FELyX
–
The Finite Element Library Experiment

Overview Mai 2002

Oliver König, Marc Wintermantel, Nino Zehnder
Institute of Mechanical Systems
ETH Zürich

http://felyx.sourceforge.net

FELyX – The Finite Element Library Experiment provides:

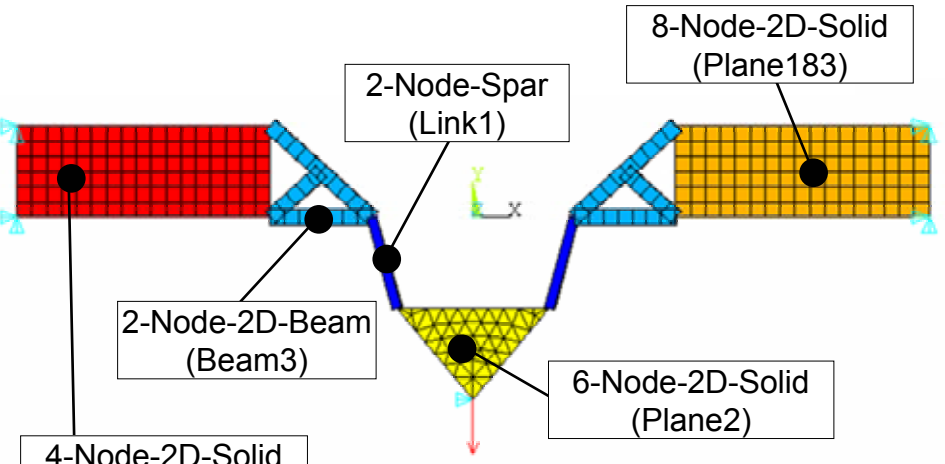
- **Object oriented structure implemented in C++**
- **Free availability under GNU license → http://felyx.sourceforge.net**
- **All relevant elements for static structural analyses.**
- **Profile / bandwidth reduction algorithms based on graph library of Boost:**
  - (Reversed) Cuthill-McKee bandwidth minimization.
  - Sloan profile minimization algorithm.
- **Efficient matrix storage schemes using the Matrix Template Library (MTL):**
  - Envelope/skyline storage scheme for direct solvers
  - Sparse storage schemes for iterative solvers
- **Efficient solvers:**
  - Direct skyline solver
  - Interface to the Iterative Template Library (ITL) and its conjugate gradient solvers
- **Interface to exchange models with ANSYS**

