DEPARTMENT OF APPLIED MATHEMATICS

INSTITUTE OF GEONICS Ostrava

Software modulus: Code for implementation of preconditioners

R. Kohut

TR 9604, Septemeber 1996

Institute of Geonics, Academy of Sciences of the Czech Republic
Studentská 1768, 708 00 Ostrava – Poruba, Czech Republic
phone: +42-69-691 86 51, fax: +42-69-691 94 52
e–mail: kohut@ugn.cas.cz.

Software modulus: Code for implementation of preconditioners.

Roman KOHUT

July 14, 1998

1 Introduction

For testing iterative solvers (preconditioners) for linear equations arising from FEM analysis of 3D linear elasticity problems we need the following codes:

- preprocessor for generating of test problems (benchmarks) TGRID
- code for assembling of the FE system of equations FEMSYS
- PCG–1 iterative solver for solving of system of linear equations
- code COMPFL for comparison of files

For the work with benchmarks and problems pased on rectangular grids, a special preprocessor TGRID was developed, (see [1]). It serves mainly for preparing rectangular grids (uniform or general) but for more complete testing of solvers it is also possible to prepare special nonrectangular grids by inclining some planes.

Preprocessor TGRID creates several files containing information about the grid, division of the domain into tetrahedra (all eight node bricks are divided into tetrahedra in a manner which can be seen from Fig. 1), information about material parameters, distribution of materials and boundary conditions. The files are following:

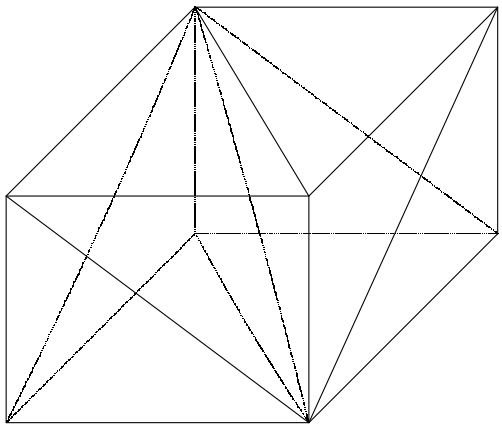

Figure 1: A division of eight-node brick into six tetrahedra

FV.G32 – FORMATTED ASCII file which contains the basic information about the solved problem, especially grid parameters NX, NY, NZ. (See also source code READFV or WRITEFV).

FMAT.G32 – FORMATTED ASCII file which contains material parameters and corresponding elastic constitutive matrices.

FX.G32 – BINARY file containing coordinates of nodes.

FEL.G32 – BINARY file containing information about tetrahedral elements. This is the basic file for the assembling of the stiffness matrix.

FR.G32 – BINARY file containing given surface forces.

FBC.G32 – BINARY file containing prescribed displacements. The value 1E9 corresponds to the displacements which are not given. The source codes needed for the forming of TGRID code can be seen from corresponding makefiles.

Make file for TGRID:

```
#  Makefile for TGRID for AIX
#  command:   make -f mktgrid

ne = tgrid
n0 = ing32acp
n2 = writefv
n3 = gmopencp
n1 = timecp
#
opt = -O3

$(ne): init $(n0).o  $(n1).o  $(n2).o  $(n3).o
     xlf  $(opt) -o $(ne) $(n0).o  $(n1).o  $(n2).o  $(n3).o

# ing32a
$(n0).o: $(n0).f
     xlf -D $(opt) -c $(n0).f 2> cerr || pg cerr
$(n0).f: $(n0).for
     grep -v "^"[eE]"" $(n0).for > $(n0).f
# writefv
$(n2).o: $(n2).f
     xlf -D $(opt) -c $(n2).f 2> cerr || pg cerr
$(n2).f: $(n2).for
     grep -v "^"[eE]"" $(n2).for > $(n2).f
# gmopencp
$(n3).o: $(n3).f
     xlf -D $(opt) -c $(n3).f 2> cerr || pg cerr
$(n3).f: $(n3).for
     grep -v "^"[eE]"" $(n3).for > $(n3).f
# timecp
$(n1).o: $(n1).f
     xlf -D $(opt) -c $(n1).f 2> cerr || pg cerr
$(n1).f: $(n1).for
     grep -v "^"[eE]"" $(n1).for > $(n1).f
```

```
#
init:
     -rm cerr            # '-' error is ignored
```

