KTH Numerical Analysis
and Computer Science

Department of Numerical Analysis and Computer Science

Benchmarking FEMLAB 3.1i: Solid Structural Mechanics

Michael Hanke

TRITA-NA-0546

NADA is a co-operating departement between the
Royal Institute of Technology and Stockholm University

Michael Hanke
Benchmarking FEMLAB 3.1i: Solid Structural Mechanics

Report number: TRITA-NA-0546
Publication date: November 2005
E-mail of author: hanke@nada.kth.se

Reports can be ordered from:

Numerical Analysis and Computer Science (NADA)
Royal Institute of Technology (KTH)
SE-100 44 Stockholm
SWEDEN

telefax: +46 8 790 09 30
http://www.nada.kth.se/

Benchmarking FEMLAB 3.1i: Solid Structural Mechanics

Michael Hanke
Royal Institute of Technology
Department of Numerical Analysis and Computer Science

November 25, 2005

Abstract

The software package FEMLAB is an environment for modelling and solving multi-physics applications which are described in terms of partial differential differential equations. In this project we tested the performance of the structural mechanics toolbox version 3.1. This version of the toolbox represents a considerable enhancement over previous releases with respect to modelling capabilities. We compare the performance of version 3.1 to that of the older version 2.3 and a state-of-the-art finite element package for solving structural mechanics problems, namely, ANSYS 9.0.

Contents

1 Introduction

Partial differential equations form the mathematical foundation for a host of important areas in engineering and physics. FEMLAB provides a powerful interactive environment for modelling and solving scientific and engineering problems which base on partial differential equations. Using FEMLAB one can model strongly nonlinear coupled multiphysics applications with ease. There is no inherent limitation on the simultaneous simulation of many physical phenomena. FEMLAB can handle (systems of) second and first order partial differential equations in one, two and three space dimensions. They are discretised by the finite element method. The (extensible) element library uses mostly polynomial elements on triangles (in 2D) and tetrahedra (in 3D), respectively. Some elements are available which are adapted to be applied for special applications.

From the point of view of applicability, it is the multiphysics feature and the extensibility which distinguish FEMLAB. A host of models from different areas of applications are prepared in an easily accessible manner (the so-called applications modes) which can be combined by simple drag-and-drop techniques into complex multiphysics models.

A graphical user interface allows for an efficient graphical design of rather complex geometries in one, two and three dimensions.

The powerful capabilities of FEMLAB give immediately rise to the question whether the user has to pay in order to use such a convenient tool. There are a number of very advanced software packages on the market competing with FEMLAB. Often, they have some emphasis on certain

applications areas. This allows for the use of numerical algorithms being more adapted to the application at hand. Compared to that, the algorithms in FEMLAB must be of a more general nature in order to cover the broad spectrum of applications FEMLAB is intended for.

The latest version of FEMLAB is 3.1i.[1] It distinguishes itself from all previous major releases in that all computational kernels have been reimplemented in C++. The previous versions are implemented in MATLAB. While opening all the features of MATLAB to be used in FEMLAB it has the drawback of slowing down the numerics and increasing the memory requirements. Even if FEMLAB 3.1i can be run standalone, an interface to MATLAB is available such that the numerical kernel of FEMLAB can be used as a computing server. We used this possibility extensively in the following benchmark tests.

In the present report, we compare the recent version of the structural mechanics toolbox of FEMLAB against an older one and ANSYS. The latter one has its strength in structural mechanics modelling. Therefore, we will use benchmark problems for the computation of structural mechanics problems. There are already some comparisons of FEMLAB 3.0a with ANSYS and Fluent for two-dimensional models [5, 6]. Here we will concentrate on three-dimensional problems.

We have three aims:

1. Compare different discretisations in FEMLAB 3.1i. Does it pay to use higher order elements? FEMLAB includes the feature of an automatic mesh adaption. Does it pay to use those automatically adapted grids?

2. Compare the performance of FEMLAB 3.1i and the previous MATLAB-based FEMLAB version. We expect shorter computation times and a more memory-economic behaviour. Can we quantify it?

3. Compare FEMLAB 3.1i with ANSYS. How is the performance of the new version compared to a well-established finite element software tool?

2 Benchmark Methodology

2.1 Benchmark Examples

The NAFEMS Benchmark LE10

NAFEMS (National Agency for Finite Element Methods and Standards) developed a number of benchmark problems for allowing the assessment and comparison of codes for the finite element analysis. One of these problems is the linear elasticity analysis of a thick plate under normal pressure. In the reference [3], it carries the label LE10.

The solid consist of a thick elliptical plate with an elliptical hole under normal pressure. Because of the symmetry, the actual computational domain is a quarter of this plate. A sketch of

[1] While preparing this report a new version 3.2 was released. The trade mark of this product has now changed from FEMLAB to Comsol Multiphysics.

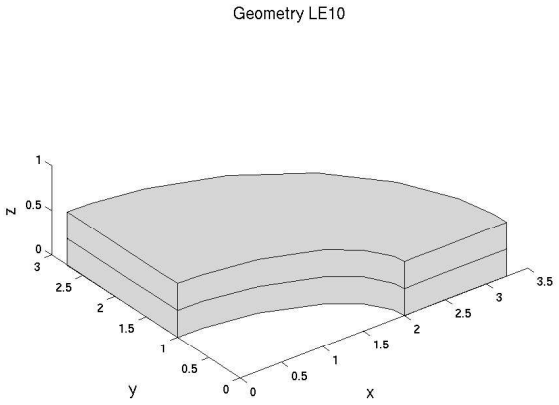

Figure 1: Domain for problem LE10

this domain is given in Figure 1. The equation for the two ellipses are given by

$$\left(\frac{x}{3.25}\right)^2 + \left(\frac{y}{2.75}\right)^2 = 1 \tag{1}$$

and

$$\left(\frac{x}{2}\right)^2 + \left(\frac{y}{2}\right)^2 = 1, \tag{2}$$

respectively. The thickness of the plate is 0.6 (m). The plate is loaded by a uniform normal pressure of 10^6 (Pa).

There are symmetry boundary conditions along the vertical planes. The outer elliptical boundary is fixed in *x*- and *y*-directions. In order to remove the rigid body movement, the *z*-deformation is restricted on the midline of the outer elliptic boundary.

The material is an isotropic elastic material with

$$E = 210 \times 10^9 (\text{Pa}), \quad \nu = 0.3. \tag{3}$$

The value to be computed is the normal stress σ_y at the point $(x,y,z) = (2,0,0.6)$,

$$\sigma_y = -5.38 \times 10^6 (\text{Pa}). \tag{4}$$

For comparison purposes, the other stresses are given below

$$\sigma_x = 0.0(\text{Pa}), \sigma_z = -1.0 \times 10^6 (\text{Pa}). \tag{5}$$

Besides a general test of the performance of the linear 3D-solvers in the codes in question, this benchmark problem tests the abilities of the geometry and mesh generators as well as the quality of the interpolation operators. Since the target point is on the boundary of the domain it will definitely not be an integration node. A detailed discussion can be found in the reference [3].

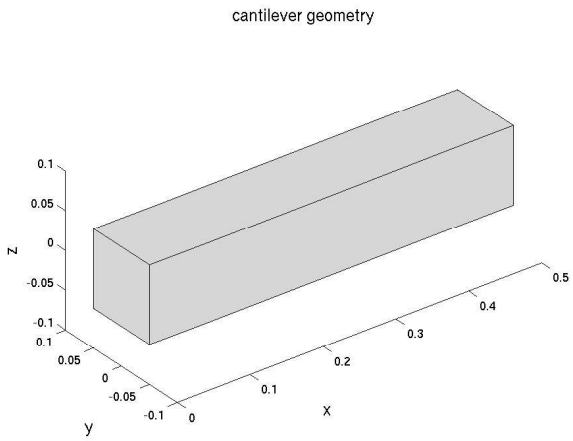

Figure 2: Cantilever beam

A Cantilever Beam Under Large Deformations

The present example is intended to test the robustness especially of the nonlinear solver. The solid is a simple block with the dimensions $0.5 \times 0.1 \times 0.1$ (m^3). For definiteness, let the coordinates be given by, Figure 2,

$$D = [0, 0.5] \times [-0.05, 0.05] \times [-0.05, 0.05]. \tag{6}$$

The body is fixed at the plane $x = 0$. On the opposite side $x = 0.5$, a tangential force is applied which twists the body:

$$F_x = 0, \quad F_y = -Fr\sin\varphi, \quad F_z = Fr\cos\varphi, \tag{7}$$

where r and φ are the polar coordinates of (y,z),

$$y = r\cos\varphi, \quad z = r\sin\varphi. \tag{8}$$

The material is elastic isotropic with

$$E = 2.1 \times 10^{11}, \quad \nu = 0.3. \tag{9}$$

There are two cases for the constant F, $F = 10^{10}$ and $F = 10^{11}$. The latter one is especially hard to solve.

