# Visualization of Finite Element Data of a Multi-Phase Concrete Model

M. Ritter, M. Aschaber, W. Benger and G. Hofstetter

2013 (post-print)
ASCE American Society of Civil Engineers
https://ascelibrary.org/doi/abs/10.1061/9780784412992.070

**Abstract**

We present a novel formulation of the data structure needed for finite elements (FE) in a data model based on the mathematics of fiber bundles. This formulation allows integrating of FE data into a context of generic data processing operations, in particular tools for multivariate data visualization. Our approach is showcased using a real-world application dataset describing drying concrete. Total stress, displacement, mean stress and drying shrinkage strain are illustrated, combining direct visualization methods for scalar-, vector- and tensor fields. Eventually this approach allows for new insights into these rather complex and otherwise opaque FE datasets and supports optimizing the underlying simulations code.

## Introduction

Computer simulations based on the finite element method (FEM) are a common tool for analyzes in continuum mechanics. The results of such simulations are space-time dependent scalar, vector and second order tensor fields. Visualization features in FEM computational frameworks or applications, such as ABAQUS are often limited to displaying scalars on geometrical boundaries or cross sections and vector arrows at vertices or integration points. Cutting planes are used to define cross sections. To get an overview of the three dimensional distribution of a scalar field one has to look at different cross sections. No volumetric rendering methods, no splat based rendering, and no integration line based vector field visualization methods are available. The FE simulation used in our work is based on a multi-phase concrete model described in [Valentini12]. In the latter, concrete is modeled as a porous material with the pores filled by liquid water and/or gas. Hence, a multi-phase concrete model consists of the solid phase, the water phase and the gas phase. Macroscopic balance laws, i.e., the mass balances for each phase, the linear momentum balance and the energy balance for the multi-phase material, together with the kinematic relations and the constitutive relations, form the set of governing equations. They describe a fully coupled hygralthermo-mechanical model in terms of the chosen solution variables gas pressure, capillary pressure, displacements and temperature. The multi-phase concrete model has been used particularly to simulate drying shrinkage of concrete in a more realistic way. After the setting of the concrete, commonly, the pore humidity is higher than the ambient relative humidity and, thus, a drying process starts. This drying process decreases the relative pore humidity of concrete and increases the capillary pressure, which exerts a hydrostatic pressure on the cement matrix resulting in volumetric compaction, known as drying shrinkage.

The visualization of FE data is still an ongoing research area. Early work was done on parallel workstation systems, e.g. based on ray-casting [Garrity90] or based on tetrahedral splatting [Williams92]. Other methods are based on particles, iso-surface or cutting planes. FE data is often re-sampled on uniform grids allowing fast and well studied texture based volume rendering techniques [Engel06]. Here, work was done on multivariate techniques, e.g., [Stompel02] also utilizing non-photo realistic techniques. Modern GPU based direct visualization techniques are usually based on ray-casting. [Bock12] was able to shift heavy computational parts into a pre-processing step improving the GPU rendering for interactive performance.

However, most approaches are limited to a certain grid type, do not use ad vanced techniques developed in texture based volume rendering or are not formulated with aspect to multivariate data. We are aiming at usage of a general data model to enable techniques independent of, or applicable to many, grid types. Our data model is motivated and introduced in the next section F5 Data Model. The modeling of FE data is presented and data conversion is described. The section Visualization presents the utilizes visualization techniques, which are applied in section Visualization of a Drying Concrete Specimen. Finally, a Summary is provided.

## F5 Data Model

**Motivation.** In order to be able to best reuse existing visualization techniques, it is necessary to find most general or common solutions for specific problems. The computational and observational sciences nowadays produce more and more datasets of different kinds. Data analysis and visualization plays an important role in the understanding and interpretation of datasets. Errors in simulation codes might not be discovered if not visualized properly. Many different data layouts are used for numerical computations dependent on the applied methods and spatio-temporal discretization: uniform grids, rectilinear grids, curvilinear grids, hexahedral grids, unstructured grids, particles, etc. Already in 1989 [Butler89] proposed to use the mathematical model of fiber bundles as a foundation for a common data model. Using such a systematic model enables to apply visualization technique across a wide range of different data sources.