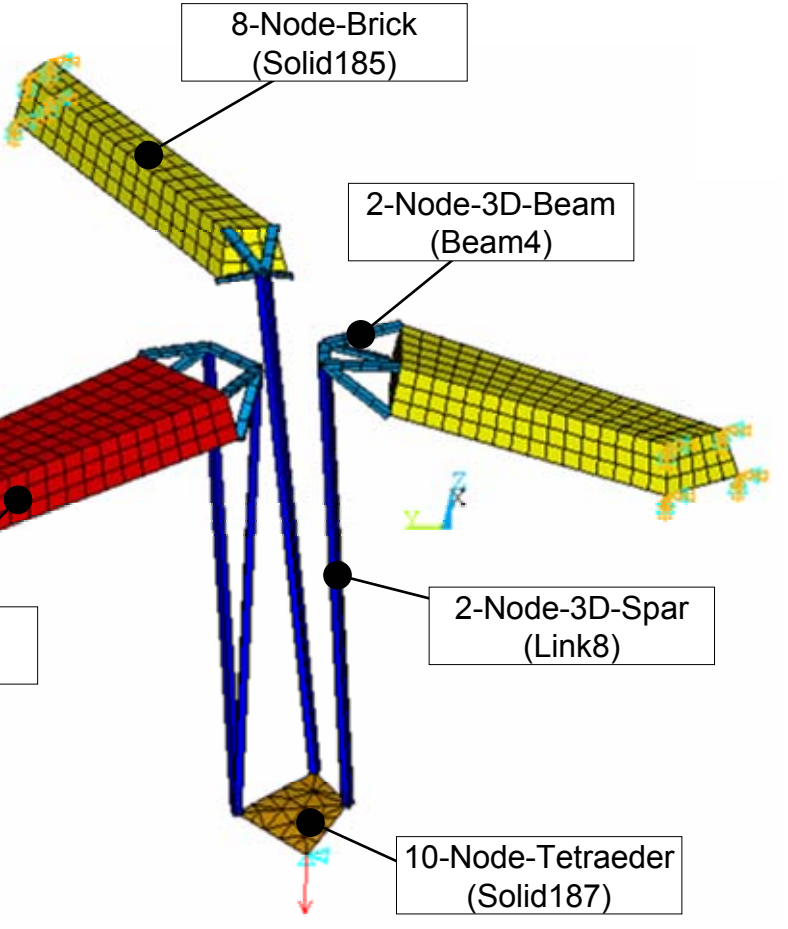
Motivation for FELyX

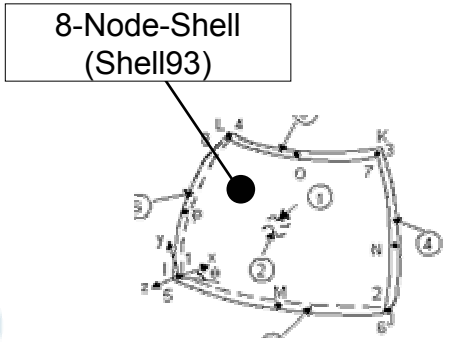
The main motivation to write FELyX was to have a free and object oriented Finite Element code that is numerically efficient and allows to access the Finite Element calculation at any level.

The need for this arised within our ongoing PhD thesis in the field of Structural Optimization at the Swiss Federal Institute of Technology Zurich (ETH Zurich) in the group Structures Technology (IMES-ST). As numerical optimization methods we mainly use Genetic Algorithms and sometimes Gradient Methods. This means that we typically have to evaluate thousands of similar Finite Elements Models (therefore efficiency is definitively a point).