Target values are the deformations at $(x,y,z) = (0.5, 0.05, 0.05)$. The following table contains the target values (in mm). They have been computed with high order elements on very fine grids. Their accuracy is believed to be ± 2 units of the least significant digits given.

F	u	v	w
10^{-10}	-0.0232	-3.842	3.589
10^{-11}	-1.74	38.14	20.98

Natural Frequencies of a Piezoelectric Transducer

This example is taken from [2]. It is also part of the ANSYS verification suite (test case VM175).

The piezoelectric transducer consists of a cube of PZT4 material with its polarisation aligned along the z-axis. Electrodes are placed on the two surfaces orthogonal to the polarisation axis. The task consists of determining the first two coupled-mode natural frequencies for the short circuit and the open circuit case.

Because of symmetry considerations, only one quarter of the cube is used as the computational domain. It is defined as being the block

$$D = [0, l/2] \times [0, l/2] \times [0, l]. \tag{10}$$

The electrodes are located at the bottom ($z = 0$) and top ($z = l$) surfaces. The surfaces $x = 0$ and $y = 0$ are subject to symmetry boundary conditions.

The constants are defined as follows:[2]

Quantity	value
density ρ	7500
length l	0.02
permittivity matrix ε_{rS}	$\begin{pmatrix} 804.6 & 0 & 0 \\ 0 & 804.6 & 0 \\ 0 & 0 & 659.7 \end{pmatrix}$
piezoelectric matrix e	$\begin{pmatrix} 0 & 0 & 0 & 0 & 10.5 & 0 \\ 0 & 0 & 0 & 10.5 & 0 & 0 \\ -4.1 & -4.1 & 14.1 & 0 & 0 & 0 \end{pmatrix}$
elasticity matrix c_E	$\begin{pmatrix} 13.2 & 7.1 & 7.3 & 0 & 0 & 0 \\ 7.1 & 13.2 & 7.3 & 0 & 0 & 0 \\ 7.3 & 7.3 & 11.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3.0 \end{pmatrix} \times 10^{10}$

The bottom electrode is always grounded, i.e., the voltage is set to 0. In the short circuit case, the voltage at the top electrode is also 0. There is no prescribed voltage on the top electrode for the open circuit case. The target values are measured results in [2]:

case	f_1 [kHz]	f_2 [kHz]
short circuit	66.56	88.01
open circuit	81.59	93.41

For the purposes of benchmark computations, these measured values are much to inaccurate. Based on the material data given above we carried out computations with very high accuracy. The eigenfrequencies of the coupled modes turned out to be

[2] In the reference [2], the values in the permittivity matrix are slightly different: 805 and 660, respectively.

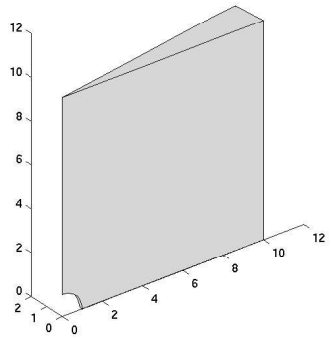

Figure 3: Computational domain for a whole-space problem

case	f_1 [kHz]	f_2 [kHz]
short circuit	65.1011	88.1364
open circuit	79.806	93.576

We believe that all digits are correct, that is, the error is below 0.5 units of the least significant digit.

A Nearly Incompressible Material

The present example is intended to test the performance of the codes at a nearly incompressible material. The maximal stress in a body stretching over the whole 3D space and having a spherical hole under uniaxial forces can be computed analytically. We use an approximation to this situation for testing the code. Let a hole with radius 1 be centred at the origin. The force is applied parallel to the z-axis. Because of symmetry it is sufficient to consider only the half space $z \geq 0$. In order to make the solid finite consider only the cylinder with radius 10 and height 10. The problem has obviously cylindrical symmetry. It would, therefore, be possible to reduce the problem to a two-dimensional one. Since we are interested in three-dimensional test cases, a slice of the cylinder is taken instead. A sketch of the computational domain is given in Figure 3.

More exactly, the computational domain is given by

$$D = \{(x,y,z) | x,y,z > 0, x^2+y^2 < 10, z < 10, x^2+y^2+z^2 > 1, 6y < x\}. \tag{11}$$

The material constants are given as

$$\nu = 0.49, \quad E = 2 \times 10^{11}. \tag{12}$$

The boundary conditions are chosen as follows:

- No constraints on the top, outer cylindrical sidewall, and the sphere near the origin;

- Symmetry constraints on all other surfaces, that is deformation in normal direction is set to zero;
- On the top surface, a force of 10^6 acting in the z-direction was applied.

The target values are the stresses at $(x,y,z) = (1,0,0)$,

$$\sigma_z = 2.164, \quad \sigma_x = 0. \tag{13}$$

The target values have been determined in the following way: For the whole-space problem, the analytical solution is given by [4]

$$\sigma_{\max} = \sigma_\infty \frac{27 - 15\nu}{14 - 10\nu}. \tag{14}$$

The problem was then solved on a large two-dimensional computational domain using the cylindrical symmetry. The computational domain was chosen large as was the number of degrees of freedom (> 200000) in FEMLAB 2.3. Moreover, the triangulation was adapted according to the energy norm error indicator. This way, the accuracy of the computed stress value was below 0.1% compared to the theoretical value. The target value on the smaller domain D was then determined by using the same procedure with a comparable number of degrees of freedom.

2.2 Test Criteria

With all of the cited software tools we computed the quantities mentioned above. The aim is to compare the accuracy of this quantities and the amount of computing resources necessary. Taking into account the very different architecture of the packages with respect to programming principles and user interfaces we decided to benchmark only the pure numerical kernels. This includes also the handling of geometry and grid generation. While this is an integral part of the FEMLAB family it is split over different modules in ANSYS. Therfore, the tests are carried out as follows:

FEMLAB The test cases were generated in the graphical user environment and saved as an MATLAB m-file. Then the generated m-files were hand-edited in order to remove all unnecessary commands. This includes all graphical output. The protocol output was restricted to the standard value `report = 'on'`. This gives negligible overhead for FEMLAB 2.3. In case of FEMLAB 3.1i, a JAVA subprocess was started whose resources we neglected. The only post-processing left is the computation of the benchmark target values. Then, MATLAB was started without its JAVA virtual machine thus reducing its resource requirements to a minimum. For FEMLAB 3.1i, the server was started in parallel such that a connection to MATLAB was established.

ANSYS The test cases were constructed in the graphical user interface and exported as an ANSYS batch command file. The latter was again edited by hand in order to reduce the output to a bare minimum on the console. In particular, no graphics output was generated. The resulting command file was run in a batch queue without any user interaction.

Our aim was to measure both execution time and memory needs.

Memory Especially in three dimensional calculations, we expect that the memory requirements will become a restricting factor. Therefore, it would be worth to measure the memory requirements. Unfortunately, these are not explicitely available in FEMLAB. Neither it is possible to use indirect measurements via tools of the operating system because multi-threading is used. So the only information available is when the FEMLAB started to swap. A problem size was considered computable if the process did not start to swap.

The situation forANSYS is different. There is a tool available which estimated the amount of space in main memory and on hard disk necessary for running the problem. It is only the size of the so-called database which is important. Most of the time, a relative small value was used here. For the adaptation runs, however, we had to increase the reserved memory for the database to 512 MB. With the exception of that case, no swapping occurred for the iterative solvers.

