# FEM calculations

KUM International, October 23, 2019

Dr. Ioannis Zotos

# Introduction.

- Why use FEM?
  - Better consideration of deformation effects, for example on shaft calculation and thus CA, or Khβ.
  - More accurate calculation of root stresses, especially in cases that the standards do not cover so good.

# FEM packages used.

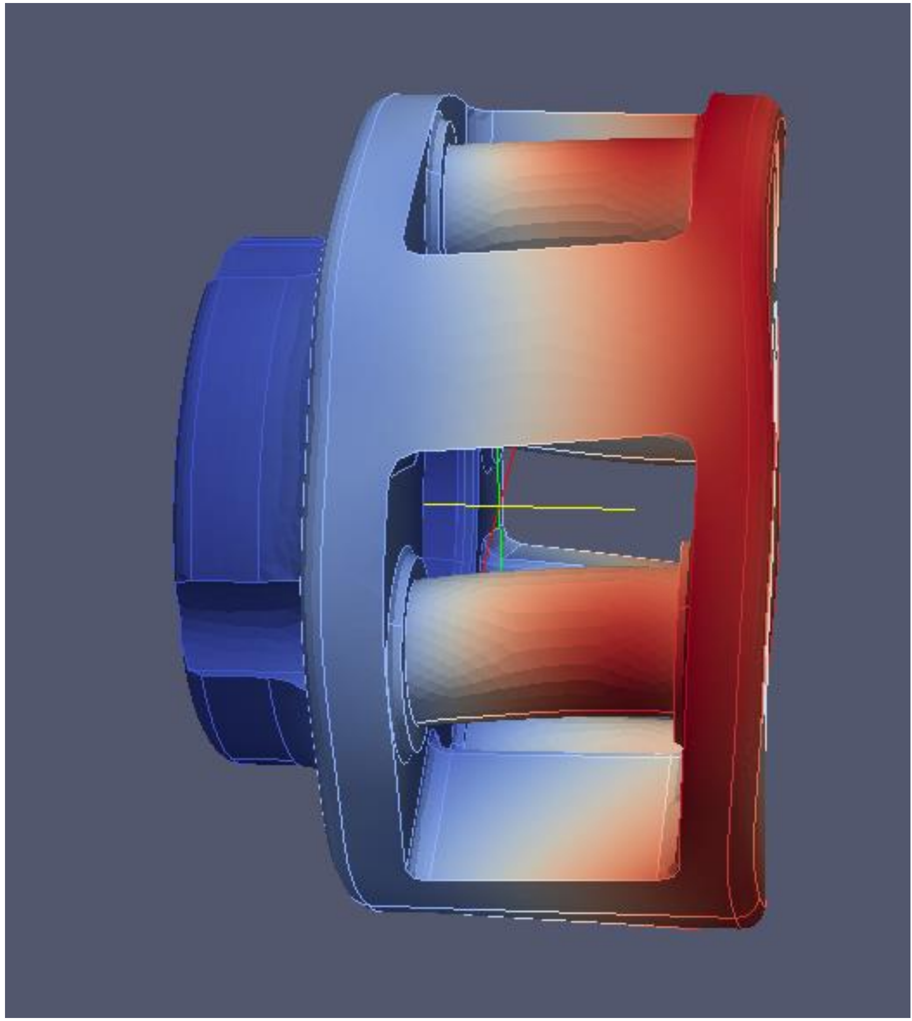
| | Pre – Post processor: SALOME http://www.salome-platform.org/ | Solver: code_aster http://www.code-aster.org/ |
|---|---|---|
| Developing partners | EDF, EADS, Bureu Veritas | EDF, Universities |
| Year of first version | 2000 | 1991, open source since 2001 |
| Platform | Linux, Windows | Linux, Windows |
| Development cycle | Every 6 months | Stable operating version every 2 years. Fixes every month. |
| Other | 6000 tests. Post processing based on Paraview. Automatic mesh generation & refinement | 2000 tests, nuclear industry quality, 14000 pages of user's manual |



KISSsoft

# Planet carrier deformation.

- Single and double sided.
- Different geometric inputs (e.g. straight flanks, cavities, etc).
- Import of STEP file