The FE system (the stiffness matrix and the load vector) is created by program FEMSYS. Input data for FEMSYS are taken from the files created by the preprocessor TGRID. The load vector includes contributions of surface forces (file FR.G32) and weight of material. Both matrix and right hand side (rhs) vector are finally modificated due to the prescribed boundary conditions (displacement).

Output files of FEMSYS are

FKBC.G32 – BINARY file containing stiffness matrix
FRBC.G32 – BINARY file containing the rhs vector .

These files are modificated due to prescribed displacement and the holes in the domain.

The complete matrix is read by cycle

```
DO 10   i=1, NN  (number of nodes)

READ (10)  (a (j),  j=1, 126 )

10 continue
```

where a contains 3 rows of matrix corresponding to one node.

The rhs vector is read by instruction
read (12) $(r(i), i = 1, ND)$, where ND is number of degrees of freedom.

The source codes needed for the forming of FEMSYS code can be seen from corresponding make file

Make file for FEMSYS

```
#  Makefile for femsys for AIX, version for Copernikus
#  command:    make -f mkfemsys

ne = femsys
```

```
n0 = mstrlecp
n1 = stiffcp
n2 = georecp
n3 = loadcp
n4 = priprcp
n5 = readfv
#
opt = -O3
#opt = -g -bloadmap:loadmap
lib = /u/lmm/libgem.a

$(ne): init $(n0).o $(n1).o $(n2).o $(n3).o $(n4).o $(n5).o
     xlf $(lib) $(opt) -o $(ne) $(n0).o $(n1).o $(n2).o $(n3).o $(n4).o \
      $(n5).o

# mstrlecp
$(n0).o: $(n0).f
     xlf -D $(opt) -c $(n0).f 2> cerr || pg cerr
$(n0).f: $(n0).for
     grep -v "^"[eE]"" $(n0).for > $(n0).f
# stiffcp
$(n1).o: $(n1).f
     xlf -D $(opt) -c $(n1).f 2> cerr || pg cerr
$(n1).f: $(n1).for
     grep -v "^"[eE]"" $(n1).for > $(n1).f
# georecp
$(n2).o: $(n2).f
     xlf -D $(opt) -c $(n2).f 2> cerr || pg cerr
$(n2).f: $(n2).for
     grep -v "^"[eE]"" $(n2).for > $(n2).f
# loadcp
$(n3).o: $(n3).f
     xlf -D $(opt) -c $(n3).f 2> cerr || pg cerr
$(n3).f: $(n3).for
     grep -v "^"[eE]"" $(n3).for > $(n3).f
# priprcp
$(n4).o: $(n4).f
```

```
     xlf -D $(opt) -c $(n4).f 2> cerr || pg cerr
$(n4).f: $(n4).for
     grep -v "^"[eE]"" $(n4).for > $(n4).f
# readfv
$(n5).o: $(n5).f
     xlf -D $(opt) -c $(n5).f 2> cerr || pg cerr
$(n5).f: $(n5).for
     grep -v "^"[eE]"" $(n5).for > $(n5).f

#
init:
     -rm cerr            # '-' error is ignored
```

The PCG-1 iterative solver is based on the preconditioned conjugate gradient method (see [2]). The solution (displacement) can be read in the same way as *rhs* vector (see above) from file FU.G32.