Time FEMLAB integrates geometry definition, mesh generation, initial value generation, numerical solution, and post-processing into one module. Therefore, we considered one test case as consisting of exactly these components. Similarly, the execution of one ANSYS batch file having the same sub-tasks was considered as one test case. The cpu time of the FEMLAB execution was measured using MATLAB's `cputime` command. ANSYS provides the cpu time explicitely in its output.

Another problem is concerned with FEMLAB 3.1i in 3D. The grid generator contains a certain randomness. So it is not possible to run the same test case twice. The number of degrees of freedom differs from run to run.

The resolution of the hardware clock is 0.01 s. In the tables below, we provided the cpu time with this accuracy. According to different states of the operating system, such an accuracy is not reproducible. For execution times below 100 s, the reproducible time granularity is around 0.1 s. For longer runs, an accuracy of at least one second can be guaranteed.

Another strategy was to mimic the average user. Both programmes have a lot of parameters which can be used to tune their behaviour. We did not try to tailor these parameters for maximal performance. Instead, the standard parameter settings were always used with the exceptions of those indicated explicitely. There is one exception to this rule: The termination criteria for the linear iterations use a scaled error estimation in FEMLAB and a scaled residual in ANSYS, respectively. In order to obtain criteria on par, the tolerance in ANSYS was changed to a value comparable to FEMLAB.

2.3 Test Environment

Hardware	Pentium 4, 2.4 GHz, 1 GB RAM (Dell Precision 340)
Operating system	SuSE Linux 9.1, kernel 2.6.5-7.155.29-default
LINPACK 1000	gnu f77 3.2: 216 MFlops; ifc 7.0: 223 MFlops
MATLAB 6.5	build 180913a
MATLAB 7.0.1	build 24704
FEMLAB 2.3	build 145
FEMLAB 3.1i	build 163
ANSYS 9.0	SP1, University Advanced

2.4 Some Comments on ANSYS

- The following elements were used in ANSYS:

element	description	FEMLAB equivalent
SOLID45	hexahedral trilinear element (Q1); 3 dof per node	none
SOLID95	hexahedral triquadratic element (Q2) of serendipity class; 3 dof per node	none
SOLID72	tetrahedral linear element (P1); 3 dof per node	Lag1
SOLID92	tetrahedral quadratic element (P2); 3 dof per node	Lag2
SOLID5	hexahedral trilinear element (Q1); 6 dof per node	none
SOLID98	tetrahedral quadratic element (P2); 6 dof per node	Lag2
SOLID226	hexahedral triquadratic element (Q2) of serendipity class; 4 dof per node	none
SOLID227	tetrahedral quadratic element (P2); 4 dof per node	Lag2

- The standard error tolerance in FEMLAB is 10^{-6}. If the linear iterative solvers are used as the primary solver in linear problems, the actual tolerance in the residual is set to that value divided by 400. Therefore, we used $10^{-6}/400$ as tolerance for the linear iterative solvers in ANSYS.

- ANSYS has a rather advanced memory management which tries to minimise the footprint in main memory. Therefore, much of the intermediate data are written to the hard disk. It is only the absolute minimum of information which is held in main memory. Besides that, some linear solvers have the option of working out-of-core. In order to keep things

comparable we did not use the out-of-core option. For the sparse solver, we explicitely required to work in-core. Since the university option of ANSYS is limited to 128000 nodes, this was seldom a restriction.

3 Results in Detail

The results of the benchmark computations are presented in tables. Each run is characterised by a certain parameter which is explained near the respective tables and used as a label in every row. The next rows contain the number of degrees of freedom (unknowns), the computed values of the targets, and the computation time (time). The last rows of every table contain these target values.

Remark:

- Cases marked with (*) lead to swapping but could be run. The memory requirements were less than 2GB in that case.

3.1 The NAFEMS Benchmark LE10

The Performance of FEMLAB 3.1i

The linear solver is `cg` with the preconditioner `gmg`. The equilibrium formulation (coefficient form) is used. All other parameters are standard.

(0) Before really trying to solve the model for the target value we carried out some experiments in order to find the best setting for the linear algebra routines, that is essentially the linear solver. We ran all available (meaningful) solvers in FEMLAB 3.1i with second order Lagrangian elements and the grid settings `normal` as well as

```
hmaxfact = 0.5, hcutoff = 0.001, hgrad = 1.5, hcurve = 0.2
```

denoted further on as `private`. The system was marked as being symmetric.

solver	parameter	normal			private		
		unknowns	iter	time	unknowns	iter	time
umfpack		9090		5.28	(*)61050		82.70
spooles		9468		5.44	60630		138.99
taucs		9222		4.47	60540		44.94
cg/umfpack	10^{-4}	9168	3	5.87			
	10^{-5}				(*)60843	3	87.34
cg/spooles	10^{-2}	9144	41	6.05	60933	94	116.54
	10^{-3}	9276	14	5.80	60210	32	110.87
	10^{-4}	9138	7	5.83	60357	10	136.15
cg/taucs	10^{-2}	8907	42	4.70	60345	96	33.64
	10^{-3}	8904	14	5.46	60630	41	43.84
	10^{-4}	9138	6	8.14	60558	13	105.80
cg/luinc	10^{-3}	9165	37	10.06	60921	88	277.08
	10^{-4}	8970	13	14.34	61209	43	656.27
cg/amg	1	9534	59	8.47	60630	105	95.10
	2	9459	58	8.38	61029	106	97.79
	3	9366	58	8.25	60753	105	95.63
	4	9234	52	13.74	60489	99	181.18
	5	9189	49	15.39	60822	96	250.46
	6	9270	49	15.93			
	7	9288	44	17.42			
	8	9192	43	18.00			
	9	9096	42	28.27			
	10	9291	41	31.00			
cg/gmg	umfpack	9126	7	7.01	61011	8	30.62
	spooles	9246	8	7.04	59316	7	28.36
	taucs	9519	7	6.99	60189	7	27.85
cg/ssor		8793	106	7.43	62082	205	83.05
cg/jac		9378	8865	127.73			

Remarks:

- The column `solver` denotes the solver according to the notation in FEMLAB. The second part of the conjugate gradient solver denotes the preconditioner.
- The column `parameter` contains parameters which were given to the preconditioner:

  `umfpack, spooles, taucs` drop tolerance;

  `amg` amgauto;

  `gmg` csolver.
- For larger drop tolerances than the one given, `cg/umfpack` converged only very slowly or not at all.

- the `luinc` preconditioner cannot use the symmetry of the problem.
- The multigrid based preconditioners used by far the lowest amount of memory.

As a consequence, the following tests were done using the conjugate gradient method together with the geometric multigrid preconditioner. We did not change any default parameters.

(A) The first test is devoted to a test of different grids generated by the mesh generator.

grid	unknowns	σ_x	σ_y	σ_z	time
normal	9087	−0.0972	−5.4593	−1.2176	10.95
normal (1)	64323	0.0089	−5.3213	−1.0489	32.52
fine	16170	−0.0560	−5.4753	−1.0680	10.98
finer	42321	0.0793	−5.2532	−0.9593	23.00
extra fine	167028	0.0610	−5.3780	−0.9889	90.02
private	60915	0.0097	−5.3590	−1.0110	32.58
		0.00	−5.38	−1.00	

Remarks:

- The mesh case `normal (1)` is generated by using the `normal` parameter set plus one refinement with `rmethod = 'regular'`.
- The mesh case `private` was constructed in order to obtain a better approximation of the curved surfaces. The chosen parameters are: `hmaxfact = 0.5, hcutoff = 0.001, hgrad = 1.5, hcurve = 0.2`.
- Finer grids did not fit into memory.

(B) The following test concerns the behaviour of different elements. The mesh case is `fine`.

elements	unknowns	σ_x	σ_y	σ_z	time
Lag1	2439	−0.8164	−5.1222	−1.6222	9.01
Lag2	16356	−0.0559	−5.4801	−1.0669	10.82
Lag3	51621	0.0082	−5.3080	−0.9490	40.26
Lag4(*)	118695	0.0717	−5.3006	−0.9376	193.84
		0.00	−5.38	−1.00	

Remarks:

- Please keep in mind that the mesh differs slightly from run to run because FEMLAB's mesh generator uses some randomness for generating the initial triangulation.

