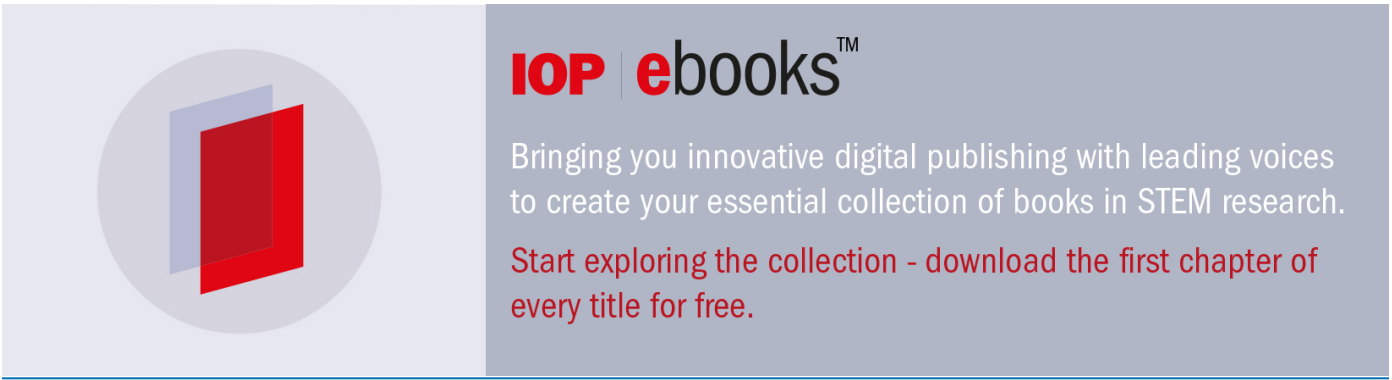
IOP Conference Series: Materials Science and Engineering

PAPER • OPEN ACCESS

Characterisation and modelling of complex textile geometries using TexGen

To cite this article: L P Brown *et al* 2018 *IOP Conf. Ser.: Mater. Sci. Eng.* **406** 012024



IOP Conf. Series: Materials Science and Engineering **406** (2018) 012024 doi:10.1088/1757-899X/406/1/012024

Characterisation and modelling of complex textile geometries using TexGen

L P Brown*, A Endruweit, A Long, I A Jones

Composites Research Group, Faculty of Engineering, University of Nottingham

* louise.brown@nottingham.ac.uk

Abstract. TexGen is open source software developed at the University of Nottingham for the geometric 3D modelling of textiles and textile composites. It has a large number of users worldwide and underpins a significant number of research publications.
While many users make simplifying assumptions about the structure of a textile, in reality the internal geometry of a textile or textile composite is complex. Capturing this complexity is vital for the prediction of properties such as permeability and mechanical failure. Examples will be given of the characterisation of a material and how the complex features are captured and implemented in TexGen, making use of functionality such as the ability to vary the cross-sectional shape along the length of a yarn. The effect on prediction of properties as a model is refined will be demonstrated.
Recent additions to the software will also be highlighted. Laminated structures can be quickly and easily constructed from a selection of textiles and several nesting options are available. A new rotate textile option can then be used to create laminates with varying ply angles. Where the unit cell is also rotated, appropriate periodic boundary conditions have been implemented and are automatically generated in an ABAQUS input file.
A new feature is described which generates a TexGen model from a weave pattern file. Future developments of this may improve accessibility of the software to the weaving community. The generation of a pattern draft output from the TexGen model is also described.

1. Introduction

Textile composites are used in a wide range of applications, either in the form of laminates from 2D textiles, 3D textiles or increasingly complex net-shaped preforms. In order to speed up the design cycle for these materials it is essential to be able to create realistic geometric models so that accurate simulation results can be obtained for the prediction of properties such as permeability and mechanical properties. TexGen [1, 2] is open source software developed in the Composites Research Group at the University of Nottingham, enabling the creation of 3D geometric models of textiles and textile composites. There is built-in capability for the creation of standard textile forms, both 2D and 3D woven, and there are refinement algorithms built into the software for automatic reduction of intersections in the model and for creating different levels of compaction. This automation makes use of the versatility of the built-in modelling capability and can be recreated using the functions in the Python scripting API. This functionality is often overlooked by users and this paper seeks to highlight how this can be used to create more realistic textile models and how this can improve the final simulation results which use these models.