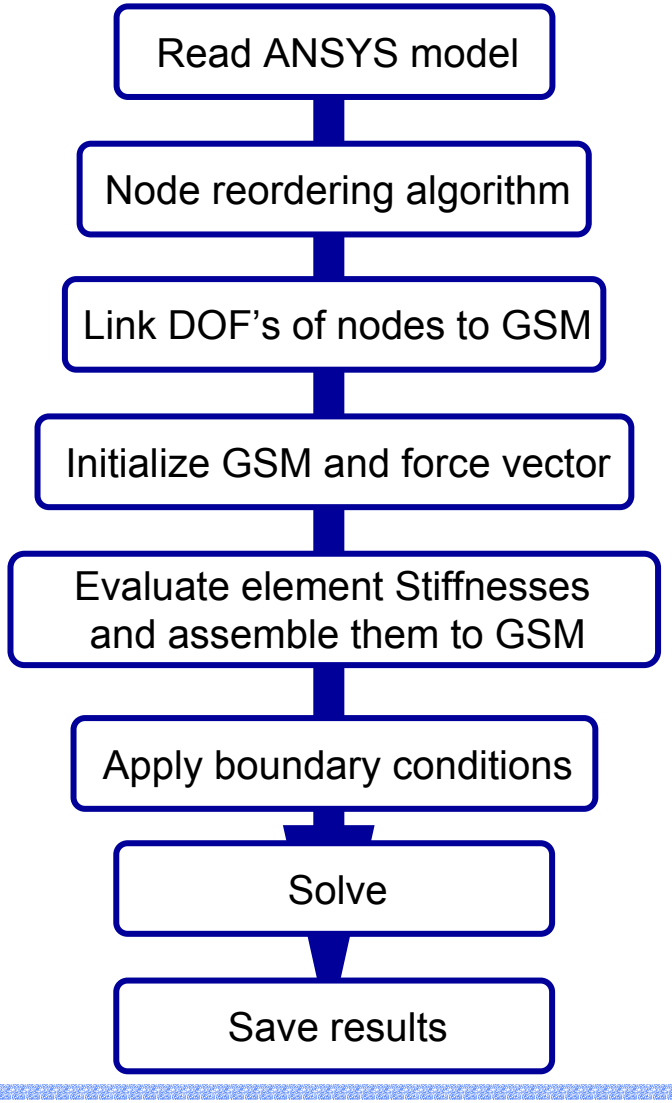
Elements implemented in FELyX



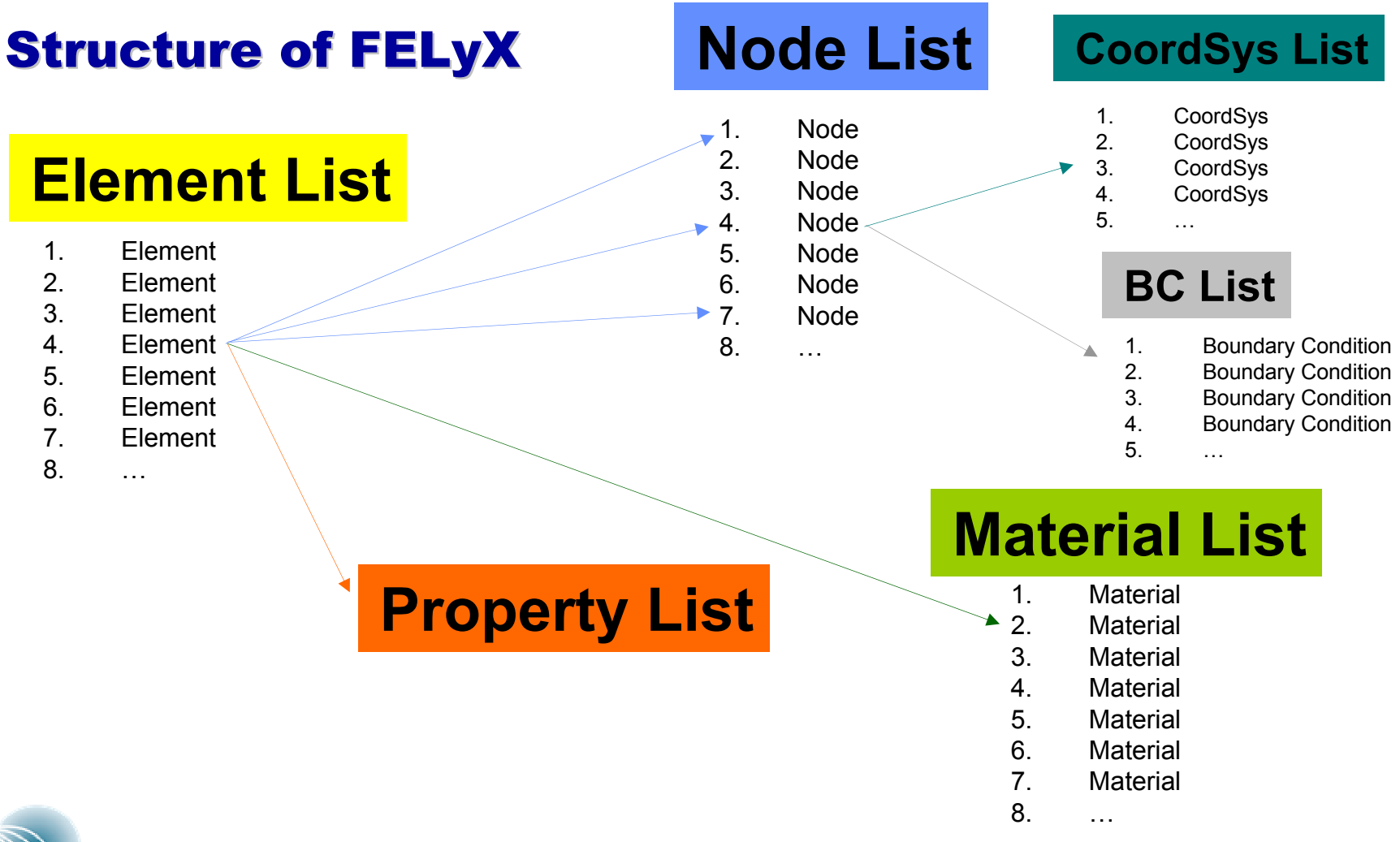




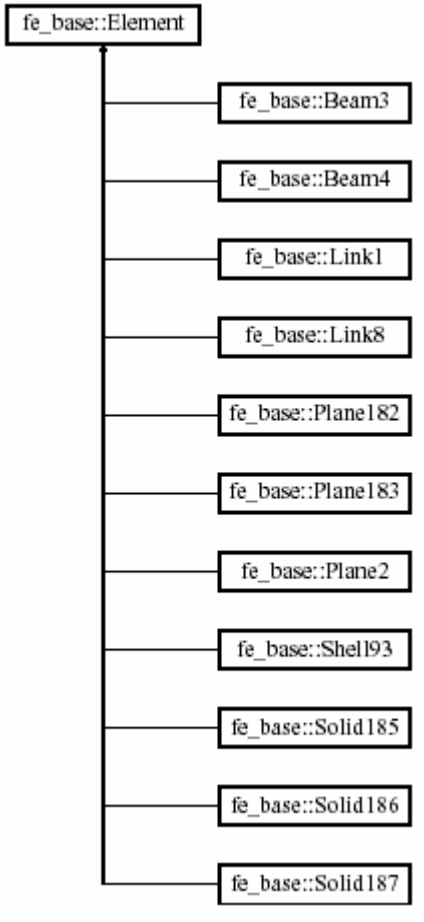
Procedure of a FELyX run



Structure of FELyX



Element data objects in FELyX




```
// Abstract base class for all elements
// No instances of "Element" can be created
class Element{
public:
  // Data members
  // ------------
  Node** NodePtr;                    // Pointer to the first node
  Material* MaterialPtr;             // Pointer to material
  PropertySet* PropertiesPtr;        // Pointer to properties
```

Node data object in FELyX

```
//data-struct for a single node
class Node{

public:
  // Data members
  // ------------
  double Cx;                          //x-coordinate
  double Cy;                          //y-coordinate
  double Cz;                          //z-coordinate

  NodeCoordSys *NodeCoordSysPtr;      // Pointer to nodal coordinate system
                                      // (Pointer is set to NULL, if nodal coord sys = global coord sys)