(C) Next we test pure mesh convergence by using the regular mesh refinement starting from a relatively coarse mesh. In order to fit in memory, first order Lagrangian elements are used. So the mesh case is `normal` and `rmethod = 'regular'`.

refinements	unknowns	σ_x	σ_y	σ_z	time
0	1437	−0.6859	−4.4601	−0.9480	8.42
1	9408	−0.5041	−5.4390	−1.1439	8.83
2	66549	−0.3665	−5.3936	−1.1397	46.79
3(*)	478659	−0.2155	−5.3745	−1.0815	313.42
		0.00	−5.38	−1.00	

(D) This test concerns the quality of the automatic grid adaption procedure. We start with the `normal` grid settings and use quadratic Lagrangian elements.

ngen	unknowns	σ_x	σ_y	σ_z	time
0	9084	−0.0761	−5.3901	−1.1725	7.10
1	19065	−0.0791	−5.4517	−1.2000	19.17
2	42945	−0.2151	−5.4665	−1.3102	51.35
3	97983	−0.2761	−5.4002	−1.2878	126.36
4	248835	0.0227	−5.2919	−1.0329	353.97
5	445179	0.0716	−5.2395	−0.9008	690.84
		0.00	−5.38	−1.00	

Remarks

- The computation time for level `ngen` contain the times for all previous levels `ngen-1` etc.
- The previous example with almost half a million unknowns fits completely into main memory.
- FEMLAB 3.1i provides only the mean square norm as an error functional. This amount to using the deformations as adaption criterion. Opposed to that, we are interested in the stresses which can be better controlled by using the energy functional.

A Comparison of FEMLAB 2.3, FEMLAB 3.1i, and ANSYS

(0) Before really trying to solve the model for the target value we carried out some experiments in order to find the best setting for the linear algebra routines, that is essentially the linear solver in ANSYS. We ran all available (meaningful) solvers with second order Lagrangian elements (`SOLID92`) using the grid settings `default` and

```
edgmx=0.121225
DESIZE,,,9999,,,,edgmx,,
```

denoted further on as `moderate`.

solver	`default`			`moderate`		
	unknowns	iter	time	unknowns	iter	time
front	12143		3.58	(*)60107		94.41
sparse	12143		3.36	60107		43.10
jcg	12143	?	4.31	60107	?	30.26
iccg	12143	147	4.06	60107	247	26.72
pcg	12143	65	2.32	60107	83	11.61

Remarks:

- The value of `edgmx` was chosen such that the number of degrees of freedom is roughly comparable to the FEMLAB runs.
- In the ANSYS university option there are no other solvers available.

As a consequence, the following tests of ANSYS were run using the pcg solver whenever possible.

(A) We start with comparing standard grid generations.

FEMLAB 3.1i

grid	unknowns	σ_x	σ_y	σ_z	time
normal	9087	−0.0972	−5.4593	−1.2176	10.95
normal (1)	64323	0.0089	−5.3213	−1.0489	32.52
fine	16170	−0.0560	−5.4753	−1.0680	10.98
finer	42321	0.0793	−5.2532	−0.9593	23.00
extra fine	167028	0.0610	−5.3780	−0.9889	90.02
private	60915	0.0097	−5.3590	−1.0110	32.58
		0.00	−5.38	−1.00	

FEMLAB 2.3

grid	unknowns	σ_x	σ_y	σ_z	time
normal	9234	−0.4017	−5.2353	−1.0742	11.42
normal (1)	65541	−0.1645	−5.3016	−1.0270	65.77
fine	19086	0.0419	−5.4023	−0.9677	19.80
finer	43620	0.00793	−5.4243	−1.0828	45.94
extra fine (*)	168867	0.1014	−5.3535	−0.9771	242.00
private	67917	−0.0259	−5.3961	−0.9996	76.45
		0.00	−5.38	−1.00	

(B) This test is devoted to test different element types. Besides that, the mesh generated is implicitely tested by these experiments.

FEMLAB 3.1i The mesh case is `fine`.

elements	unknowns	σ_x	σ_y	σ_z	time
Lag1	2439	−0.8164	−5.1222	−1.6222	9.01
Lag2	16356	−0.0559	−5.4801	−1.0669	10.82
Lag3	51621	0.0082	−5.3080	−0.9490	40.26
Lag4(*)	118695	0.0717	−5.3006	−0.9376	193.84
		0.00	−5.38	−1.00	

ANSYS The two following experiments use both `VMESH` as well as `VSWEEP` for mesh generation. In case of hexahedral elements and `VMESH`, is was required to generate hexahedrons by using the option

```
MSHAPE,0,3
```

The first table contains results without adapting any parameters.

elements	meshing	unknowns	σ_x	σ_y	σ_z	time
s72	VMESH	4568	−1.0838	−6.6629	−1.7659	3.46
s92	VMESH	12143	−0.2143	−5.4542	−1.0587	2.32
s45	VMESH	1114	0.0727	−5.2959	−0.8638	0.59
s95	VMESH	1514	−0.7220	−5.2919	−1.1531	0.78
s45	VSWEEP	4604	−0.3146	−5.4944	−1.1390	1.53
s92	VSWEEP	8878	−0.1281	−5.4727	−1.0906	2.38
			0.00	−5.38	−1.00	

Next we tried to adapt the maximal element size such that roughly the same number of degrees of freedom was used for linear elements.

elements	meshing	ESIZE	unknowns	σ_x	σ_y	σ_z	time
s72	VMESH	0.11375	22924	−0.8790	−5.4556	−1.3487	19.89
s92	VMESH	0.11375	85000	−0.1481	−5.3762	−1.0290	16.51
s45	VMESH	0.0975	24376	−0.2744	−5.5450	−1.1418	6.53
s95	VMESH	0.0975	94270	−0.0780	−5.4036	−1.0181	24.55
s45	VSWEEP	0.08125	21540	−0.2684	−5.5424	−1.1433	6.02
s92	VSWEEP	0.08125	83289	−0.0659	−5.4085	−1.0151	21.32
				0.00	−5.38	−1.00	

(C) It would be interesting to compare the mesh adaption procedure if the energy functional in FEMLAB 2.3 is used. Note that ANSYS uses always the energy norm in error estimates.

FEMLAB 2.3 We started with the `normal` mesh and second order Lagrange elements.

ngen	unknowns	σ_x	σ_y	σ_z	time
0	9234	−0.4017	−5.2353	−1.0742	12.57
1	40395	−0.3698	−5.4692	−1.0178	55.84
2 (*)	183780	−0.3675	−5.4179	−1.0202	275.65
		0.00	−5.38	−1.00	

Remarks:

- Because of memory restrictions, it was not possible to test the other error indicators in FEMLAB 2.3.

ANSYS The ANSYS elements `SOLID45`, `SOLID92`, and `SOLID95` allow for grid adaption. In the following experiments, there are no restriction on the grid given with the only exception of requiring hexahedrons for hexahedral elements.

elem	target	nsoln	unknowns	σ_x	σ_y	σ_z	time
s92	15%	1	2489	−0.2143	−5.4542	−1.0587	2
		2	69829	−0.4101	−5.3285	−1.1556	19
		3	136739	−0.2902	−5.3358	−1.0714	60
	5%	1	2489	−0.2143	−5.4542	−1.0587	2
		2	197182	−0.2711	−5.4345	−1.2084	49
s95	22%	1	1514	−0.7220	−5.2919	−1.1531	1
		2	21134	−0.3294	−5.4380	−1.1307	7
		3	273782	−0.0570	−5.4108	−1.0171	118
	5%	1	1514	−0.7220	−5.2919	−1.1531	1
		2	310468	−0.1039	−5.3693	−1.0153	138
s45	25%	1	1114	0.0727	−5.2925	−0.8638	1
		2	26780	−0.3029	−5.5009	−1.1026	9
(*)		3	335217	−0.1486	−5.4761	−1.0712	164
	15%	1	1114	0.0727	−5.2925	−0.8638	1
		2	78310	−0.2640	−5.4872	−1.0860	25
				0.00	−5.38	−1.00	

Remarks:

- The error control was restricted to the structural variables. The target value was selected such that at least some refinements could be carried out without exceeding the limit on the number of nodes.
- The regridding strategy is very aggressive compared to FEMLAB such that the number of elements grows rapidly in every refinement step.