New features are also described, including an option to build up laminates using a variety of textile layers at varying rotations, and new input from weave pattern data as well as its output from a TexGen model.

2. Geometric modelling

The functionality used within TexGen to enable the creation and refinement of complex textile models is highlighted here and guidance is given on how these methods can be used for geometry model generation. The example used here is a 3D orthogonal weave shown in Figure 1a [3]. An idealized model of the geometry can be created using the 3D wizard in the TexGen graphical user interface (GUI), Figure 1b, in this case using the measured yarn widths and heights and a default power ellipse with a power of 0.6 as the constant cross-section. Nodes are created on a grid with one at each crossing of the warp and weft yarns. The binder yarns have extra nodes inserted around the top and bottom weft yarns in order to follow their shape, shown in Figure 1c. This uses a function ShapeBinderYarns which automatically calculates the yarn positions.

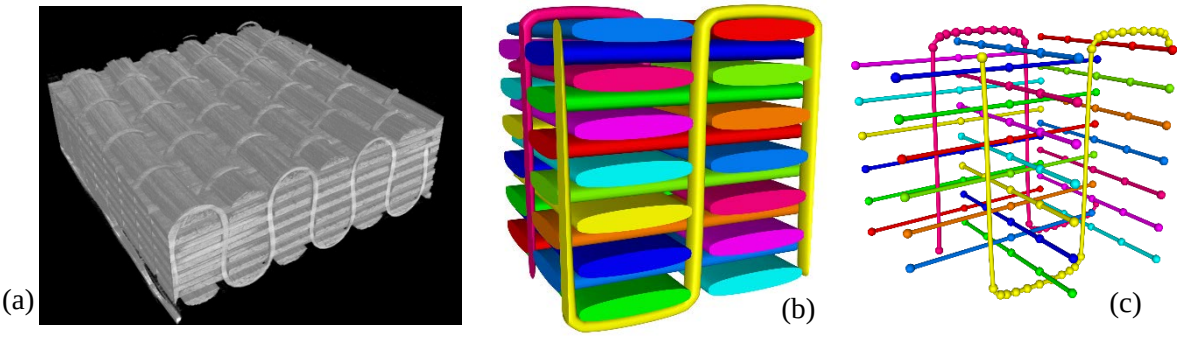


Figure 1. a) µCT image of 3D orthogonal textile b) Idealised model created using TexGen 3D wizard c) Nodes and paths in idealised model

2.1. Use of complex cross-sections

Figure 2a shows a slice from the µCT image of the orthogonal textile. It can be seen that the weft cross-sections, seen end-on in the image, show two distinct shapes. Where the binder yarns wrap around the wefts at the top and bottom of the textile the wefts form a curved section on the outer surface whilst remaining flat where they lie next to the warp yarn underneath. The weft yarns in the inner section of the textile form into a more rectangular shape which can be represented using a power ellipse section.