**Mathematics.** The data model utilized in our work is based on the theory of fiber bundles. This models any data as a total space, which is constructed from a so-called base space and fiber space. While the base-space corresponds to a manifold, the fiber space corresponds to data attached to each of the points of the entity. The discretized base space describes a finite sampling of data points including their (discrete, integer) neighborhood information, whereas the fiber space usually is continuous, containing

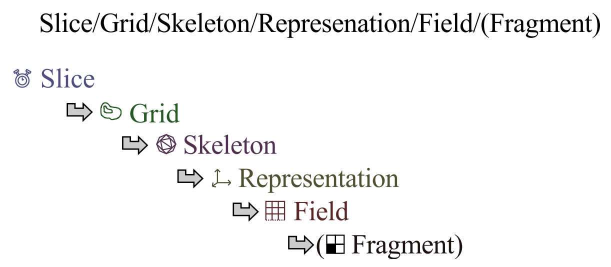


Figure 1: Hierarchical data organization of the F5 data model.

floating-point data such as scalar, vectorial and tensorial quantities. This data model suits very well spatio-temporal data occurring in scientific visualization and covers most numerical simulations. Data structures used for finite elements add a new property to the data though, which are interpolation weights and evaluation points. These are not inherently spatio-temporal, but rather "supplementary information" as required for the numerical simulations. There is no explicit base space associated with these FE data as they describe information between data points. However, the data model used in our work [Ben04] allows to also formulate data given on relationships between base spaces, properties of a base space (i.e, the skeletons of a CW-complex ) and even entirely separate manifolds. This capability serves well to express the properties of a FE data structure in the existing data model without need to modify the data model itself, as will be elaborated in the following sections.

**Hierarchy.** The data model is organized in a hierarchy, see Figure 1. First element is the time level storing time as a double floating point variable, called Slice. Inside a time slice geometric objects are stored, called Grid objects. A grid object is described by at least one topology object, called Skeleton. The properties of a skeleton are defined by the dimensionality of the skeleton (point=0, line=1, surface=2, ...), the refinement level, and the index depth. The index depth is the number of indirections to reach the most basic point skeleton. An edge skeleton would have dimensionality one and index depth one, because an edge is defined as a pair of point indices and, thus, an indirection of one to the points. A line skeleton defined via indices of edges would have a dimensionality of one and an index depth of two. Inside a skeleton object, data can be defined in one or more Representations. Here, certain coordinate systems can be specified, such as 3D-Cartesian or 3D-Spherical. Representations can also be specified relative to another skeleton. For example, if an edge is defined via point indices, an according representation would be located in the edge skeleton called 'EdgeAsPoints', which indicates that the used indices are those of the point skeleton. Numerical data is stored in the Field. A field is a named data array of (compound) elements, e.g. a three dimensional array of second order tensors. A field called 'Positions' describes the geometry of the skeleton. For example, a 'Positions' field in a point skeleton stores the coordinates in the Cartesian representation, a 'Positions' field in the edge skeleton stores the point indices in the EgdesAsPoints representation. All the skeletons in combination with their according 'Positions' fields form the base space, the manifold, of a grid object. Data fields can be added in any representation

Table 1: FE types. The values of columns 3, 4 and 5 uniquely identify the FE type.

| Cell Type | Abaqus Name | Points (Nodes) | Integration Points | Dimens-ionality |
|---|---|---|---|---|
| Hexahedral Linear | C3D8 | 8 | 8 | 3 |
| Hexahedral Quadratic Reduced | C3D20R | 20 | 8 | 3 |
| Tetraeder Linear | C3D4 | 4 | 1 | 3 |
| Quadrilateral Linear | S4 | 4 | 4 | 2 |
| Quadrilateral Linear Reduced | S4R | 4 | 1 | 2 |

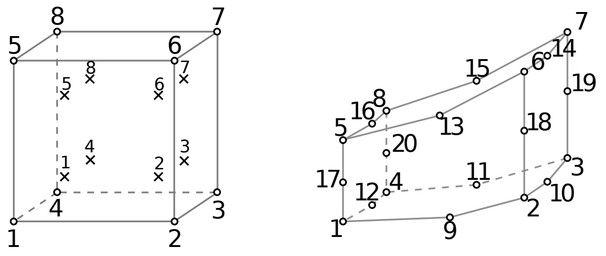


Figure 2: Left: Nodes and integration points of an undeformed C3D8 element. Right: Nodes of a deformed C3D20 element. Element faces and edges are curved.

to any skeleton. For example, a pressure scalar field on the points, a second order tensor field on 3D-cells, a velocity vector field on edges, etc. If a field is separated into multiple blocks or fragments, the Field level is a container for named fragments, which are then the data arrays.