3.2 A Cantilever Beam Under Large Deformations

The aim of this test is essentially to estimate the performance of the nonlinear solver. The geometry of the problem is so simple that the mesh generators are not expected to have any problems. While in the linear examples the performance of the solver for linear algebraic systems of equations dominate the runtime we expect a much stronger influence of the assembly process as well as of the control strategies of the nonlinear solver.

The Performance of FEMLAB 3.1i

In accordance with the experiences with the NAFEMS benchmark problem we chose the conjugate gradient iterative solver together with the geometric multigrid preconditioner as the linear solver. With the exception of first order Lagrangian elements, order reduction was used in order to obtain the "coarse grid" operators. In the case of first order elements, grid coarsening was used. As in the linear case, the nonlinear tolerance was set to 10^{-6}.

(A) Here we test the easier case $F = 10^{10}$. Although the full Newton method converges very fast (usually within 4 to 5 steps), it is often possible to save a considerable amount of computation time if a two-step strategy is used: First, an approximation of the solution on a very coarse grid is computed. Interpolating this approximation to a finer grid, a good initial guess for that grid is obtained. Such a strategy is suggested by FEMLAB 3.1i. In the following table, the column `iter` provides informations about the number of iterations in this strategy: "a/b" means that a iterations on the coarsest mesh (using the direct linear solver `spooles` and quadratic Lagrangian elements independent of the final element order) are performed. The number of degrees of freedom on the coarse grid is about 3600, varying a little bit because of the randomness in the mesh generator. The number of iterations on the finer grid is b. Note that, for linear elements, such a strategy did not work. The computation time indicated is the overall time used. The column `iter` indicates the cumulative number of linear conjugate gradient steps.

element	unknowns	u	v	w	$iter$	lin	time
Lag1	570	-0.007852	-2.2191	2.0966	$-/4$	31	14.50
	3594	-0.007878	-3.1775	2.9790	$-/4$	27	20.96
	24807	-0.016041	-3.6261	3.3899	$-/4$	21	89.83
	185691	-0.021128	-3.7797	3.5343	$-/4$	17	716.92
(*)	520329	-0.023129	-3.8065	3.5577	$-/4$	18	1848.38
Lag2	3432	-0.017537	-3.7407	3.4980	$-/4$	28	18.53
	22515	-0.021036	-3.8210	3.5695	$5/3$	19	60.77
	176727	-0.022398	-3.8366	3.5842	$5/3$	20	431.30
(*)	518757	-0.022729	-3.8392	3.5867	$5/3$	24	1281.36
Lag3	10476	-0.021275	-3.8266	3.5748	$5/3$	17	46.26
	79521	-0.022372	-3.8370	3.5846	$5/3$	19	282.38
(*)	239907	-0.022704	-3.8391	3.5867	$5/3$	20	872.87
Lag4	24879	-0.022146	-3.8350	3.5827	$5/3$	19	117.86
	77271	-0.022290	-3.8381	3.5848	$5/3$	20	400.76
Lag5	46473	-0.022663	-3.8382	3.5858	$5/3$	23	285.80
(*)	111783	-0.022975	-3.8402	3.5878	$4/3$	20	990.21
		-0.0232	-3.842	3.589			

Remarks:

- We used the standard strategy for the geometric multigrid preconditioner: Effectively, it is the *p*-version of the two-grid method (with the exception of first order Lagrangian elements where the *h*-version is used). On the coarse approximation, a direct solver is used. This strategy requires relatively large amounts of memory. Similarly, the low number of linear iterations indicates that the preconditioner is "too good". The assembly process dominates the computation time.

- FEMLAB 3.1i has the option of using a Broyden method. In the present example, the number of undamped Newton steps is rather low such that the gain in computing time is only marginal. The following table is illustrating ($F = 10^{10}$). It corresponds to linear Lagrangian elements.

unknowns	iter	Broyden steps	time
570	4		14.50
594	5	3	9.38
3594	4		20.96
3597	5	2	17.27
24807	4		89.83
25517	5	2	79.57
185691	4		716.92
174159	6	2	597.33
520329	4		1848.38
551739	5	2	1794.19

(B) The case of $F = 10^{11}$ is much harder to solve. Although FEMLAB 3.1i is very robust and can solve this problem without any imbedding or similar strategies the number of damped Newton iteration is rather large. Therefore, we decided to use the same strategy as in the case of $F = 10^{11}$.

element	unknowns	u	v	w	*iter*	lin	time
Lag1	558	−0.51162	−22.935	16.046	−/10	75	29.45
	3399	−1.00933	−32.375	19.294	−/12	68	52.08
	23403	−1.35814	−36.315	20.424	−/13	53	263.68
	171063	−1.56273	−37.627	20.791	−/14	39	2196.23
(*)	520803	−1.65606	−37.849	20.880	−/13	31	5854.03
Lag2	3633	−1.37813	−37.408	20.663	−/13	77	54.10
	24909	−1.58953	−37.988	20.883	13/3	20	86.21
	173907	−1.68432	−38.101	20.952	13/3	19	443.92
(*)	545979	−1.70860	−38.121	20.970	13/3	20	1366.83
Lag3	11007	−1.59424	−38.018	20.893	13/3	15	70.09
	71878	−1.68538	−38.102	20.954	13/3	15	272.40
(*)	238785	−1.70651	−38.120	20.971	13/3	19	901.30
Lag4	24807	−1.66321	−38.086	20.941	13/3	16	133.29
	80187	−1.69271	−38.108	20.952	13/3	19	426.85
Lag5	42498	−1.70210	−38.111	20.960	13/3	19	259.23
(*)	79683	−1.70767	−38.115	20.966	13/3	19	582.93
		−1.74	−38.14	20.98			

A Comparison of FEMLAB 3.1i and ANSYS

(A) Besides the solid elements given below, the torsion force was applied via a surface element `surf154`.

The following comparisons are only made with ANSYS because the structural mechanics toolbox of FEMLAB 2.3 does not allow for nonlinear effects. The first table concerns the simple problem with $F = 10^{10}$.

element	esize	unknowns	u	v	w	$iter$	time
s45	0.018	3970	−0.009667	−3.7540	3.5101	5	4.58
	0.009	23760	−0.017202	−3.7989	3.5801	6	33.67
	0.00435	180918	−0.013856	−3.8342	3.5803	6	298.83
	0.0035	358266	−0.014517	−3.8359	3.5841	6	618.52
s72	0.1	180	−0.15557	−7.3234	8.6074	6	0.77
	0.023	3282	−0.17814	−3.4961	4.7506	5	8.50
	0.012	25170	0.003599	−3.8557	3.8485	5	89.51
	0.007	126834	0.002750	−3.7872	3.7802	5	636.54
s92	0.1	462	−0.007837	−4.0064	3.7270	5	0.73
	0.038	4128	0.082689	−3.6244	2.7571	6	3.95
	0.02	26529	−0.014219	−3.8415	3.5914	6	27.81
	0.01	162306	−0.014149	−3.8480	3.5915	6	195.59
	0.008	318138	−0.013464	3.8569	3.5925	6	417.42
s95	0.1	180	−6.4293	−4.8667	3.6783	6	0.51
	0.03	2688	−0.016743	−4.0063	3.7743	6	3.44
	0.028	4590	−0.14565	−3.5058	4.1357	6	6.35
	0.015	23760	−0.0072307	−3.8980	3.5923	6	36.55
	0.0071	182700	−0.014842	−3.8506	3.5962	6	335.14
	0.0055	379620	−0.015091	−3.8476	3.5943	6	753.98
			−0.0232	−3.842	3.589		

Remarks:

- The grid is generated by either the `VSWEEP` or `VMESH` commands depending on the type of element.

- The parameter `ESIZE` is chosen in such a way that the number of degrees of freedom is comparable to those in the FEMLAB runs. The ANSYS version used is restricted to handle at most 128000 nodes. For the largest number of degrees of freedom, the maximal number of nodes is almost reached.