  BoundCon *BoundConPtr;              // Pointer to the boundary conditions of the node
                                      //  (Pointer is set to NULL if no BD are present)

  const static int NodeDofVecCount = 6;

  DofSet  NodeDofSet;                 // bitset<6> containing the information which nodal degrees of
                                      // freedom are activ. (ux, uy, uz, rx, ry, rz)
  unsigned Idx2Gsm;                   // integer giving the position of the first active DOF in the
                                      // Global Stiffness Matrix (GSM)
```

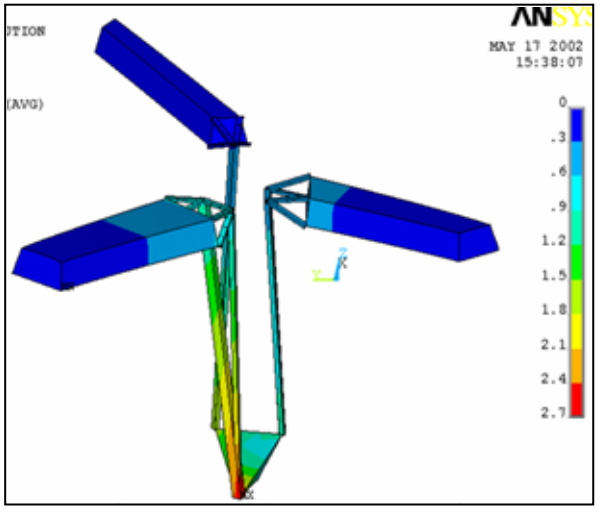
Contents of performance tests for the direct solver

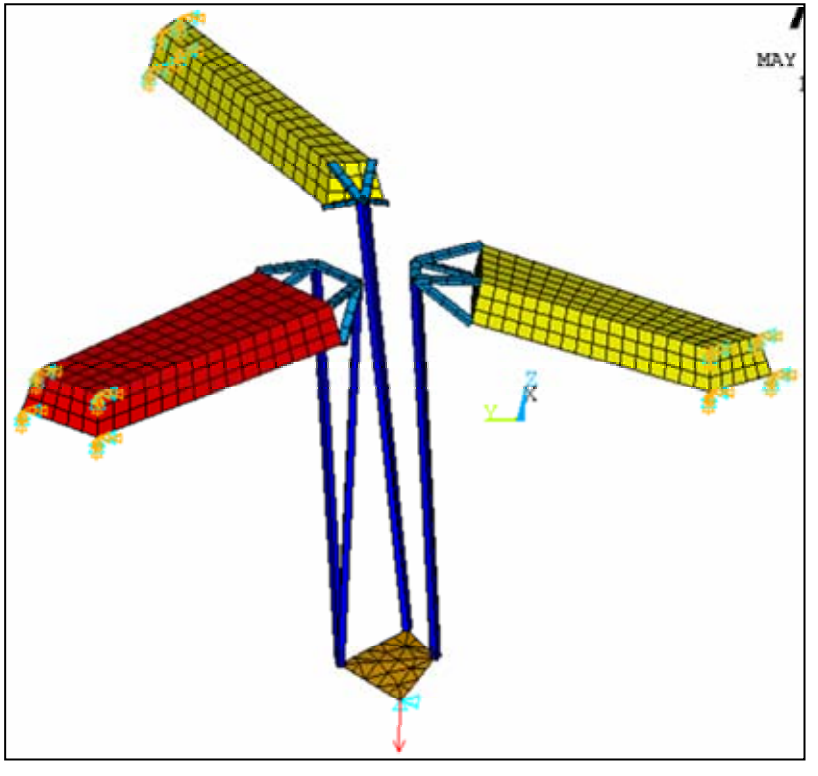
- **Test problem 1: Small sized 3D structure composed of 5 element types**
  - Solving test problem in ANSYS
  - Solving test problem in FELyX without any reordering strategies.
  - Solving test problem in FELyX using Reversed Cuthill-McKee bandwidth minimzation and banded GSM
  - Solving test problem in FELyX using Sloan's profile reduction algorithm and envelope storage format for the GSM