**Modeling FE Data.** A wide range of different types of finite elements are used for numerical simulations. In structural analysis the solutions of a typical simulation are the displacements at the points of the FE mesh, which may be, e.g., composed of hexahedral or tetrahedral cells. The number of points per cell is defined by the shape functions used for data interpolation inside a cell. Within the framework of the FE method the solution variables are computed at the nodes, whereas all other output data, e.g. strain and stress tensors, are computed at the integration points. The number of integration points is defined by the method of integration inside the cell, independent from the interpolation type. Table 1 shows some types of finite elements and lists their number of points, number of integration points and their dimensionality.

*Nodal Data.* Arranging data at points is straightforward in the data model. A list of Cartesian coordinates is stored in a point skeleton as 'Positions' field. The field is a one dimensional array of 3D points. Additional data is stored with an according name in the same coordinate representation, e.g. , the vector field describing the displacement. Figure 3 illustrates the layout of the nodal data of a linear hexahedral mesh including the data fields: displacement and pressure.

*Integration Point Data.* We now model the integration points in a second grid object. The name of the object is the name of the nodal FE-grid concatenated by

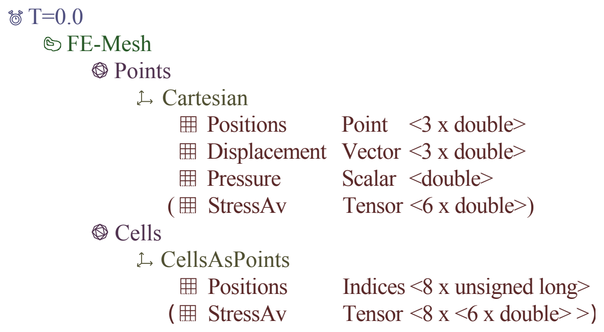

T=0.0
FE-Mesh
Points
Cartesian
Positions Point <3 x double>
Displacement Vector <3 x double>
Pressure Scalar <double>
( StressAv Tensor <6 x double>)
Cells
CellsAsPoints
Positions Indices <8 x unsigned long>
( StressAv Tensor <8 x <6 x double> >)

Figure 3: F5 layout of a C3D8 element's nodal data. Data fields displacement and pressure are stored in the 'Points' skeleton. Cells are defined in another skeleton via indices to the point 'Positions' in the relative representation 'CellsAsPoints'.

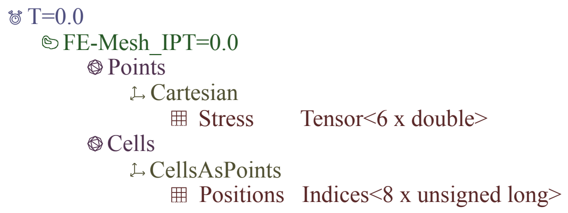

T=0.0
FE-Mesh_IPT=0.0
Points
Cartesian
Stress Tensor<6 x double>
Cells
CellsAsPoints
Positions Indices<8 x unsigned long>

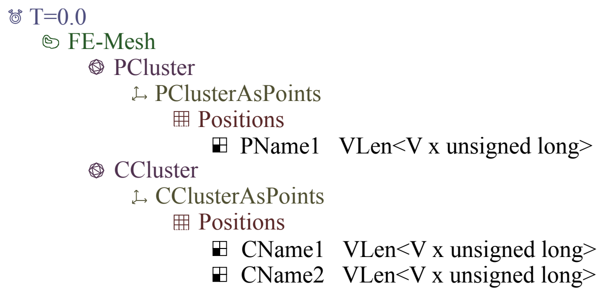
Figure 4: F5 layout of a C3D8 element's integration point data. One exemplary stress data field is stored in the 'Points' skeleton. No 'Points' 'Positions' field is required as integration point coordinates can be computed from the nodal grid.

a standardized postfix. Figure 4 illustrates this second grid. The 'Positions' field of the 'Points' skeleton is not required, since it can be computed from the nodal points and cells. The two 'Cell' topologies in the two grid object share the same index base: same index for the same cell, thus, relating integration points to nodes and vice versa.