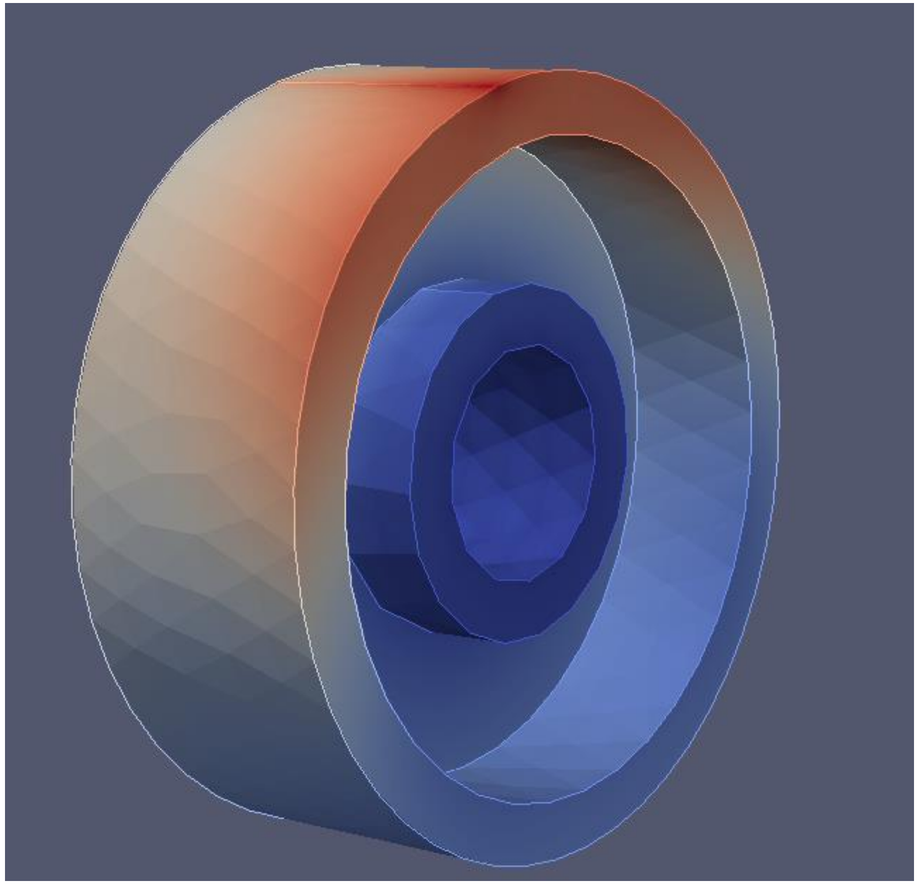


KISSsoft

# Analyses currently available.

## Gear body deformation

- Different geometric inputs.
- Straight or inclined web.
- Deformation or stiffness matrix output.
- Stiffness matrix can be imported and used in tooth trace modification or contact analysis calculation.