The source codes needed for forming PCG-1 code can be seen from corresponding make file

Make file for PCG-1

```
#  Makefile for solver - COPERNICUS version
#  make -f mksolver

ne = solver
n0 = itera
n1 = pcg
n2 = mxv3da
n3 = pcinit
n4 = pcsolv
n5 = readfv
n6 = gmopencp
#
opt = -O3

$(ne): init $(n0).o $(n1).o $(n2).o $(n3).o $(n4).o $(n5).o $(n6).o
     xlf $(opt) -o $(ne) $(n0).o $(n1).o $(n2).o $(n3).o $(n4).o \
      $(n5).o $(n6).o
```

```
# itera
$(n0).o: $(n0).f
     xlf -D $(opt) -c $(n0).f 2> cerr || pg cerr
$(n0).f: $(n0).for
     grep -v "^"[eE]"" $(n0).for > $(n0).f
# pcg
$(n1).o: $(n1).f
     xlf -D $(opt) -c $(n1).f 2> cerr || pg cerr
$(n1).f: $(n1).for
     grep -v "^"[eE]"" $(n1).for > $(n1).f
# mxv3da
$(n2).o: $(n2).f
     xlf -D $(opt) -c $(n2).f 2> cerr || pg cerr
$(n2).f: $(n2).for
     grep -v "^"[eE]"" $(n2).for > $(n2).f
# pcinit
$(n3).o: $(n3).f
     xlf -D $(opt) -c $(n3).f 2> cerr || pg cerr
$(n3).f: $(n3).for
     grep -v "^"[eE]"" $(n3).for > $(n3).f
# pcsolv
$(n4).o: $(n4).f
     xlf -D $(opt) -c $(n4).f 2> cerr || pg cerr
$(n4).f: $(n4).for
     grep -v "^"[eE]"" $(n4).for > $(n4).f
# readfv
$(n5).o: $(n5).f
     xlf -D $(opt) -c $(n5).f 2> cerr || pg cerr
$(n5).f: $(n5).for
     grep -v "^"[eE]"" $(n5).for > $(n5).f
# gmopencp
$(n6).o: $(n6).f
     xlf -D $(opt) -c $(n6).f 2> cerr || pg cerr
$(n6).f: $(n6).for
     grep -v "^"[eE]"" $(n6).for > $(n6).f
```

```
#
init:
     -rm cerr            # '-' error is ignored
```

The code COMPL enables the comparing of FKBC.G32, FRBC.G32, FU.G32 files with FKBC1.G32, FRBC1.G32, FU1.G32 files by the maximum norm ($\| u - u_1 \|_\infty$.) This comparison serves for the testing of new solvers and watching the errors.

The source codes needed for forming COMPFL code can be seen from corresponding make file.

Make file for COMPFL

```
#  Makefile for COMPFL for AIX
#  command:   make -f mkcompfl

ne = compfl
n0 = compflcp
n1 = gmopencp
#
opt = -O3

$(ne): init $(n0).o  $(n1).o
     xlf  $(opt) -o $(ne) $(n0).o  $(n1).o

# compfl
$(n0).o: $(n0).f
     xlf -D $(opt) -c $(n0).f 2> cerr || pg cerr
$(n0).f: $(n0).for
     grep -v "^"[eE]"" $(n0).for > $(n0).f
# gmopencp
$(n1).o: $(n1).f
     xlf -D $(opt) -c $(n1).f 2> cerr || pg cerr
$(n1).f: $(n1).for
     grep -v "^"[eE]"" $(n1).for > $(n1).f

#
init:
```

```
-rm cerr              # '-' error is ignored
```

2 Technical decsription

The system of linear equations is formed from FEM analysis of 3D linear elasticity problems. The used elements are tetrahedrons, elements with four nodal corners. The functions space contains piesewise linear functions, which are linear on each tetrahedron.

A specific feature of our code is the using of regular structured grids which, simply speaking, are the grids arising as a result of deformation of some regular rectangular grid (the pattern grid). The discretization is given by the decomposition of the domain into eight–node bricks and consequently into tetrahedral finite elements. Hence, the stiffness matrix (each brick is divided into 6 tetrahedra) has all nonzero entries within a 27–node (81 degrees of freedom (DOF)) regular stencil. Supposing the symmetry of the matrix, we can additionally store only the upper triangular part of the matrix. It can be done row–by–row by using regular 42 element stencil for the storage of the nonzero matrix entries.