*Time Independent Data.* Some data fields are time independent. The cell connectivities need not be stored in every time slice. Also, the node positions can be static and current ones can then be computed from the original coordinates and the current displacement vector. In these cases the data field objects are replaced by pointers to the according fields stored in the first time slice. When working with the hierarchy this data reduction is hidden and the field can be normally accessed at any slice.

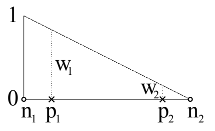
*Averaged and Cell Relative Nodal Data.* For the purpose of visualization, data is mostly requested on nodes because the shape functions require this for data interpolation. Thus, data given on integration points only, have to be recomputed into representative values at the nodes. This is done in two steps: computation of nodal values per cell, and averaging all nodal values per node (arithmetic mean). The per cell data set is stored as a data field in the nodal 'CellsAsPoints' representation. The averaged nodal field is stored in the nodal Cartesian points representation. Figure 3 illustrates the two computed fields 'StressAv' in round brackets.

*Selection Sets.* Sets of elements are used to define properties, such as a materi-



```
T=0.0
    FE-Mesh
        PCluster
            PClusterAsPoints
                Positions
                    PName1   VLen<V x unsigned long>
        CCluster
            CClusterAsPoints
                Positions
                    CName1   VLen<V x unsigned long>
                    CName2   VLen<V x unsigned long>
```

Figure 5: F5 layout of selection sets. One nodal selection set and two elemental selection sets are stored in two optional topologies in a fragmented 'Position' fields.



$$
\begin{aligned}
f_{[ni]} &= \frac{\sum f_{pj} w_j}{\sum w_j} \\
f_{[n1]} &= (f_{[p1]} w_1 + f_{[p2]} w_2)/(w_1 + w_2)
\end{aligned}
$$

Figure 6: Nodal weighting shown at a 1D linear element for the node $n1$.

aproperty. Material might change from one element to the other, e.g. from concrete to steel. When visualizing, averaging over such boundaries is not desired. Thus, selection sets can be stored to group cells. They are stored as lists of indices of nodes or elements in two optional topologies: 'PointClusters' with dimension 0 and index depth 1, and 'CellClusters' with dimension 3 and index depth 2. Figure 5 illustrates these optional topologies. To give each selection a name, each set of indices is stored as a named fragment holding a variable length array of indices, nodes or cells, respectively.

**Data Conversion.** A converter from the ABAQUS file format odb was developed in C++ using the API of ABAQUS and the F5 file library. The library, described in [Ritter09], is a small C library providing most high level functions to read and write F5 data. It is build upon the HDF5 library [HDF-Group], a binary container format, developed by and for the high performance computing community, originating from the National Center for Supercomputing Applications. The converter is a command line tool taking an odb file and some flags as arguments producing a HDF5 file in the F5 layout. Building upon HDF5 guarantees transparent data storage, sustainability, high performance, and easy data exchange independent of closed or non documented formats.

## Visualization

**Visualization Shell.** We extend the visualization shell (VISH) [Benger07] to handle FE data. VISH is an open academic visualization framework developed over multiple platforms in C++. External library dependencies are kept as minimal as possible.

A visualization task is solved by connecting data, computational, and visualization modules in a visualization network, with data modules being sources and visualization modules being sinks. OpenGL including GLSL shaders are used for rendering. VISH uses the fiber bundle data model to manage data and provides basic modules for visualization, e.g., of scalar, vector, and second order tensor fields on uniform, curvilinear and particle grids. We extended the framework to support hexahedral and tetrahedral meshes and the FE data structure described in the previous section. Several derived data fields, such as tensor invariants, are computed on demand. All fields given at integration points can be requested at the nodes, either averaged over cell boundaries or relative to a cell.