- Even for the largest number of unknowns, the memory consumption was below 1 GB. But ANSYS wrote up to 0.5 GB of data to the hard disk during the run such that, for a larger number of degrees of freedom, the wall clock time is much larger than the pure cpu time.

(B) This is the harder case $F = 10^{11}$.

element	esize	unknowns	u	v	w	*iter*	time
s45	0.018	3970	−0.73849	−37.577	20.680	13	15.02
	0.009	23760	−1.0027	−38.048	21.064	13	105.46
	0.00435	180918	−1.0894	−38.448	21.016	14	980.22
	0.0035	358266	−1.1160	−38.479	21.047	14	2066.23
s92	0.1	462	−0.46636	−39.489	20.936	12	1.26
	0.038	4128	−0.55373	−36.121	12.231	13	10.55
	0.02	26529	−0.99018	−38.362	20.703	13	78.43
	0.01	162306	−1.1152	−38.550	21.002	14	609.95
	0.008	318138	−1.1327	−38.670	21.034	14	1292.02
s95	0.1	180	−106.39	−42.364	234.92	15	0.92
	0.03	2688	−0.98139	−39.418	21.320	13	9.07
	0.028	4590	−2.0491	−35.219	25.314	13	17.23
	0.015	23760	−1.0344	−38.970	20.853	13	102.04
	0.0071	182700	−1.1505	−38.630	21.087	14	1029.41
	0.0055	379620	−1.1647	−38.618	21.094	14	2304.92
			−1.74	−38.14	20.98		

Remarks:

- The same comments as in the case $F = 10^{10}$ apply.
- In order to get Newton's method to converge the line search option was set to on. Nevertheless, we were unable to obtain a converged solution for the element `SOLID72`.

3.3 Natural Frequencies of a Piezoelectric Transducer

3.3.1 The short circuit case

The Performance of FEMLAB 3.1i

This example was run using the eigenvalue analysis option of the piezoelectric application mode. The implementation is straightforward. The model description does not prevent a rigid body motion in z-direction. Therefore, there is always an eigenmode with frequency 0 [Hz] present. In the following computations, 11 eigenvalues are computed. Similarly as in ANSYS, an shift of 50 [kHz] was used.

Especially when using higher-order elements, this examples requires a huge amount of memory. Therefore, the basic grid was chosen to be based on the `extra coarse` grid generation parameters.

(A) The first test concerns the performance of different Lagrangian elements. The linear system solver is `umfpack`.

element	unknowns	f_1	f_2	time
Lag1	288	69.3314	108.0887	1.35
Lag2	1596	65.1126	88.5408	11.46
Lag3	4680	65.1012	88.1432	49.36
Lag4	10316	65.1011	88.1367	21.09
Lag5	19344	65.1011	88.1364	58.35
		65.1011	88.1364	

(B) Here, we are interested in investigating the grid convergence. Because of memory restrictions, this could only be done for lower order elements. Starting out with the coarsest grid, the mesh is refined a number of times using the regular refinement method. This amounts to roughly halving the step size.

In contrast to the previous test, one must use an iterative linear solver. The most efficient combination (avoiding swapping) turned out to be `gmres` with the preconditioner `gmg`.

Lag1 elements

steps	unknowns	f_1	f_2	time
0	288	69.3314	108.0887	3.67
1	1592	65.3735	92.9170	7.21
2	10236	65.1895	89.3917	40.16
3	73956	65.1253	88.4931	451.02
		65.1011	88.1364	

Lag2 elements

steps	unknowns	f_1	f_2	time
0	1596	65.1126	88.5408	15.27
1	10316	65.1020	88.1679	73.85
2	73620	65.1012	88.1386	916.88
		65.1011	88.1364	

Lag3 elements

steps	unknowns	f_1	f_2	time
0	4680	65.1012	88.1432	56.06
1	32288	65.1011	88.1366	643.56
		65.1011	88.1364	

Remarks:

- It is tempting to use the eigenvalue/eigenvector approximations on a coarse grid as starting values on a finer grid. Unfortunately, we did not find a way to do so.

- The solution algorithm complains sometimes about the chosen preconditioner being bad. The incomplete LU preconditioners are either too memory demanding (`umfpack`, `spooles`) or very slow for the higher order Lagrangian elements (`luinc`).
- The problem at hand is very smooth such that Richardson extrapolation can be applied.

(C) In a last comparison, the grid is chosen such that the discretisation has roughly the same amount of degrees of freedom. Again, the iterative solver was used.

element	unknowns	f_1	f_2	time
Lag1	73956	65.1253	88.4931	451.02
Lag2	73620	65.1012	88.1386	916.88
Lag3	73752	65.1011	88.1365	1760.61
Lag4	74964	65.1011	88.1364	10176.09
Lag5	60424	65.1011	88.1364	6348.34
		65.1011	88.1364	

A Comparison of FEMLAB 3.1i, and ANSYS

For the example at hand we did not test FEMLAB 2.3 for two reasons: There is no piezoelectric application mode available such that its implementation is very cumbersome, and the expected memory consumption seems to prevent any meaningful comparisons.

Since this benchmark example is present in the ANSYS validation suite (test case VM175), a template for the ANSYS input file was available. This template was reduced to the bare minimum in order to compute the target values. The eigenvalue analysis was done using the block Lanczos solver (`LANB`). 10 eigenvalues in the range between 50 [kHz] and 150 [kHz] were required. The solution of the linear systems is done by the frontal solution method. As in all of our tests, the solution is requested to be performed completely in-core. ANSYS rejected other settings for the linear equation solver.

The elements tested are `SOLID5`, `SOLID226`, `SOLID98`, and `SOLID227`. The two first elements are hexahedral ones while the latter two are tetrahedral elements. All elements (with the exception of `SOLID5`) are second order accurate.

(A) A comparison on relatively coarse grids using direct solvers is given below.

FEMLAB 3.1i

element	unknowns	f_1	f_2	time
Lag1	288	69.3314	108.0887	1.35
Lag2	1596	65.1126	88.5408	11.46
Lag3	4680	65.1012	88.1432	49.36
Lag4	10316	65.1011	88.1367	21.09
Lag5	19344	65.1011	88.1364	58.35
		65.1011	88.1364	

ANSYS

element	unknowns	f_1	f_2	time
s5	132	66.4474	92.9997	0.26
s98	676	65.1458	89.3280	0.50
s226	448	65.1223	88.4182	0.39
s227	676	65.1458	89.3280	0.62
		65.1011	88.1364	

(B) The next comparison is devoted to estimate the performance of different elements types. In ANSYS, the `ESIZE` parameter was selected such that roughly the same number of degrees of freedom was achieved.

FEMLAB 3.1i

element	unknowns	f_1	f_2	time
Lag1	73956	65.1253	88.4931	451.02
Lag2	73620	65.1012	88.1386	916.88
Lag3	73752	65.1011	88.1365	1760.61
Lag4	74964	65.1011	88.1364	10176.09
Lag5	60424	65.1011	88.1364	6348.34
		65.1011	88.1364	

ANSYS

element	unknowns	f_1	f_2	time
s5	132	66.4474	92.9997	0.26
	19264	65.1381	88.2744	34.68
(*)	69720	65.1161	88.1945	354.01
s98	676	65.1458	89.3280	0.50
	13476	65.1019	88.1593	17.86
	32780	65.1014	88.1430	89.02
(*)	59472	65.1013	88.1395	246.72
s226	448	65.1223	88.4182	0.39
	5110	65.1017	88.1457	5.99
	34536	65.1012	88.1372	154.26
(*)	58964	65.1012	88.1368	409.17
s227	676	65.1458	89.3280	0.62
	13476	65.1019	88.1593	18.45
	32780	65.1014	88.1430	88.87
(*)	59472	65.1013	88.1395	244.01
		65.1011	88.1364	

3.3.2 The open circuit case

The Performance of FEMLAB 3.1i

This example was run using the eigenvalue analysis option of the piezoelectric application mode. The implementation in this case is not straightforward. There is no simple way to model the top electrode. In order to do so, the multiphysics feature must be used for modelling a floating electrode. This method is described in a technical paper [1]. Since this idea amounts to use weak constraints, there isn't any good preconditioner available for the iterative solver. So we are bound to use variants of the incomplete LU preconditioner.

