ASCII post-processing file format version 1.2

File format of this file comes from the GMSH system. Following text is copied from the GMSH documentation.

`============== BEGIN OF INSERTED TEXT ==============`

The ASCII post-processing file is divided in several sections: one format section, enclosed between `$PostFormat-$EndPostFormat` tags, and one or more post-processing views, enclosed between `$View-$EndView` tags:

`$PostFormat`
`1.2` *file-type data-size*
`$EndPostFormat`
`$View`
view-name nb-time-steps
nb-scalar-points nb-vector-points nb-tensor-points
nb-scalar-lines nb-vector-lines nb-tensor-lines
nb-scalar-triangles nb-vector-triangles nb-tensor-triangles
nb-scalar-quadrangles nb-vector-quadrangles nb-tensor-quadrangles
nb-scalar-tetrahedra nb-vector-tetrahedra nb-tensor-tetrahedra
nb-scalar-hexahedra nb-vector-hexahedra nb-tensor-hexahedra
nb-scalar-prisms nb-vector-prisms nb-tensor-prisms
nb-scalar-pyramids nb-vector-pyramids nb-tensor-pyramids
nb-text2d nb-text2d-chars nb-text3d nb-text3d-chars
<*time-step-values*>
<*scalar-point-values*>
<*vector-point-values*>
<*tensor-point-values*>
<*scalar-line-values*>
<*vector-line-values*>
<*tensor-line-values*>
<*scalar-triangle-values*>
<*vector-triangle-values*>
<*tensor-triangle-values*>
<*scalar-quadrangle-values*>
<*vector-quadrangle-values*>
<*tensor-quadrangle-values*>

<scalar-tetrahedron-values>
<vector-tetrahedron-values>
<tensor-tetrahedron-values>
<scalar-hexahedron-values>
<vector-hexahedron-values>
<tensor-hexahedron-values>
<scalar-prism-values>
<vector-prism-values>
<tensor-prism-values>
<scalar-pyramid-values>
<vector-pyramid-values>
<tensor-pyramid-values>
<text2d> <text2d-chars>
<text3d> <text3d-chars>
`$EndView`

where:

file-type is an integer equal to 0 in the ASCII file format.

data-size is an integer equal to the size of the floating point numbers used in the file (usually, *data-size* = sizeof(double)).

view-name is a string containing the name of the view (max. 256 characters).

nb-time-steps is an integer giving the number of time steps in the view.

nb-scalar-points, *nb-vector-points*, ... are integers giving the number of scalar points, vector points,... in the view.

nb-text2d, *nb-text3d* are integers giving the number of 2D and 3D text strings in the view.

nb-text2d-chars, *nb-text3d-chars* are integers giving the total number of characters in the 2D and 3D strings.

time-step-values is a list of *nb-time-steps* double precision numbers giving the value of the time (or any other variable) for which an evolution was saved.

scalar-point-value, *vector-point-value*, ... are lists of double precision numbers giving the node coordinates and the values associated with the nodes of the *nb-scalar-points* scalar points, *nb-vector-points* vector points,..., for each of the *time-step-values*.

For example, *vector-triangle-value* is defined as:

coord1-node1 coord1-node2 coord1-node3
coord2-node1 coord2-node2 coord2-node3
coord3-node1 coord3-node2 coord3-node3
comp1-node1-time1 comp2-node1-time1 comp3-node1-time1
comp1-node2-time1 comp2-node2-time1 comp3-node2-time1
comp1-node3-time1 comp2-node3-time1 comp3-node3-time1
comp1-node1-time2 comp2-node1-time2 comp3-node1-time2
comp1-node2-time2 comp2-node2-time2 comp3-node2-time2
comp1-node3-time2 comp2-node3-time2 comp3-node3-time2
...

text2d is a list of 4 double precision numbers:

coord1 coord2 style index

where *coord1* and *coord2* give the coordinates of the leftmost element of the 2D string in screen coordinates, *index* gives the starting index of the string in *text2d-chars* and *style* is currently unused.

text2d-chars is a list of *nb-text2d-chars* characters. Substrings are separated with the '^' character (which is a forbidden character in regular strings).

text3d is a list of 5 double precision numbers

coord1 coord2 coord3 style index

where *coord1*, *coord2* and *coord3* give the coordinates of the leftmost element of the 3D string in model (real world) coordinates, *index* gives the starting index of the string in *text3d-chars* and *style* is currently unused.

text3d-chars is a list of *nb-text3d-chars* chars. Substrings are separated with the '^' character.

```
=============== END OF INSERTED TEXT ===============
```

More information about GMSH can be found at its homepage:
`http://www.geuz.org/gmsh/`

Comments concerning `FFLOW20`:

- `FFLOW20` generates `.POS` file with four views: Elements' pressure, edges' pressure, interelement fluxes and complex view. First three views shows "raw data", results obtained by the solver without any interpolations, smoothing etc. The fourth view contains data processed in this way.

  Elements' pressure: Contains only *scalar-triangle-values*. Triangles are the same as the elements of the original mesh. We prescribe constant value of the pressure on the element, as it was calculated by the solver as the unknown p. Therefore, the three values on every triangle are the same.

  Edge pressure: Contains only *scalar-line-values*. The lines are the same as the edges of the elements of the original mesh. We prescribe constant value of the pressure on the edge, as it was calculated by the solver as the unknown λ. Therefore, the two values on every edge are the same.

  Interelement flux: Contains *vector-point-values* and *scalar-triangle-values*. The *scalar-triangle-values* carry no information, all values are set to 0, these are in the file only to define a shape of the elements. The points for the *vector-point-values* are midpoints of the sides of the elements. The vectors are calculated as $u\mathbf{n}$, where u is value of the flux calculated by the solver and $\mathbf{n}$ is normalized vector of outer normal of the element's side.

  Complex view: Contains *scalar-triangle-values* and *vector-point-values*. The *scalar-triangle-values* shows the shape of the pressure field. The triangles are the the same as the elements of the original mesh. Values of pressure in nodes are interpolated from ps and λs. The *vector-point-values* shows the velocity of the flow in the centres of the elements.