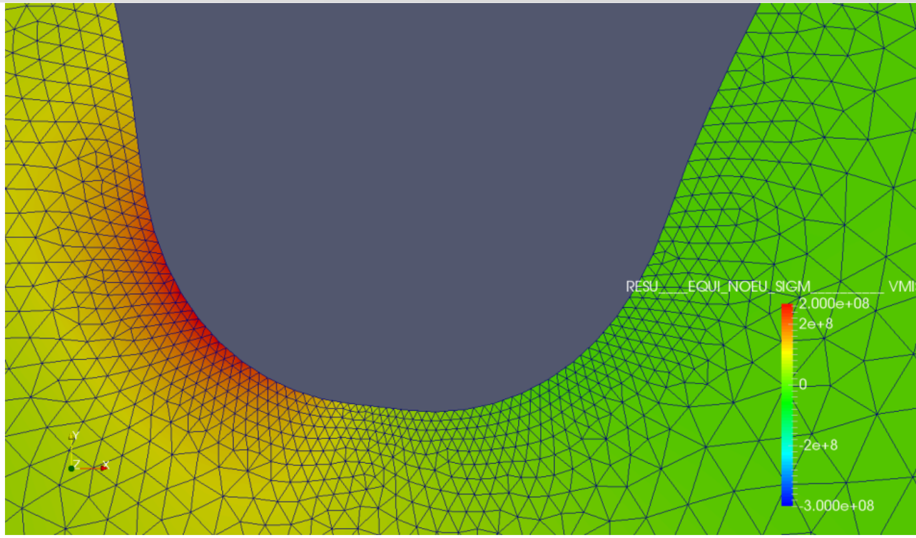


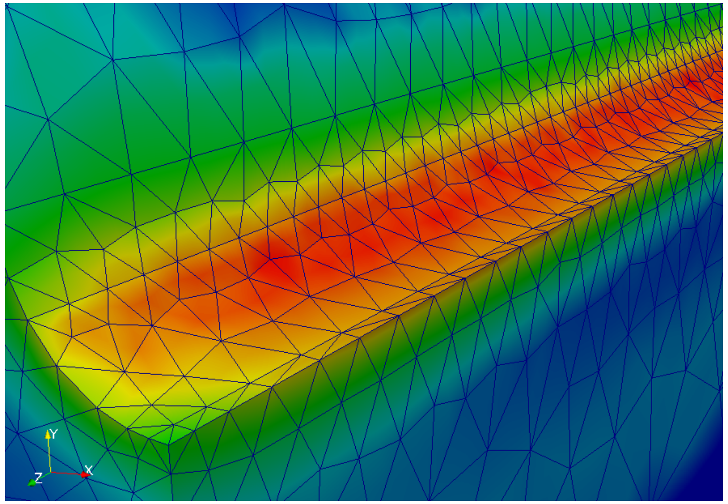


KISSsoft

# Gear root stress (both 2D and 3D).

- Geometric input directly from KISSsoft tooth form.
- Load from contact analysis (3D).
- Different types of boundary conditions.