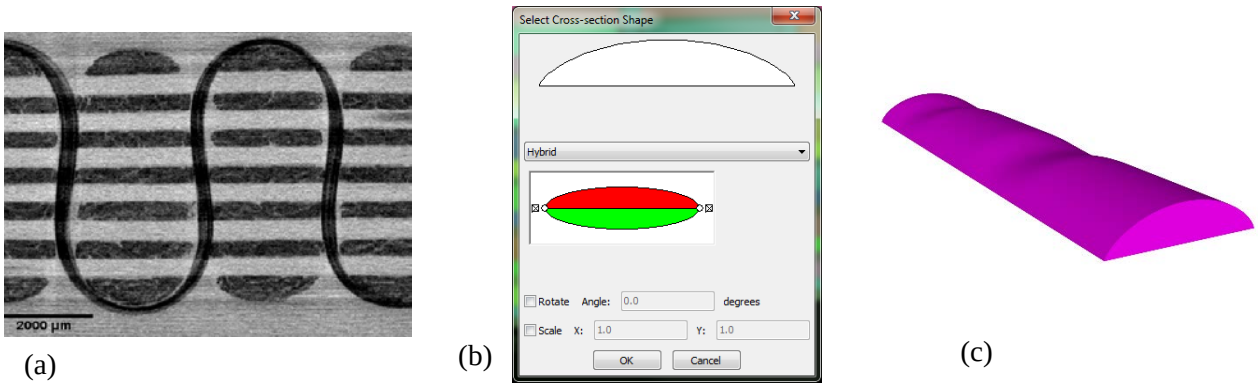


Figure 2. a) µCT image showing weft cross-sections b) Hybrid section dialog c) Yarn with hybrid sections

2.1.1. Hybrid sections. The asymmetric outer sections can be created using the hybrid section type. From the GUI the dialog in Figure 2b allows the shape to be divided into sections, here showing two but more can be chosen if desired. The size and section type of each division can then be entered to form the combined shape. Using Python the CSectionHybrid class is used followed by AddDivision and AssignSection to specify how the section is divided and which section type and size is assigned to each division. An example yarn using this cross-section is shown in Figure 2c.

2.1.2. Polygon sections. In some cases the required cross-section may not conform to a standard shape, or may be difficult to assemble as a composite shape using the hybrid option. In this case the CSectionPolygon class can be used. This option is only available via the Python scripting interface. In the case of the 3D orthogonal textile, as the textile is compacted crimp is induced in the top weft yarns which in turn affects the shape of the warp yarns underneath as shown in Figure 3a. During the refinement process implemented automatically in TexGen this effect is captured, Figure 3b, and the warp yarns shape is automatically changed to avoid intersections. The polygon section is used thus enabling sections such as that shown in Figure 3c to be produced. When the section is initialised a vector of points is specified which define the outer surface of the boundary.

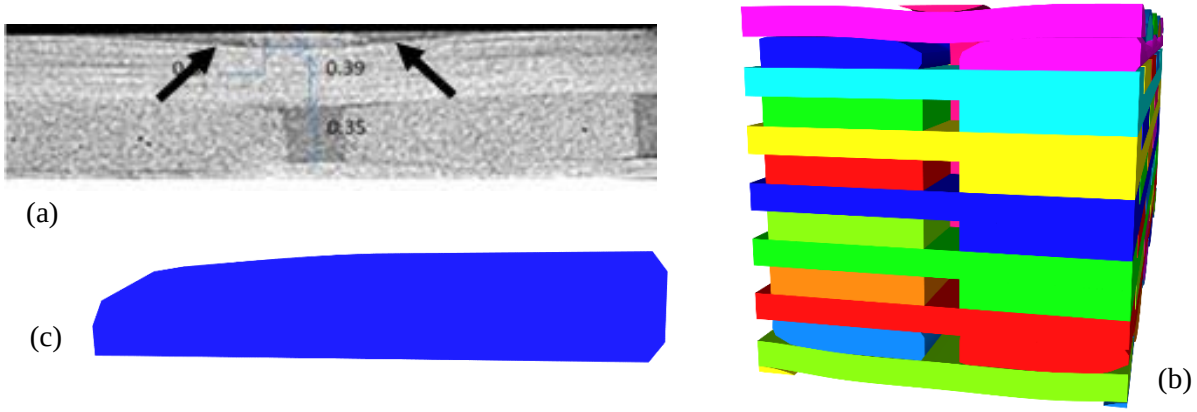


Figure 3. a) µCT image showing local geometry variations b) Refined geometry for compacted textile c) Polygon section

