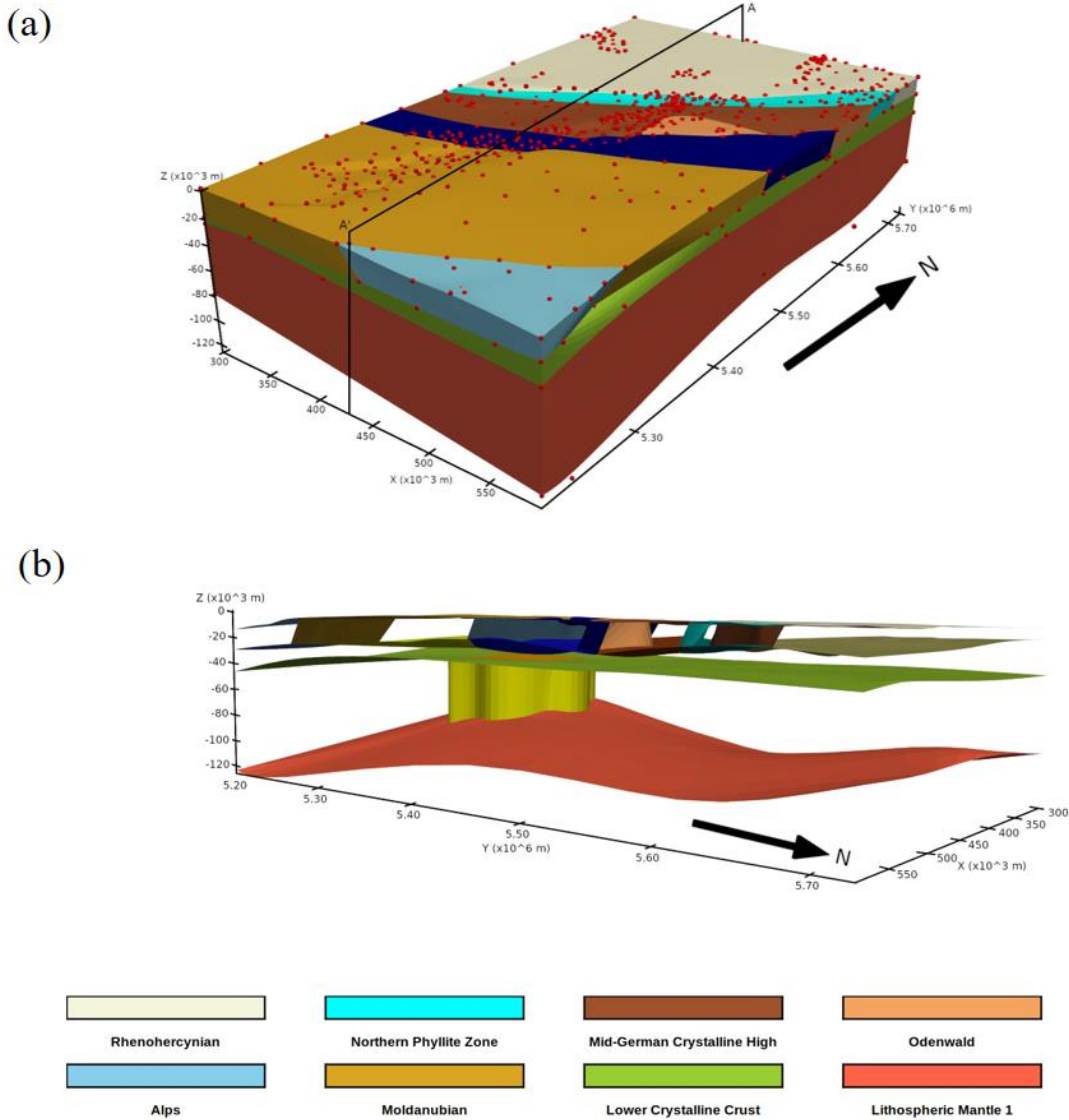
Figure 9: (a) Initial and unfitted control mesh of the URG model. Surfaces are coloured concerning the different geological units. The surfaces of the Lower Crystalline Crust and the Lithosphere mantel are hidden by the boundary surface. The 832 control points are coloured in red; (b) Approximately representation of the cross-section of the model along the profile AA′.

syntax. Cross-section ≠ 3D side view.