**Visualization Techniques.** For extrapolating a value to a node a weighted sum of values at the integration points is used. The weights are values of the shape function at integration points, Figure 6. Interpolation is done via the shape functions. Hexahedral linear and quadratic interpolations were implemented. A CPU parallel computation module that re-samples a FE data field on a uniform grid has been implemented as a pre-step for direct volume rendering of hexahedral and unstructured meshes.

*Solid Cell Rendering.* A computational module was developed to display the FE geometry and, optionally, one scalar field via color map. FE data is transformed into a triangular surface grid allowing the reuse of existing surface visualization modules. Cells can be created as solid cells or as cages. Parameters are a scaling factor to locally scale down an element and the width of the cage bar. Figure 7 illustrates solid rendering of a hexahedral mesh of $9 \times 13 \times 9$ linear cells.

*Volume Rendering.* Re-sampled FE data on a uniform grid can be displayed via the existing texture based volume rendering module. This technique uses the texture unit of the GPU to combine many slices through a 3D volume texture (uniform grid) [Engel06]. Figures 7 and 9 illustrate the technique applied to a $70 \times 70 \times 25$ uniform grid. The transparent volume rendering can be combined with the solid cage allowing to display two different scalar fields. Two scalar fields can also be visualized by dual volume rendering using one scalar field for coloring and the other for transparency scaling (iso surfaces) [Benger12].

*Tensor Splats.* The tensor splatting technique stems from the direct visualization of the space curvature tensor in numerical relativity and/or the diffusion tensor of magnet resonance imaging. Dominant directions of the tensors, Eigenvectors, are visualized. The shape factors, computed from the Eigenvalues, of the tensor are used to adjust color and texture of the splats [Benger04]. Green color and fiber texture represents a tensor having positive Eigenvalues with one being dominant, and red color and flat texture represent a tensor having positive Eigenvalues with two being dominant. When three Eigenvalues are dominant, the isotropic case, splats are faded to full transparency. Splats are oriented according to their Eigenvectors. This technique was enhanced to also display tensors with at least one negative Eigenvalue in blue color.

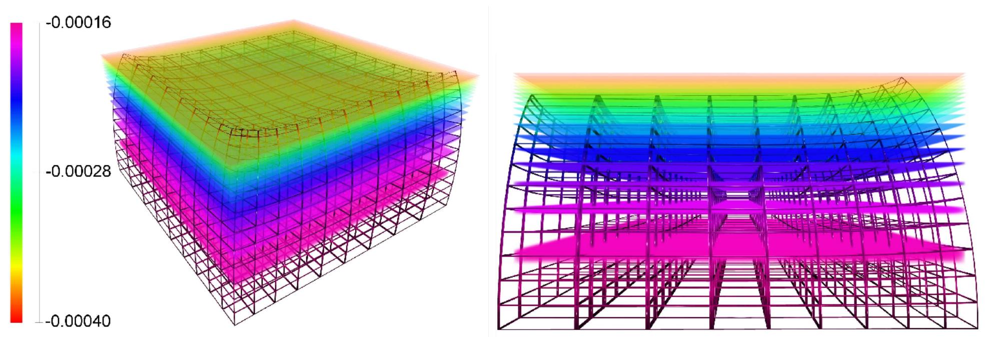


Figure 7: Cages and volume rendering of the concrete specimen after 30 days of drying showing the drying shrinkage strains $\epsilon_{sh}$ illustrated by 18 transparent iso surfaces and color on the cages. Cages are deformed by the displacement field, scaled by a factor of 1000 and rescaled into the undeformed bounding box. For clarity, volume visualization is done on the undeformed grid.