2.2. Variation of cross-section along length of yarn

When a yarn is created it will be assigned a constant cross-section by default using the CYarnSectionConstant class. In reality yarns will deform along their length and a realistic model should reflect these changes in cross-section. In TexGen it is possible to specify varying cross-sections along the length of a yarn either by specifying the sections at each node using the CYarnSectionInterpNode class or at specific points along the yarn using the CYarnSectionInterpPosition class. These classes are all inherited from the abstract CYarnSection base class. After these classes have been created sections are added using the AddSection function, either one section for each node or one section at each position as required. Once all of the sections have been added then the yarn section object is assigned to the yarn using the AssignSection function. These can also be selected in the GUI using the Select Yarn Section dialog given by the Modeller->Assign Section option. An example of a yarn with varying cross-sections can be seen in Figure 2c.

2.3. Local checking of volume fraction

It is important when adjusting cross-sections to ensure that the volume fraction within the yarn does not exceed a realistic limit. If yarn properties have been assigned to a yarn then the yarn volume fraction at a given section can be calculated.

$$Volume\ Fraction = \frac{Fibre\ Area}{Section\ Area}$$

where fibre area can be obtained using the yarn GetFibreArea function and the local yarn section can be extracted from the appropriate CYarnSection class described in section 2.2. This can be obtained using the yarn GetYarnSection function and then getting the actual cross-section at the required point using the GetSection function. From this the GetArea function will calculate the section area.

2.4. Permeability prediction

Flow through the 3D orthogonal textile was simulated using Computational Fluid Dynamics (CFD) software in order to determine the textile permeability [3]. A model of an orthogonal weave was created and then meshed using the voxel meshing built into TexGen. Incremental changes were made to the geometric model to assess the sensitivity of the permeability prediction to the level of geometric detail in the model using the techniques described in the previous sections. Changes were made to the binder cross-sections, surface weft yarn crimp and warp yarn cross-sections in successive models. The CFD simulation predictions are plotted with experimental data in Figure 4, demonstrating significant improvement in permeability prediction with inclusion of local variations in the geometric model.