4.2 Optimization of the control mesh

The reconstructed mesh is generated after applying two ~~times~~ subdivision surfaces to the control mesh (Fig. 10a). It consists

290   of roughly 15000 vertices. ~~Also~~ To evaluate the reconstructed structure, we calculated the distance metric between the points

of the original and the reconstruction model (Fig. 10b). Red-coloured areas imply high deviations from the subdivision surface

towards the original model. The mean error for the whole domain is approximately 496 m ± 362 m which is around 0.6% of

the total elevation height. Areas of high error concentrate mainly in regions where two individual geological units are connected

and further on the lower boundary of the Lithospheric Mantle (lower boundaries of Fig. 9b). On the Lithospheric Mantle, the

295   highest errors are up to 5700 m. However, regarding the large scale of the model, this deviation is acceptable and merely 4.75

% of the total elevation which is approximately 120 km. Most parts of the model consist of small deviation indicated by the

dark blue colour, especially the higher elevated layers are well-fitted.

How much did the optim move the points ?
(or what were the initial errors ?)

300

Figure 10: (a) Watertight representation of the fitted subdivision surface of the URG. The final mesh is refined two times, yielding roughly 15000 vertices. Control points are displayed as red spheres; (b) The distance map of the subdivision surface. Red areas indicate high deviation towards the original input mesh while blue areas indicate low deviation.

[handwritten annotations:] Difficult to see. use light grey mesh! Compute on initial mesh?

compute how?

305

5 Discussion

The following section discusses the advantages and limitations of PySubdiv in complex geological reconstruction.

5.1 User and PySubdiv interaction

PySubdiv provides a computational framework to generate meshes with limited user interaction. The control points play a key
310 role in estimating, generating and exploiting the reconstruction mesh. The position and number of the control points can be estimated based on three major criteria; (1) the goal of the user for reconstruction e.g. uncertainty analysis of the whole or specific part of the structure, (2) salient features of the input mesh e.g. maxima, minima, and points with high curvature, and (3) topological limitations e.g. surface intersections.

The goal of the user for reconstruction e.g. uncertainty analysis of the specific part of the structure, cannot be automatically
315 recognized by the PySubdiv. Therefore, the user is asked to interact with the software over the GUI to indicate the desired important locations. However, salient features of the input mesh (e.g. maxima, minima) and topological limitations (e.g. surface boundaries and intersections) can be recognized automatically by PySubdiv and proposed to the user. After selecting the control points by the user, the initial estimation of the control mesh will be generated.

It should be mentioned that although extensive research has been carried out on using the simplification of the input mesh for
320 generating the control mesh (Hoppe et al., 1994; Suzuki et al., 1999; Kanai, 2001), simplifications can be challenging when modelling and reconstructing complex structures (Ma et al., 2015; Caumon et al., 2004). More recent arguments against using simplification for geological reconstructions have been summarised in our previous work (Wellmann et al., 2021).

5.2 Number of control points

325 The number of control points plays a key role in geological reconstruction. Although considering that a small number of control points may not satisfy the reconstruction goals, carefulness in increasing the number of control points will also increase the computational cost of the reconstruction and ultimately may not be necessary since PySubdiv exploits semi-sharp creases for preserving the details of the structure.

The utilization of semi-sharp creases may reduce the requirements of adding control points for generating sharp or semi-sharp
330 parts of the structure. For example, (Marinov and Kobbelt, 2005) proposed the reconstruction of the surfaces based on smooth subdivision rules and used parameter correction for preserving the geometrical details. However, (Wu et al., 2017) who exploit semi-sharp creases in the reconstruction, mentioned that their method generates the control mesh containing a fewer number of control points compared to the method of (Marinov and Kobbelt, 2005) while having a comparable error.

5.3 External libraries limitation

From the computational point of view, the major goal of PySubdiv is the calculation of the new location for the vertices based on the subdivision algorithm. Therefore, NumPy (Van Der Walt et al., 2011) and Scipy (Virtanen et al., 2021) libraries play key roles in the core of the software which can be the source of problems when the input mesh is large e.g. limited memory. Although the structure of the PySubdiv is planned to avoid the generation of unnecessary big matrixes, the initial input data can also help to tackle this problem. For example, the suitable number of iterations of the subdivision is one of the key inputs data in reconstruction. Applying a small number of iterations, e.g. one iteration in a one-time subdivision procedure, usually cannot guarantee the acceptable generation of smooth and semi-sharp parts of a reconstructed mesh. However, applying a large number of iteration, not only does not make significant differences in the smoothness of the reconstructed mesh but also, remarkably increase the unnecessary calculation costs since more iteration means dealing with more vertices. From our experience, in practice for geological structures, applying two-three times subdivisions may lead to generally acceptable results and the level of subdivisions, therefore, depends on the subsequent application of the model.