- **Test problem 2: Moderate sized 3D structure with 10-node tetraeder**
  - Solving test problem in FELyX using Reversed Cuthill-McKee bandwidth minimzation and banded GSM
  - Solving test problem in FELyX using Sloan's profile reduction algorithm and envelope storage format for the GSM

Test problem1 : Small 3D-mixed structure

- # element types involved: 5
- # elements: 718
- # DOF's 6796
- ANSYS runtime: ca 4s





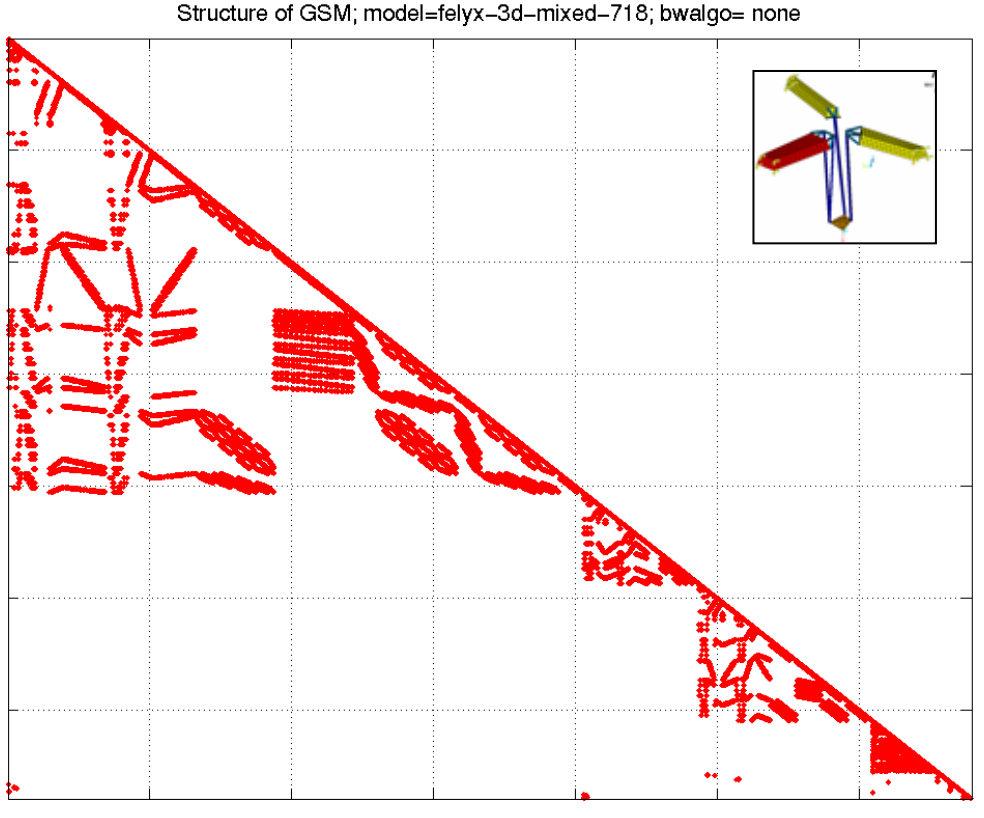
Solving test problem in FELyX without any reordering strategies