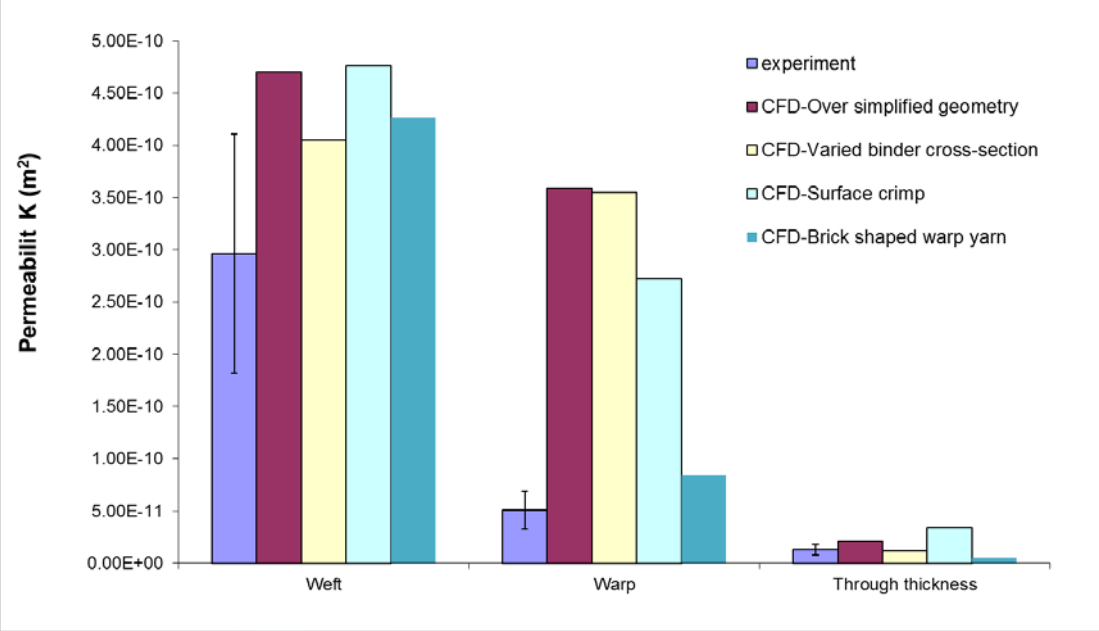


Figure 4. Permeability predictions with variations in model geometry

3. Constructing laminates

It is common for textiles to be built into laminates, often with varying rotations. The CTextileLayered class allows textiles to be combined into a single unit.

3.1. Layered textiles

In the GUI a layered textile can be created from any combination of textiles currently loaded. The Textiles->Create Layered option enables the selection of textiles and the ordering of the layers using the dialog shown in Figure 5a. The domain is created to be the size of the largest of those selected, therefore there may be more than one repeat of weaves with smaller unit cells. By default the textiles are stacked using the sizes of the domains in the z direction to govern the distance between the layers.

An example layered textile is shown in Figure 5b. In a Python script the CTextileLayered class is created and then textiles are added together with their x-y offsets using the AddLayer function.

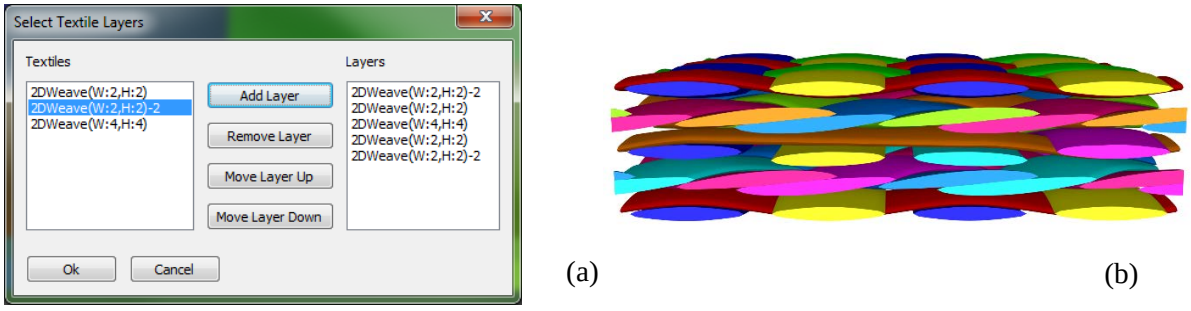


Figure 5. a) Dialog for selection of layers b) Layered textile

3.2. Layer-Offsets and Nesting

Layered textiles are created with an initial offset of zero in the x-y plane when created using the GUI. Using a script offsets can be specified for the individual layers as described in Section 3.1. Using the GUI offsets can either be specified as a constant offset between layers, random or specific offsets specified for each layer. A layered textile with random offsets is shown in Figure 6a.

Nesting of layers is implemented either by keeping the offsets, moving layers as close to each other as possible until the first contact is made between layers, or by finding the maximum nesting. In this case the offset is found between each pair of layers which gives the maximum vertical displacement of the layers towards each other to the point at which they make contact. In this case both the offset in the x-y plane and the vertical displacement are adjusted. Nesting retaining original offsets is shown in Figure 6b and the same layered textile with maximum nesting is shown in Figure 6c.