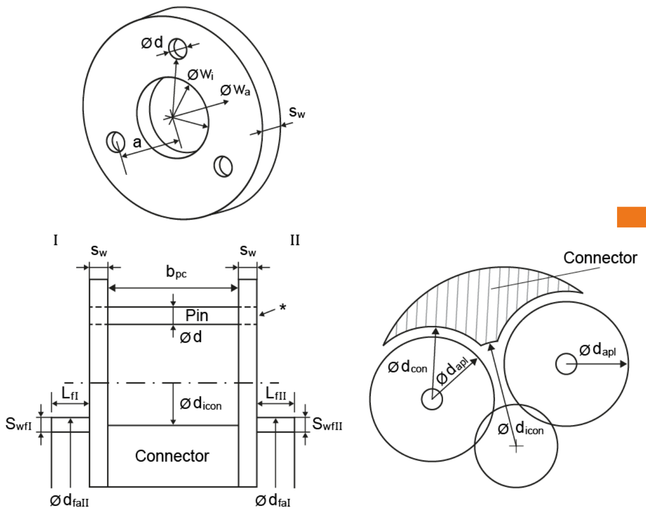


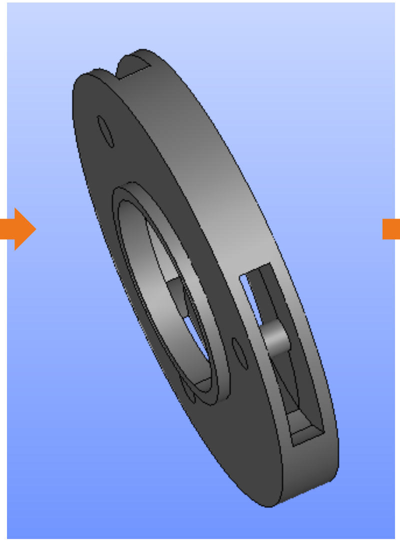



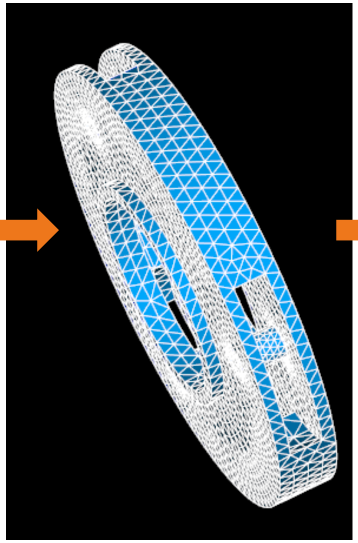


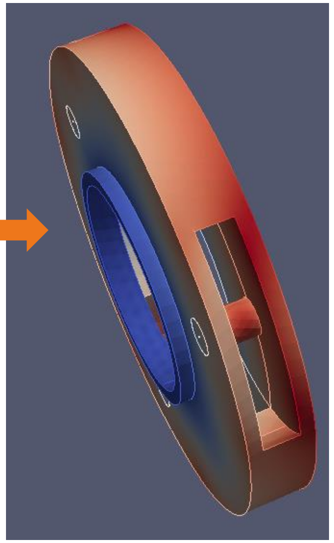

KISSsoft

# Calculation steps.







1. Parametric model

2. 3D model

3. FEM mesh

4. Results

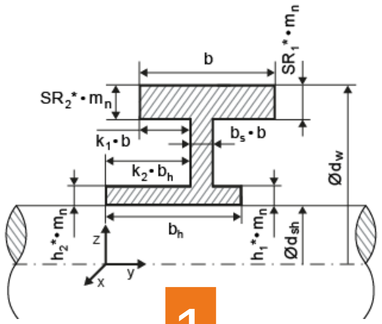
User input in KISSsoft

Automatic FEM analysis
in Salome and code_aster



KISSsoft

# Integration procedure.




1

```
if (coordsDif > 1e-14):
    Edge = geompy.MakeEdge(Vertex1, Vertex2)
    Wire = geompy.MakeWire([Edge], 1e-007)
IF (%i==0) THEN        {mGearBody.innerCylinderLine}
IF (%i==0) THEN        {mGeometry.isInternalGear}
    geompy.addToStudy( Wire, 'ShaftLine' )
ELSE
    geompy.addToStudy( Wire, 'LoadLine' )
END;
END;
IF (%i==0) THEN        {mGearBody.outerCylinderLine}
    outerRadious = z
    center1 = geompy.MakeVertex(0, %3.4f, 0) {mGearBody.pointCoords[#id1, 1]}
    center2 = geompy.MakeVertex(0, %3.4f, 0) {mGearBody.pointCoords[#id2, 1]}
    Circle = geompy.MakeCircle(center1, OY, outerRadious)
    geompy.addToStudy( Circle, 'Circle1' )
    Circle = geompy.MakeCircle(center2, OY, outerRadious)
    geompy.addToStudy( Circle, 'Circle2' )
IF (%i==0) THEN        {mGeometry.isInternalGear}
    geompy.addToStudy( Wire, 'LoadLine' )
ELSE
    geompy.addToStudy( Wire, 'ShaftLine' )
END;
END;

##Continue with the other connection points in a loop
FOR count=1 TO %i BY 1 DO {mGearBody.pointCoords.size()-1}
SETVAR id1 = %i {mGearBody.conPoint1[#count]}
x = %3.4f{mGearBody.pointCoords[#id1, 0]}
y = %3.4f{mGearBody.pointCoords[#id1, 1]}
z = %3.4f{mGearBody.pointCoords[#id1, 2]}
Vertex1 = geompy.MakeVertex(x, y, z)
SETVAR id2 = %i {mGearBody.conPoint2[#count]}
```

2

```
if (coordsDif > 1e-14):

    Edge = geompy.MakeEdge(Vertex1, Vertex2)

    Wire = geompy.MakeWire([Edge], 1e-007)

    geompy.addToStudy( Wire, 'ShaftLine' )



##Continue with the other connection points in a loop

x = 0.0000

y = 0.0450

z = 0.0200

Vertex1 = geompy.MakeVertex(x, y, z)

x = 0.0000

y = 0.0450

z = 0.0300

Vertex2 = geompy.MakeVertex(x, y, z)
```