- **Memory needs for the GSM**
  - Storing lower triangle of matrix
  - Double values ⇔ 8 bytes

  → ca 185 MB

- **Runtimes**

  → > 600s



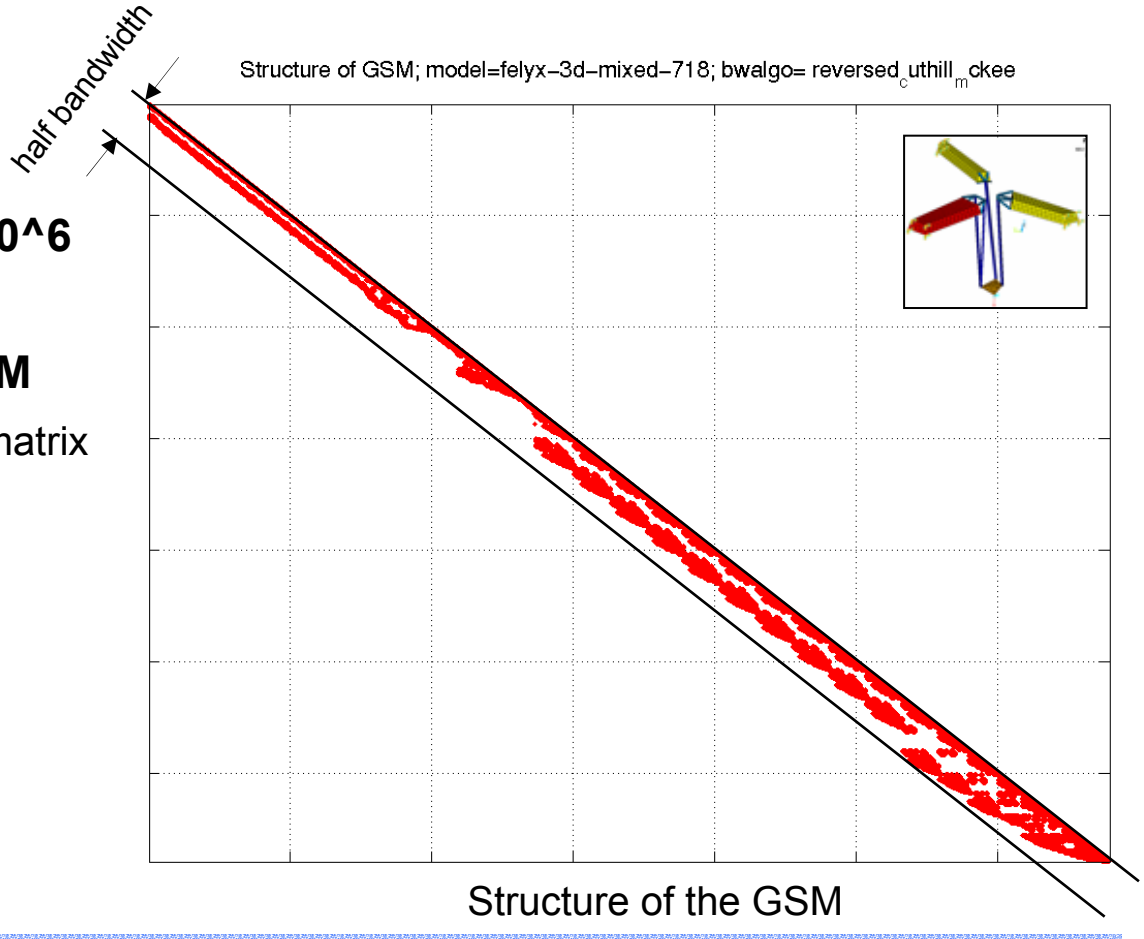
Structure of the GSM

Solving test problem in FELyX using Reversed Cuthill-McKee bandwidth minimzation and banded GSM

- **Half bandwidth: 513**
- **Profile: 1.20 x 10^6**
- **Memory needs for the GSM**
  - Storing lower band of the matrix