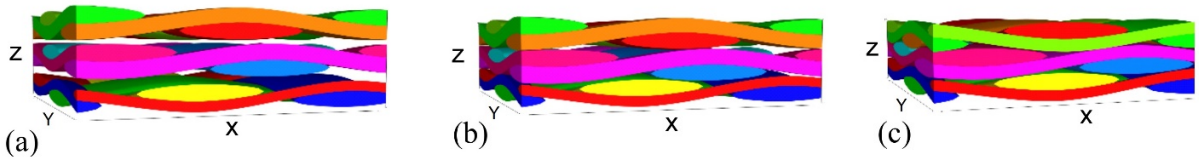


Figure 6. a) Textile with random offset b) Nested with same offset c) Maximum nesting

3.3. Rotated textiles

Textiles can be rotated either using the Textile->Rotate Textile option in the GUI or using the CTextile Rotate function which takes as parameters a quaternion, giving the axis and angle of rotation, and the centre of rotation. In the GUI the axis of rotation is limited to the x, y and z axes, however using the function in a script any axis of rotation can be selected. The domain can also be rotated if desired.
Voxel file export has been modified for the rotated domain so that voxels are generated which are aligned to the domain axes and periodic boundary conditions have been implemented which account for the rotation.

4. Weave pattern input/output

4.1. Weave pattern import

Although it can be straightforward to create a ‘standard’ textile using the TexGen user interface, textile designers often design weave patterns using a weave pattern matrix. The ability to input these and automatically generate the corresponding textile model will speed up the modelling process significantly. In order to interpret a weave pattern in TexGen some extra information about the number of warp yarns in each stack is added as well as a keyword to indicate the type of weave generated.

Generation of general and 3D orthogonal weave patterns were reported in [4, 5]. This has now been extended to include 3D angle interlock weaves.

The weave pattern and layer-ID information are read from a text file in the format shown in Figure 7 using the WeavePattern Python script included in the TexGen installation. This is accessed from the GUI using the File->Import->WeavePatternFile option. The number of warp and weft yarns is extracted and the number of binder yarns deduced. From this the cell structure used to store weave patterns within TexGen is generated.

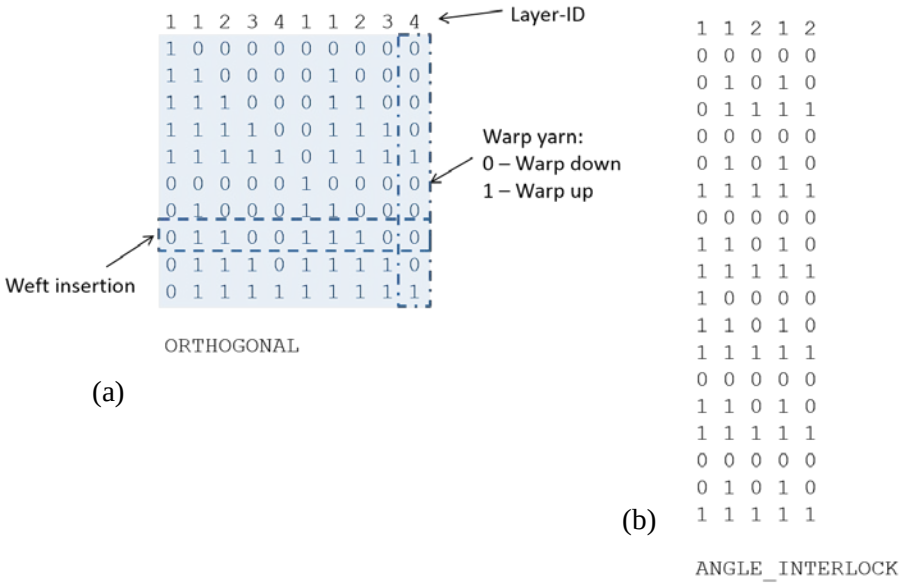


Figure 7. Weave pattern input file format a) Orthogonal weave b) Angle interlock weave

The appropriate class, CTextile3DWeave, CTextileOrthogonal or CTextileAngleInterlock, is used to create a textile each of which contain the cell array which describes the configuration of warp and weft yarns within the textile. The cell array is initialized by processing the weave pattern one row at a time, calling the SetupWeftRow function. This uses the layer-ID information to delimit the binder and warp stack information in the chart as illustrated by the dotted separators in Figure 8. For any stack of warp layers the weft will be positioned where the warp changes from being up to down.