The model description does not prevent a rigid body motion in z-direction. Therefore, there is always an eigenmode with frequency 0 [Hz] present. In the following computations, 11 eigenvalues are computed. Similarly as in ANSYS, an shift of 50 [kHz] was used.

Especially when using higher-order elements, this examples requires a huge amount of memory. Therefore, the basic grid was chosen to be based on the `extra coarse` grid generation parameters.

(A) On the coarsest used grid, the different elements give the following results. The solver is a direct one (`umfpack`).

element	unknowns	f_1	f_2	time
Lag1	302	86.161	112.266	1.77
Lag2	1634	79.931	93.898	2.91
Lag3	4830	79.808	93.584	12.33
Lag4	10622	79.807	93.577	24.93
Lag5	19566	79.806	93.576	71.50
		79.806	93.576	

(B) The second experiment concerns grid convergence. As mentioned above, an iterative linear solver (`gmres`) with an incomplete LU preconditioner (we chose `spooles`) must be used. The drop tolerance was set to $5 \cdot 10^{-4}$.

Lag1 elements

step	unknowns	f_1	f_2	time
0	302	84.991	109.559	1.96
1	1654	81.658	96.883	4.78
2	10502	80.427	94.517	49.60
3	73830	79.959	93.797	1470.62
		79.806	93.576	

Lag2 elements

step	unknowns	f_1	f_2	time
0	1650	79.947	93.927	4.86
1	10502	79.817	93.604	53.58
2	72822	79.807	93.579	1634.43
		79.806	93.576	

Lag3 elements

step	unknowns	f_1	f_2	time
0	4846	79.808	93.584	31.96
1	33162	79.806	93.577	747.20
		79.806	93.576	

(C) Compare now the approximation accuracy with different element types and a comparable number of degrees of freedom.

element	unknowns	f_1	f_2	time
Lag1	73830	79.959	93.797	1470.62
Lag2	72822	79.807	93.579	1634.43
Lag3	74661	79.806	93.577	1789.21
		79.806	93.576	

Remarks:

- Higher order elements which a comparable number of degrees of freedom could not be run because of memory limitations.

A Comparison of FEMLAB 3.1i, and ANSYS

The experimental setting is exactly as before in the short circuit case.

(A) A comparison on relatively coarse grids using direct solvers is given below.

FEMLAB 3.1i

element	unknowns	f_1	f_2	time
Lag1	302	86.161	112.266	1.77
Lag2	1634	79.931	93.898	2.91
Lag3	4830	79.808	93.584	12.33
Lag4	10622	79.807	93.577	24.93
Lag5	19566	79.806	93.576	71.50
		79.806	93.576	

ANSYS

element	unknowns	f_1	f_2	time
s5	133	84.261	96.988	0.26
s98	677	80.185	94.417	0.56
s226	449	79.922	93.811	0.39
s227	677	80.185	94.417	0.50
		79.806	93.576	

(B) The next comparison is devoted to estimate the performance of different elements types. In ANSYS, the `ESIZE` parameter was selected such that roughly the same number of degrees of freedom was achieved.

FEMLAB 3.1i

element	unknowns	f_1	f_2	time
Lag1	73830	79.959	93.797	1470.62
Lag2	72822	79.807	93.579	1634.43
Lag3	74661	79.806	93.577	1789.21
		79.806	93.576	

ANSYS

element	unknowns	f_1	f_2	time
s5	133	84.261	96.988	0.26
	19265	79.920	93.677	35.38
(*)	69721	79.853	93.619	349.16
s98	677	80.185	94.417	0.56
	13477	79.814	93.596	17.99
	32781	79.809	93.583	89.52
(*)	59473	79.808	93.580	265.47
s226	449	79.922	93.811	0.39
	5111	79.810	93.585	5.94
	34537	79.807	93.578	151.50
(*)	58965	79.807	93.577	405.82
s227	677	80.185	94.417	0.50
	13477	79.814	93.596	19.10
	32781	79.809	93.583	85.54
(*)	59473	79.808	93.580	269.76
		79.806	93.576	

3.4 A Nearly Incompressible Material

The Performance of FEMLAB 3.1i

The linear solver is `cg` with the preconditioner `gmg`. The equilibrium formulation (coefficient form) is used. All other parameters are standard.

(A) The first test is devoted to a test of different grids generated by the mesh generator.

grid	unknowns	σ_x	σ_z	time
normal	14163	−0.0080	2.1875	12.00
normal (1)	98865	−0.023	2.1609	61.27
fine	22344	−0.1261	2.0499	14.00
finer	49023	−0.1328	2.0338	39.53
extra fine	160512	−0.0563	2.1221	110.38
private	60939	0.0114	2.1762	40.11
		0.000	2.164	

(B) In a next step, consider different Lagrangian elements on (roughly) the same grid. The first table uses the mesh settings `fine`.

elements	unknowns	σ_x	σ_z	time
Lag1	3447	0.2732	2.4971	5.6
Lag2	21879	−0.0994	2.0840	13.24
Lag3	64794	0.0030	2.1635	50.11
		0.000	2.164	

Here, we do the same test using the mesh settings `normal`.

elements	unknowns	σ_x	σ_z	time
Lag1	2331	2.3087	4.4052	7.52
Lag2	14037	−0.0411	2.1549	9.15
Lag3	41274	−0.0225	2.1417	31.19
Lag4	95085	−0.0075	2.1555	127.71
		0.000	2.164	

Remarks:

- An approximation with even higher order elements was not possible because of memory restrictions. This is even more remarkable as the number of degrees of freedom was not exceptional high.

(C) The next two tables contain the results of the adaption procedure. The grids on refinement level 0 are constructed according to the `normal` mesh generation setting.

The first test was done with second order Lagrangian elements.

ngen	unknowns	σ_x	σ_z	time
0	14958	−0.0603	2.1772	9.95
1	26541	−0.0050	2.1583	27.40
2	44961	0.0094	2.1712	70.24
3	66195	−0.0539	2.1195	160.69
4	98049	−0.0692	2.1027	246.49
5	153747	−0.0374	2.1311	824.25
		0.000	2.164	

Here, the same results are provided using third order Lagrangian elements.

ngen	unknowns	σ_x	σ_z	time
0	42744	0.0017	2.1654	32.19
1	97629	−0.0144	2.1480	231.69
2	233286	0.0012	2.1646	2112.60
		0.000	2.164	

Remarks:

- In the last case (third order Lagrangian elements and `ngen = 2`), the number of linear iterations increased to 216 thus indicating a bad preconditioner. This is in contrast to *all* other test cases where the number of linear iterations ranges between 8 and 15.

A Comparison of FEMLAB 2.3, FEMLAB 3.1i, and ANSYS

(A) We test the performance of different element types and different setting for the grid quality.

ANSYS The grids for the following test were generated by using different values for the parameter `ESIZE`. Since the domain has a very sharp edge, many warnings about elements with small angles appeared. ANSYS provides the option of coarsening the mesh in order to obtain better shaped elements. When trying meshes with and without coarsening we found that without coarsening the target values were approximated more accurate than with coarsening. Therefore, the following experiments were done without coarsening.

For the tetrahedral elements, the grid was generated using `VMESH` with `SMRTSIZE,1` and different settings for `ESIZE`. For tetrahedral elements, `VSWEEP` was used instead.

element	ESIZE	unknowns	σ_x	σ_z	time
s72	0.68	2398	−0.9168	−2.4496	2.71
	0.6	3398	−0.6926	−2.3044	3.56
s92	0.601	14023	−0.0248	−2.1755	6.78
	0.52	21583	−0.0459	−2.1627	10.93
	0.3856	49186	−0.0774	−2.1842	23.49
	0.302	98697	−0.0625	−2.1815	50.68
	0.25	172637	−0.0930	−2.2094	87.10
	0.195	358188	−0.0717	−2.1548	206.99
s45	0.7	1234	−0.1915	−2.3280	0.96
s95	0.7	4276	−0.4318	−2.5433	3.18
			0.000	2.164	

Remarks:

- It was not possible to use smaller values of `ESIZE` for the hexahedral elements because the mesh generator crashed with “segmentation violation”.
- The `ESIZE` parameter of the second order Lagrangian element s92 was chosen such that the number of degrees of freedom is comparable to the similar runs of FEMLAB 3.1i.