  → ca 28 MB
- **Runtimes**

  → ca 10s

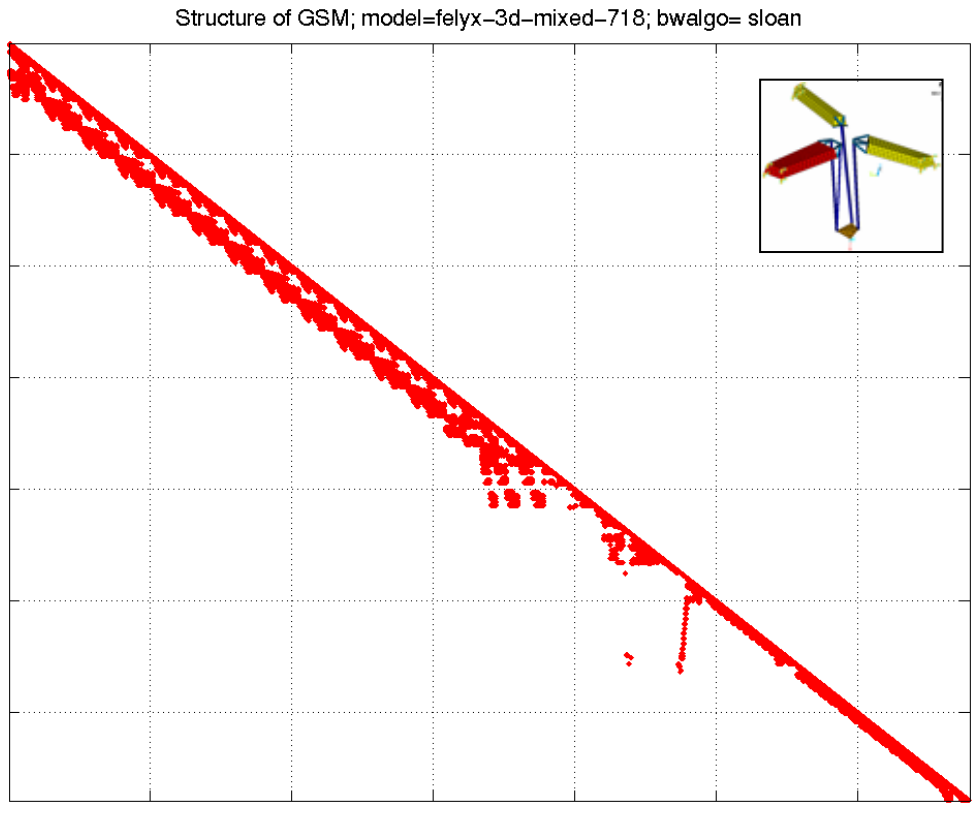


Solving test problem in FELyX using Sloan's profile reduction algorithm and envelope storage format for the GSM

- **Half bandwidth: 1192**
- **Profile: 1.17 x 10^6**
- **Memory needs for the GSM**
  - Storing lower envelope of the matrix