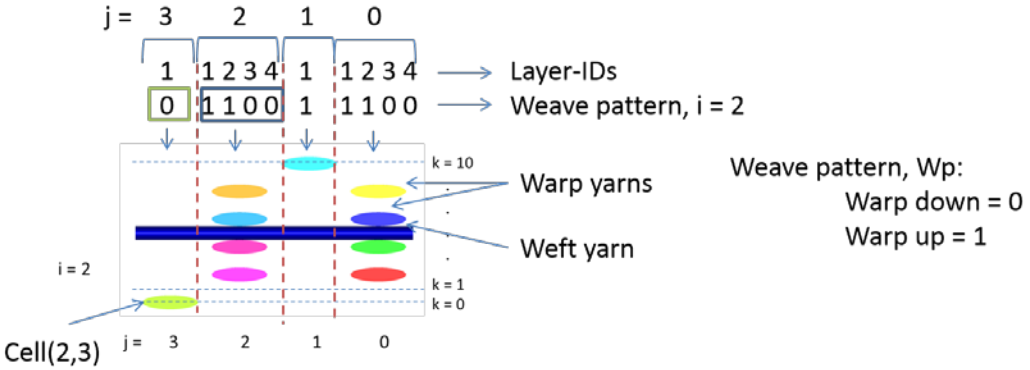


Figure 8. Creation of cell array entries

For an orthogonal weave, for each binder/warp stack, j, the cell array is populated as illustrated in Figure 8 which shows the row for i = 2 as highlighted in Figure 7a. In this case the binder yarn is assumed to be at either the top or bottom of the stack depending on whether the warp (ie binder) is up or down.

The angle interlock case is more complicated as the binder travels through the thickness of the textile at varying weft heights and the correct position needs to be retained in the cell array. In the first instance the binder position is set to one below the weft if the warp is down and to one above the weft if the warp is up. This results in a textile such as that shown in Figure 9a. At this point in the case of both types of weave a textile is formed with the warps stacked but the weft yarns spread out, one yarn for each cell row i.

A subsequent process, ConsolidateCells, moves the weft yarns into stacks where there is an empty cell in the previous row. For angle interlock yarns it can be seen from Figure 9a that the correct binder position for each stack is either where the position is constant between two rows or where it is at the top or bottom. This information is stored in a vector for each binder yarn and then the appropriate binder positions are assigned as the cells are consolidated. The resulting textile is shown in Figure 9b.

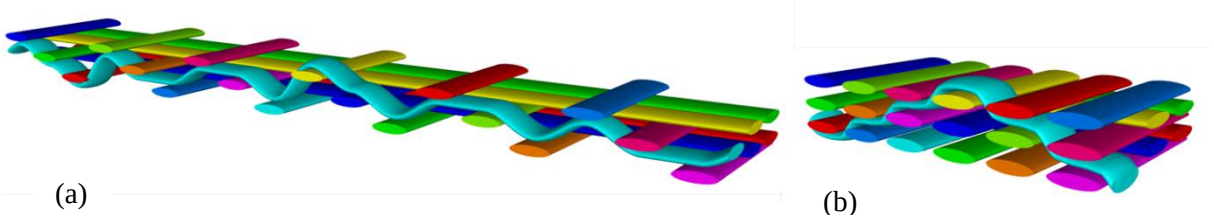


Figure 9. Angle interlock textile a) unconsolidated b) after consolidation

4.2. Weave pattern export

For textiles which use either the 2D or 3D weave base classes the process described in Section 4.1 can be reversed in order to generate a weave pattern matrix from a TexGen model. For 3D textiles the ordering of the weft insertions, either stacking from top to bottom or bottom to top will change the weave pattern generated. In the user interface there is a dialog to select the weft insertion ordering (Figure 10b). From a Python script the ordering can be passed as a parameter to the weave ConvertToPatternDraft function.

A pattern draft is also generated based on a 1:1 tie up. This uses an adaptation of the algorithm defined for hand looms given by Griswald [6] and Glasner [7]. The pattern draft can either be printed or saved as a text file as shown in Figure 10c.