5.4 Subdivision surfaces or spline surfaces e.g. NURBS for complex geological reconstruction

Both subdivision surfaces and spline surfaces e.g. NURBS have been studied for fitting purposes in several previous works (Ma et al., 2004; Lavou'e et al., 2007). In contrast to subdivision surfaces, spline surfaces have regular structures that make it difficult for the control mesh to adapt to the local shape complexity (Marinov and Kobbelt, 2005). In addition, most existing reconstruction methods which exploit spline surfaces are only applicable for structures with simple topology (Ma et al., 2015). From the geological modelling point of view, subdivision surfaces provide significant gains over spline surfaces e.g. NURBS by (1) increasing the accuracy of modelling surface intersections e.g. between faults and horizons, and (2) supporting geological structures with complex topology, and (3) better management of the control points since subdivision surfaces, unlike spline surfaces, do not require grid-structures, which can lead to problems in the connect of multiple patches (Wellmann et al., 2021).

6 Conclusion

This study illustrated the framework (PySubdiv) to generate suitable control meshes and fitted reconstructed meshes for complex geological structures and reservoir models based on the non-manifold subdivision surface algorithm. The reconstructed mesh is watertight and topologically similar to the input mesh. Also, the control mesh consists of those control points which play a key role in the flexibility and management of the reconstructed mesh. Subdivision surfaces, unlike spline surfaces, support arbitrary topology which gives more freedom to the user during generating the control mesh. The fitted models are useful for solving inverse problems and can reduce the number of vertices which results in reducing the cost of modelling and optimization.

Code availability

PySubdiv, is a free, open-source Python library licensed under the GNU Lesser General Public License v3.0 (GPLv3). It is hosted on the GitHub repository https://github.com/SimBe-hub/PySubdiv (https://doi.org/10.5281/zenodo.6878051)

370 **Author Contributions**

MM and **FW** contributed to project conceptualization and method development. **SB** wrote and maintained the code with the help of **MM**. Also, **SB** was involved in visualizing the results. **MM** prepared the manuscript with the contributions of both co-authors. **FW** provided overall project supervision and funding.

375 **Acknowledgements**

We would like to thank our colleagues for their support. This project has received funding from the European Institute of Innovation and Technology— Raw Materials—under grant agreements No 16258 and No 19004. We thank the developers of the Blender project for the amazing open-source computer graphics software (http://www.blender.org/), which we used for rendering several figures of this paper.

380 **References**

Börner, J. H., Bär, M., and Spitzer, K.: Electromagnetic methods for exploration and monitoring of enhanced geothermal systems - A virtual experiment, Geothermics, 55, 78–87, https://doi.org/10.1016/j.geothermics.2015.01.011, 2015.

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Levy, B.: Polygon Mesh Processing, CRC press, https://doi.org/10.1201/b10688, 2010.

385 Cashman, T. J.: NURBS-compatible subdivision surfaces, 2010. ✓ Incomplete !

Caumon, G.: Towards stochastic time-varying geological modeling, Mathematical Geosciences, 42, 555–569, https://doi.org/10.1007/s11004-010-9280-y, 2010.

Caumon, G., Lepage, F., Sword, C. H., and Mallet, J. L.: Building and editing a sealed geological model, Mathematical Geology, 36, 405–424, https://doi.org/10.1023/B:MATG.0000029297.18098.8a, 2004.

390 Caumon, G., Collon-Drouaillet, P., Le Carlier De Veslud, C., Viseur, S., and Sausse, J.: Surface-based 3D modeling of geological structures, Mathematical Geosciences, 41, 927–945, https://doi.org/10.1007/s11004-009-9244-2, 2009.

DeRose, T., Kass, M., and Truong, T.: Subdivision surfaces in character animation, in: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1998, 85–94, https://doi.org/10.1145/280814.280826, 1998.

395 Freymark, J., Bott, J., Scheck-Wenderoth, M., Bär, K., Stiller, M., Fritsche, J.-G., Kracht, M., and Gomez Dacal, M. L.: 3D-URG: 3D gravity constrained structural model of the Upper Rhine Graben, 2020.

Hoppe, H., Derose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J., and Stuetzle, W.: Piecewise smooth

surface reconstruction, in: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1994, 295–302, https://doi.org/10.1145/192161.192233, 1994.

400 Jacquemyn, C., Jackson, M. D., and Hampson, G. J.: Surface-Based Geological Reservoir Modelling Using Grid-Free NURBS Curves and Surfaces, Mathematical Geosciences, 51, 1–28, https://doi.org/10.1007/s11004-018-9764-8, 2019.