  → **9 MB**
- **Runtimes**

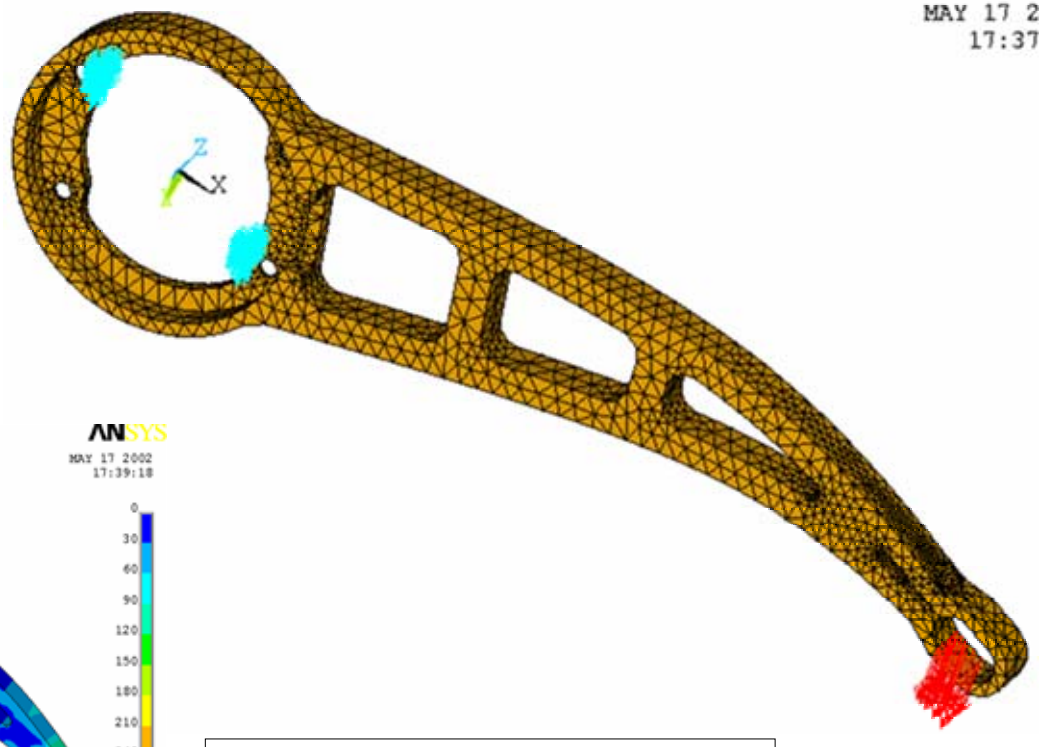
  → **ca 3.5s**

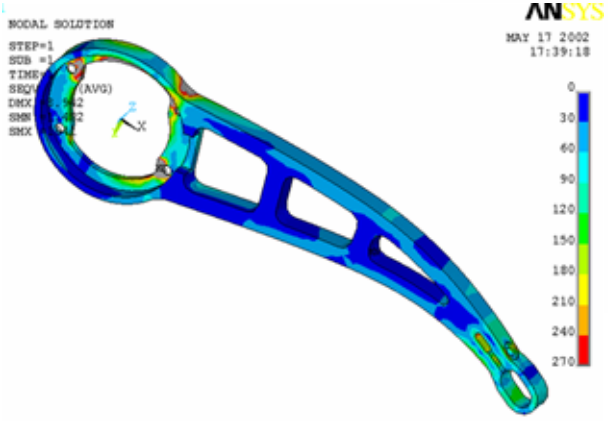


Structure of the GSM

Test problem 2 : Moderate 3D structure

- 10 node tetraeder model
- # elements: 12839
- # DOF's 67629
- ANSYS runtime using direct solver: ca 120s

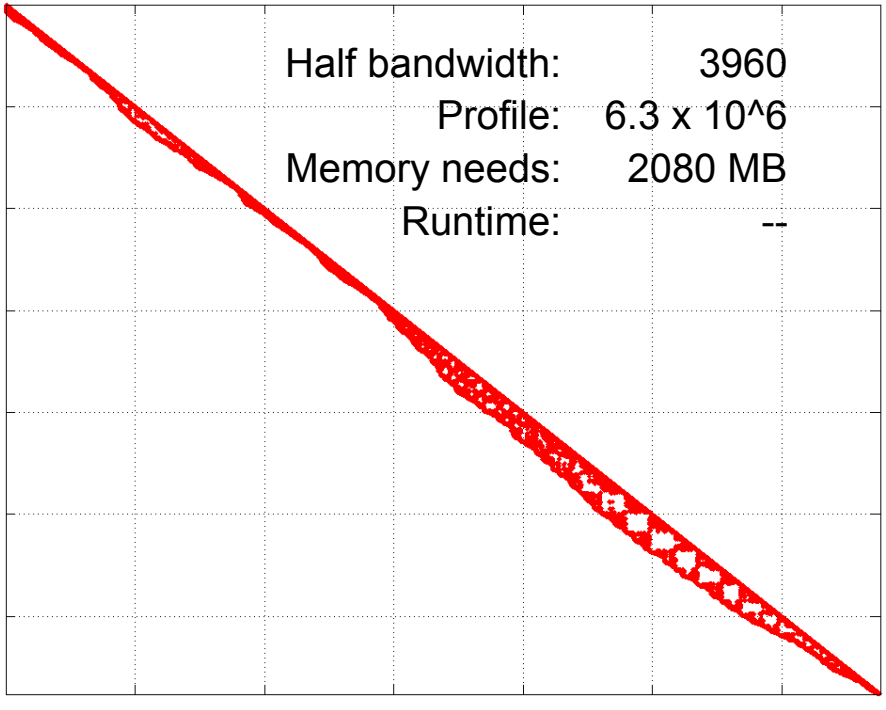




CAD Part could be used by courtesy of Tribecraft AG, Zurich

Solving a larger model with 3D tetraeders

Using reversed cuthill-mckee and band matrix



Using sloan algorithm and envelope matrix