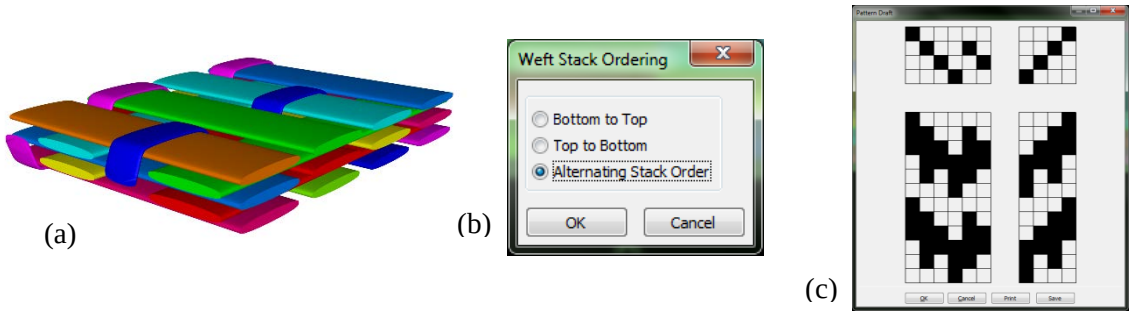


Figure 10. Weave pattern export a) Example weave b) Dialog to select weft stack ordering c) Pattern draft

5. Conclusions

Methods for increasing the accuracy of geometric textile models created using TexGen software have been described. The benefits of improved model accuracy were demonstrated for a case study where in-plane permeability was predicted using CFD simulation with models of increasing geometric accuracy.

Methods to quickly and easily create models of textile laminates using either the TexGen GUI or Python scripting are described. This includes nesting and new options for rotation of layers.

A method for importing a weave pattern matrix and converting it into a TexGen model is described enabling fast creation of models from a standard textile design format. The method has recently been extended to angle interlock textiles and this will be included in the next TexGen release. Creation of pattern drafts from a TexGen model is also described.

Acknowledgements
This work is funded by the Engineering and Physical Sciences Research Council RSE Fellowship [Grant number: EP/N019040/1].

References

[1] Brown, Louise P, & Sherburn, Martin. (2017, December 13). louisepb/TexGen: TexGen v3.10.0 (Version v3.10.0). Zenodo. http://doi.org/10.5281/zenodo.1115604
[2] A C Long and L.P. Brown, *Modelling the geometry of textile reinforcements for composites: TexGen*, in *Composite reinforcements for optimum performance* P. Boisse, Editor. 2011, Woodhead Publishing Ltd.
[3] Xuesen Zeng, Louise P. Brown, Andreas Endruweit, and A.C. Long. *Advanced Geometry Modelling for 3D Woven Fabrics and its Application in Permeability Prediction*. in *The 4th World Conference on 3D Fabrics and Their Applications*. 2012. RWTH Aachen, Germany.
[4] L P Brown, B Wendland, S Schaeffer, A C Long, and T. Gries, *3D-Woven Profile Design Tool for Users with a Non-Textiles Background*, in *10th Int. Conf. on Manufacturing of Advanced Composites (ICMAC 2015)*. 2015: Bristol, UK.
[5] L P Brown, S. Yan., X Zeng, and A.C. Long, *Mesoscale Geometric Modelling of Bifurcation in 3D Woven T-Beam Preforms*, in *12th Int. Conf. on Textile Composites*. 2015: Raleigh, NC, USA.
[6] Griswold, R.E. *Fabric Analysis - From Drawdown to Draft*. 2000 23/09/2014]; Available from: http://www.cs.arizona.edu/patterns/weaving/webdocs/mo/D/FabricAnalysis.pdf.
[7] Glassner, A., *Digital weaving.2*. Computer Graphics and Applications, IEEE, 2003. **23**(1): p. 77-90.