(B) This test tries to estimate the adaption procedure using the energy error indicator.

ANSYS For the present problem, it is only meaningful to test the tetrahedral element s92. The hexahedral elements are not well-suited for the domain in question.

target	nsoln	unknowns	σ_x	σ_z	time
1%	1	3230	−0.1550	2.2002	1.90
	2	2038	−0.0105	2.1614	3.21
0.1%	1	3230	−0.1550	2.2002	1.90
	2	14379	−0.0026	2.1655	8.46
	3	9041	0.0106	2.1614	13.55
	4	11683	−0.0770	2.1640	19.87
	5	7924	0.0626	2.1734	24.91
	6	10223	−0.0175	2.1610	30.61
	7	9473	−0.0047	2.1670	35.81
	8	7957	−0.0083	2.1622	40.72
	9	9117	−0.0029	2.1619	45.62
0.01%	1	3230	−0.1550	2.2002	1.90
	2	17663	−0.0007	2.1661	10.13
	3	128975	−0.0007	2.1646	99.26
0.02%	1	3230	−0.1550	2.2002	1.90
	2	16331	−0.0007	2.1662	9.53
	3	56949	−0.0007	2.1640	40.96
	4	181849	−0.0037	2.1640	156.61
			0.000	2.164	

Remarks:

- For all tolerances, the required accuracy was not met. As can be seen in the case of a tolerance of 0.1%, there is no convergence. For the smaller tolerances, we arrived at the limit of 128000 nodes which our ANSYS version could handle.

FEMLAB 2.3 This test uses the mesh generation option `normal` and second order Lagrangian elements

ngen	unknowns	σ_x	σ_z	time
0	59976	−0.2379	1.9708	68.73
1 (*)	179760	−0.1110	2.0593	282.61
		0.000	2.164	

Remarks:

- The grid generator seems to do a bad job. Even if the number of degrees of freedom is relatively large, the results are not very accurate.

Let us try private mesh generation parameters:

```
hmaxfact = 1.5, hcutoff = 0.001, hgrad = 1.5, hcurve = 0.2
```

ngen	unknowns	σ_x	σ_z	time
0	52134	-0.2103	1.9738	57.96
1	151992	-0.0999	2.0659	231.01
		0.000	2.164	

Higher order elements are impossible to use because of memory limitations.

4 Conclusions

General Observations

1. For structural mechanics problems, the geometric multigrid preconditioner is a very efficient and robust preconditioner. This holds in both implementations (equilibrium equations and principle of virtual work, respectively). If applicable, the equilibrium equations seem to be slightly more robust.

2. With the standard settings, this preconditioner is "too good" in the sense that the number of linear iterations (especially when used in the nonlinear problem) is very low. This wastes both space (by having a too large coarse grid problem) and time (by requiring too many reassembly steps).

3. Compared too ANSYS, the assembly process in FEMLAB 3.1i is rather slow.

4. The memory allocation strategies in FEMLAB 3.1i and ANSYS are very different. While FEMLAB 3.1i allocates all its necessary data in random access memory (and thus relying on a clever swap algorithm of the underlying operation system), ANSYS keeps a lot of informations in external files on the hard disk thus trying to minimise the foot print in random access memory. Consequently, FEMLAB 3.1i gains a lot in performance with respect to wall clock time if plenty of memory is available.

5. Given the hardware used, the linear elements do not perform very well both in FEMLAB 3.1i and in ANSYS. The best compromise, in many cases, seems to be second order Lagrangian elements.

6. It is much more convenient and efficient to construct models in FEMLAB 3.1i than in ANSYS.

7. The old MATLAB-based version of FEMLAB is clearly ruled out by the new version. However, the adaption procedure in FEMLAB 2.3 is superior if memory restrictions allow its usage. The main problem in FEMLAB 3.1i in this context is the lack of an energy error estimator.

The NAFEMS Benchmark LE10

1. This benchmark tests the approximation quality of the grid generator as well as the performance of the linear solvers.

2. When using low order elements, mesh convergence can be observed.

3. It does not pay to used higher order elements. It seems to be more more important to have a good resolution of the boundary. The best results were obtained using second order Lagrangian elements.

4. The automatic mesh adaption procedure in FEMLAB 3.1i is misleading. It uses a criterion including only displacements while the target values consists of normal stresses. Application of the energy functional would be more adequate in this case.

5. In ANSYS, the automatic grid adaption bases on the energy error estimator. Convergence can be observed. It must be noted, however, that the adaption procedure is restricted to only a few elements and few analysis types.

6. For the same number of unknowns and comparable elements, ANSYS is often twice as fast as FEMLAB 3.1i.

A Cantilever Beam Under Large Deformations

1. FEMLAB 3.1i could solve both the simple and the harder case without any problems.

2. In the present example it pays to use higher order elements.

3. Surprisingly,the use of the Broyden method did not decrease the computation time considerably. Since there is no need to assemble the Jacobian of the system in that case, we expected a much higher efficiency.

4. ANSYS provide rather large errors for the deformation u. The reason is unknown.

Natural Frequencies of a Piezoelectric Transducer

1. As expected in the case of eigenvalue problems, the use of higher order Lagrangian elements gives rise to very accurate and fast approximations. The geometry is so simple that no errors arise from the geometry approximation.

2. The eigenvalue problem is the only one where the memory consumption of ANSYS is larger than those of FEMLAB 3.1i.This may be due to the fact that is is not possible to use an iterative solver together with the Lanczos algorithm.

A Nearly Incompressible Material

1. This test case includes two challenges: approximation of the domain and the material properties.

2. The behaviour of FEMLAB 3.1i does not differ essentially from that observed in the NAFEMS LE10 benchmark problem. Therefore, the conclusions drawn there are valid here, too.

3. It is not at all surprising that hexahedral elements are unusable for the present geometry. We would have expected that ANSYS is more robust when handling such situations.

4. The adaption procedure of FEMLAB 3.1i gave satisfactory results at the cost of long computation times. Surprisingly, the adaption procedure with the energy error estimator did a bad job in FEMLAB 2.3. Contrary to that, ANSYS' adaption procedure worked very well.

5 Summary

We have tested the structural mechanic toolbox version 3.1 and compared its performance to its predecessor version 2.3 and a state-of-the-art tool ANSYS 9.0.

The structural mechanics toolbox has now reached a state that it is capable of solving even three-dimensional problems on a machine with rather limited hardware ressources. It is definitely far beyond its predecessor.

Especially for problems where higher order elements can be used efficiently, accurate approximations can be obtained.

A critical problem with our tests is the speed of the core numerics – with the exception of the eigenvalue problems. The versatility, convenience of use, robustness, and generality of the FEMLAB approach takes its toll.

Acknowledgements The author likes to thank Fredrik Wredenberg for preparing all ANSYS batch files. Moreover, we would like to thank the Department of Solid Mechanics for their support using ANSYS.

References

[1] Comsol Support. Floating potential with integral constraints. Technical report, Comsol, Inc., 2005.

[2] D. Boucher, M. Lagier, and C. Maerfeld. Computation of the vibration modes for piezoelectric array transducers using a mixed finite element perturbation method. *IEEE Trans. Sonics and Ultrasonics*, SU-28(5):318–330, 1981.

[3] G.A.O. Davies, R.T. Fenner, and R.W. Lewis. *Background to Benchmarks*. NAFEMS, Glasgow, 1993.

[4] R.S. Lakes. Design considerations for negative Poisson's ratio materials. *ASME J. Mech. Design*, 115:696–700, 1993.

[5] Michael Hanke. Benchmarking FEMLAB 3.0a: Laminar Flows in 2D. PSCI Report 04:01, Royal Inst. of Technology, Stockholm, 2004.

[6] Olivier Verdier. Benchmark of FEMLAB, FLUENT and ANSYS. Preprints in Math. Sc. 2004:6, Lund University of Technology, Center for Mathematical Sciences, Lund, 2004.