3

C:\Windows\system32\cmd.exe

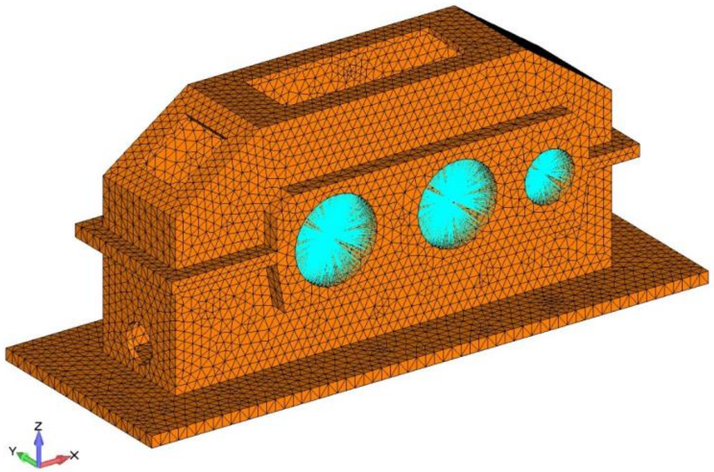
```
                    DDL_IMPO=_F(GROUP_NO='HOLD',
                                LIAISON='ENCASTRE'),
                    LIAISON_XFEM='NON',
                    MODELE=MOD,
                    );

 #  USAGE DE LA MEMOIRE JEVEUX
 #     - MEMOIRE DYNAMIQUE CONSOMMEE :      15.96 Mo (MAXIMUM ATTEINT :      17.92 Mo)
 #     - MEMOIRE UTILISEE            :       5.79 Mo (MAXIMUM ATTEINT :      14.36 Mo)
 #  FIN COMMANDE NO : 0008   USER+SYST:      0.02s (SYST:       0.00s, ELAPS:       0.00s)
 #  ----------------------------------------------------------------------------------------

 #  ----------------------------------------------------------------------------------------
 #  COMMANDE NO :  0009           CONCEPT DE TYPE : char_meca
 #  -------------                 -----------------
Force=AFFE_CHAR_MECA(FORCE_ARETE=_F(FX=5.E5,
                                FY=0.0,
                                GROUP_MA='LOADLN',
                                FZ=0.0),
                  INFO=1,
                  VERI_NORM='OUI',
                  LIAISON_XFEM='NON',
                  MODELE=MOD,
                  );
```

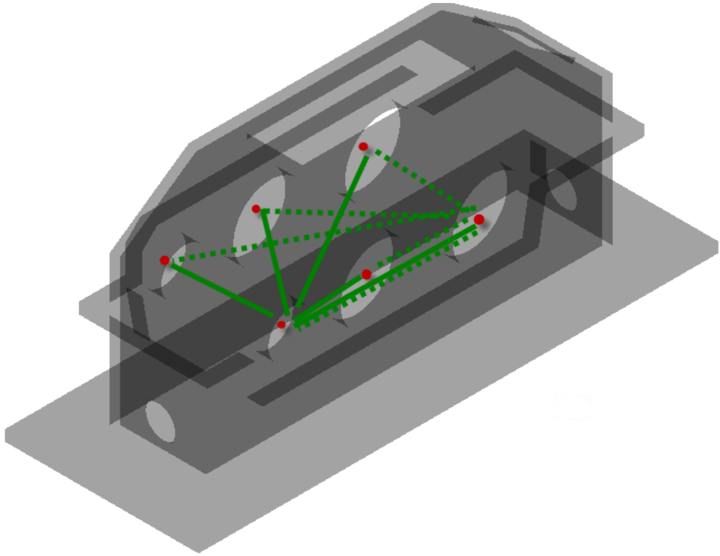


KISSsoft

# Housing deformation.

- Read-in of the finite element reduced stiffness matrix:
  - “spring like” constants connecting the “bearing nodes” of the FE grid, for all degrees of freedom.
- Interfaces to:
  - ANSYS
  - NASTRAN
  - ABAQUS
  - ALTAIR OptiStruct





KISSsoft

# Conclusions

1. Robust open source FEA software integrated with KISSsoft (seamless integration).
2. Calculations:
   - Gear body deformation.
   - Planet carrier deformation.
   - Tooth root stress.
   - Housing stiffness (through file input).
3. Powerful post processing tool.

# Thank you for your attention!

Sharing Knowledge

KISSsoft AG, A Gleason Company
Rosengartenstrasse 4, 8608 Bubikon, Switzerland
T. +41 55 254 20 50, info@KISSsoft.AG, www.KISSsoft.AG