James V. Miranda, L.: PySwarms: a research toolkit for Particle Swarm Optimization in Python, The Journal of Open Source Software, 3, 433, https://doi.org/10.21105/joss.00433, 2018.

Kanai, T.: MeshToSS: Converting Subdivision Surfaces from Dense Meshes, in: Proceedings of the Vision Modeling and
405 Visualization Conference 2001, 325–332, 2001.

Loop, C.: Smooth Subdivision Surfaces Based on Triangles, Acm Siggraph, 74, 1987.

Ma, X., Keates, S., Jiang, Y., and Kosinka, J.: Subdivision surface fitting to a dense mesh using ridges and umbilics, Computer Aided Geometric Design, 32, 5–21, https://doi.org/10.1016/j.cagd.2014.10.001, 2015.

Marinov, M. and Kobbelt, L.: Optimization methods for scattered data approximation with subdivision surfaces, Graphical
410 Models, 67, 452–473, https://doi.org/10.1016/j.gmod.2005.01.003, 2005.

Paluszny, A., Matthäi, S. K., and Hohmeyer, M.: Hybrid finite element-finite volume discretization of complex geologic structures and a new simulation workflow demonstrated on fractured rocks, Geofluids, 7, 186–208, https://doi.org/10.1111/j.1468-8123.2007.00180.x, 2007.

Rossignac, J. and Cardoze, D.: Matchmaker: Manifold BReps for non-manifold r-sets, in: Proceedings of the Symposium on
415 Solid Modeling and Applications, 31–41, 1999.

Rossum, G. van and Boer, J. de: Interactively Testing Remote Servers Using the Python Programming Language, CWI Quarterly, 4, 283–303, 1991.

Sederberg, T. W., Thomas Finnigan, G., Li, X., Lirp, H., and Ipson, H.: Watertight trimmed NURBS, SIGGRAPH'08: International Conference on Computer Graphics and Interactive Techniques, ACM SIGGRAPH 2008 Papers 2008, 27, 1–8,
420 https://doi.org/10.1145/1399504.1360678, 2008.

Slowik, A.: Particle Swarm Optimization, in: The Industrial Electronics Handbook - Five Volume Set, 1942–1948, https://doi.org/10.1007/978-3-319-46173-1_2, 2011.

Sullivan, C. and Kaszynski, A.: PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK), Journal of Open Source Software, 4, 1450, https://doi.org/10.21105/joss.01450, 2019.

425 Suzuki, H., Takeuchi, S., Kanai, T., and Kimura, F.: Subdivision surface fitting to a range of points, in: Proceedings - 7th Pacific Conference on Computer Graphics and Applications, Pacific Graphics 1999, 168–178, https://doi.org/10.1109/PCCGA.1999.803359, 1999.

Thiele, S. T., Jessell, M. W., Lindsay, M., Wellmann, J. F., and Pakyuz-Charrier, E.: The topology of geology 2: Topological uncertainty, Journal of Structural Geology, 91, 74–87, https://doi.org/10.1016/j.jsg.2016.08.010, 2016.

430 Virtanen, P., Gommers, R., Burovski, E., Oliphant, T. E., Weckesser, W., Cournapeau, D., Peterson, P., Reddy, T., Wilson, J., and Haberland, M.: scipy/scipy: SciPy 1.5. 3, Zenodo, 2021.

Mallet 2022 (handwritten annotation)

Van Der Walt, S., Colbert, S. C., and Varoquaux, G.: The NumPy array: A structure for efficient numerical computation, Computing in Science and Engineering, 13, 22–30, https://doi.org/10.1109/MCSE.2011.37, 2011.

Wellmann, F. and Caumon, G.: 3-D Structural geological models: Concepts, methods, and uncertainties, in: Advances in
435 Geophysics, vol. 59, Elsevier, 1–121, https://doi.org/10.1016/bs.agph.2018.09.001, 2018.

Wellmann, F., de la Varga, M., and Bommes, D.: Subdivide and Conquer: Adapting Non-Manifold Subdivision Surfaces Method to Represent and Approximate Complex Geological and Reservoir Structures, 2021.

Wu, X., Zheng, J., Cai, Y., and Li, H.: Variational reconstruction using subdivision surfaces with continuous sharpness control, Computational Visual Media, 3, 217–228, https://doi.org/10.1007/s41095-017-0088-2, 2017.

440 Ying, L. and Zorin, D.: Nonmanifold subdivision, in: Proceedings of the IEEE Visualization Conference, 325–331, https://doi.org/10.1109/visual.2001.964528, 2001.

445