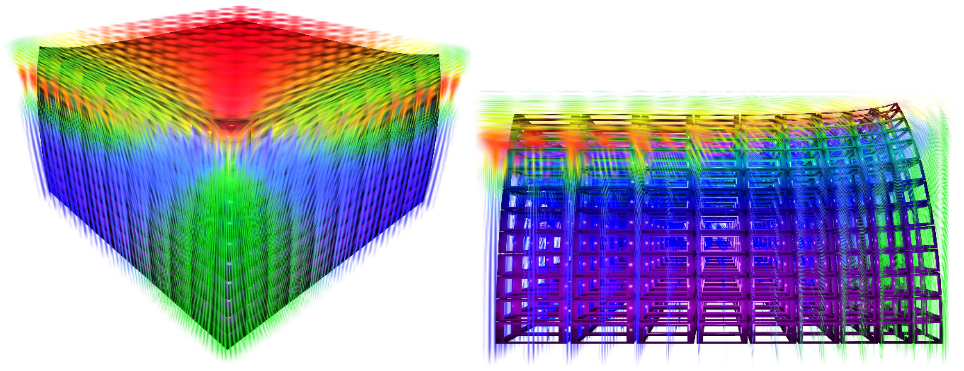
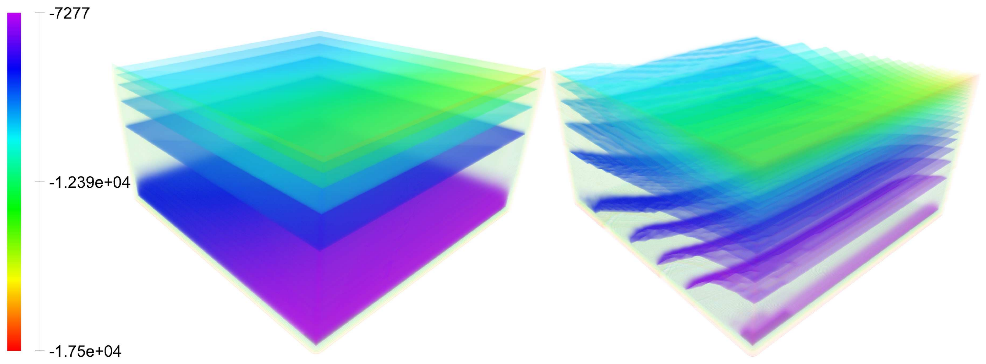


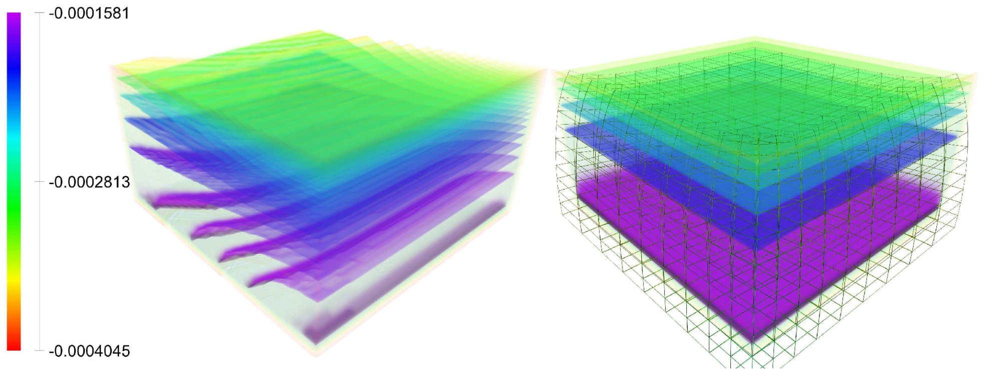
Figure 8: Tensor splats directly visualize the total stress field at the undeformed nodes after 30 days of drying. Green and red represents tensors having positive Eigenvalues. Green represents a tensor with one dominant (linear) and red a tensor with two dominant (planar) Eigenvalues. The Eigenvectors of the stress tensor are visualized by orientation and texture of the splats. Planar tension on the top and linear tension at the edges of the prism are illustrated. Blue represents negative values of at least one Eigenvalue. Also, $\epsilon_{sh}$ is illustrated similar to Figure 7.

## Visualization of a drying contrete specimen

Shrinkage tests on the specimen with dimensions of 100x100x56 mm [Theiner12] are the groundwork for the a numerical simulation, done on one eight of the specimen in ABAQUS 6.10. The three presented visualization techniques were combined to visualize the resulting stress tensor, the drying shrinkage strains, and the deformed structure. Figures 7, 8, and 9 show data after 30 days of drying. The shrinkage tests



(a) $\epsilon_{sh}$ via iso surface and $s_{11}$ via color (b) $s_{11}$ via iso surface and via color



(c) $s_{11}$ via iso surface and $\epsilon_{sh}$ via color (d) $\epsilon_{sh}$ via iso surface and via color

Figure 9: Dual volume rendering of the first component of the effective stresses ($s_{11}$ in [GPa]) and the drying shrinkage ($\epsilon_{sh}$). The images on the left show the two fields in one image via iso surfaces and color, (a) being inverse to (c). The images on the right illustrate one field via iso surfaces and color

on the concrete prisms have been conducted with sealed lateral surfaces to enforce one dimensional moisture migration. The distribution of the computed drying shrinkage strains (sh) is therefore just nonlinear in direction of the drying process (Figure 9). Regions near the surface dry faster resulting in higher values of sh at the surface. This causes, as can be seen in Figure 7 at the deformed structure, an uplift of the edges and the corner of the prism at the surface. The distribution of the effective stress component s11, depicted in Figure 9, illustrates the increasing capillary pressure in regions near the surface and the stresses caused by the deformations due to drying.

## Summary

We presented an approach for advanced visualization of stress and scalar fields given on a data structure describing finite elements. We propose a highly systematic approach to (re-)organize the data based on the mathematical concepts of fiber bundles. This formulation of interpolation nodes and weights as spatio-temporal data allows to reuse visualization components for displaying scalar fields of our visualization framework, hereby being extended for handling FE data. We demonstrated visualization methods for displaying displacement and stress field visualization.

## References

Benger, W., Hege, H.-C. (2004). “Tensor Splats”, Conference on Visualization and Data Analysis 2004, *Proceedings of SPIE*, vol. #5295, p. 151-162.

Benger, W., Ritter, G., and Heinzl, R. (2007). “The Concepts of VISH.”, *Proc. 4th High End Visualization Workshop Obergurgl*, Lehmanns Media, p. 26-39.

Benger, W., Haider, M., Stoeckl, J., Biagio, C., Ritter, M., Steinhauser, D., and Hoeller, H. (2012). “Visualization methods for numerical astrophysics”, Chapter in *Astrophysics*, InTechOpenAccess publisher, ISBN 978-953-51-0473-5.

Bock, A., and Sunden, E., Liu, B., Wuensche, B., Ropinski, T. (2012). “Coherency-Based Curve Compression for High-Order Finite Element Model Visualization”, *IEEE TVCG (SciVis Proceedings)*, vol. #18, p. 2315-2324.

Butler, D. M., and Pendley, M. H. (1989). “A visualization model based on the mathematics of fiber bundles”, *Comp. in Physics*, 3(5), 45-51.

Engel, K., Hadwiger, M., Kniss, J. M., Rezk-Salama, C., and Weiskopf, D. (2006). “Real-Time Volume Graphics”, AK Peters, ISBN: 1-56881-266-3.

Garrity, M.P. (1990), “Raytracing irregular volume data”. *SIGGRAPH Comput. Graph.*, Vol. # 24, p. 35-40, ISSN 0097-8930, ACM, New York, USA.

HDF-Group. (2013). “HDF5 - Home Page”, http://www.hdfgroup.org/HDF5.

Ritter, M. (2009). “Introduction to HDF5 and F5”, *CCT Technical Report Series*, Lousiana State University, CCT-TR-2009-13.

Valentini, B., Theiner, Y., Aschaber, M., Lehar, H., and Hofstetter, G. (2012). “Single-phase and multi-ph. modeling of concrete struct.”, *Eng. Str.*, ISSN 0141-0296.

Stompel, A., Lum, E.B., Ma, K. (2002). “Visualization of multidimensional, multivariate volume data using hardware-accelerated non-photorealistic rendering techniques”, *In Proc. of Pacific Graph. 2002 Conference*, IEEE, p. 394-402.

Theiner, Y., and Hofstetter, G. (2012). “Evaluation of the effects of drying shrinkage on the behavior of concrete structures strengthened by overlays”, *Cement and Concrete Research*, vol. 42, i. 9, p. 1286-1297, ISSN 0008-8846.

Williams, P. L. (1992). “Interactive Direct Volume Visualization of Curvilinear and Unstructured Data”, PHD Thesis, *University of Illios*.