The nodes of our grids can be therefore easily indexed by the triples (I,J,K),

$$1 \leq I \ \leq NX, \quad 1 \leq J \leq NY, \quad 1 \leq K \ \leq NZ$$

or enumerated by 1D numbers

$$INDN = I \ + (J-1) * NX + (K \ -1) * NX * NY.$$

For 3D elasticity problems, we have additionally 3 degrees of freedom (DOF) in each node. These DOF correspond to the displacements in the directions X, Y, Z . The overall number of degrees of freedom is

$$ND = 3 * NX * NY * NZ$$

and the degrees of freedom at the node with index INDN can be indexed by the numbers

$$
\begin{aligned}
INDX &= 3 * INDN - 2, \\
INDY &= 3 * INDN - 1, \\
INDZ &= 3 * INDN.
\end{aligned}
$$

The row numbers are given by the order of the records, column numbers have not to be stored due to the regular stencil. More exactly, the column numbers for the regular stencil and for the row index I will be the following:

If **I = INDX** for some DOF,
then the column numbers used in the 42 element stencil are subsequently:

$$\mathrm{J = I, I + 1, \ldots, I + 5}$$

and further

$$\mathrm{J = K, K + 1, \ldots, K + 8} \quad \text{for } \mathrm{K = I + K1, I + K2, I + K3, I + K4},$$

where

$$K1 = 3 * NX - 3, \quad K2 = 3 * NX * NY - 3 * NX - 3,$$

$$K3 = 3 * NX * NY - 3, \quad K4 = 3 * NX * NY + 3 * NX - 3.$$

If **I = INDY** for some DOF,
then the last entry in the stencil is not used and for the remaining positions, we have,

$$J = I, I + 1, ..., I + 4$$

and further

$$J = K, K+1, ..., K\ +8$$

for $\quad K\ = I-1+K1, \quad I-1+K2, \quad I-1+K3, \quad I\ -1+K4.$

If $\mathbf{I = INZ}$ for some DOF,
then the last two entries in the stencil are not used and

$$J = I, I\ +1, ..., I\ +3$$

and further

$$J = K, K\ +1, ..., K\ +8$$

for $\quad K = I-2+K1, \quad I-2+K2, \quad I-2+K3, \quad I-2+K4.$

In the FKBC.G32 file, one record involves three stencils for the rows which correspond to three degrees of freedom in one node. This enlarging of the record improves the speed of I/0 operations.

- **Local stiffness matrix**

  The global stiffness matrix is assembled from local stiffness matrices (12x12). The preparation of local matrix is done by LSTIFF code (see source code STIFF.FOR). The information about tetrahedral elements are read from thefile
  FEL.G32, the material (elasticity) matrices are read from file FMT.G32 (the number of the record is equal to the number of the material).

- **Right hand side vector**

  The vector of surface forces (file FR.G32) is completed by introducing weight of the material (code LOAD) automatically in this version. If you do not want to work with weight forces, you must input the specific weight (in TGRID) to be zero. The resulting vector is stored into a working file FRW.G32.

- **Modification of the global matrix and the right hand side vector**

The code GEORE serves for the modification of the matrix and the right hand side vector due to the prescribed displacement and the holes in the domain. Note that an additional vector U (stored in file FBC.G32) contains values of the prescribed displacements. In positions of U, corresponding to DOF where displacement is not prescribed, we can find the values 1E9.

The vector $A(42)$ is used for storing the matrix entries in one row according to the described regular stencil. The vector R serves for rhs vector of the system. The modification is then the following:

1. If the node with index INDN is in the void space then the k–th rows of matrix, $k = 3*INDN-2$, $3*INDN-1$, $3*INDN$, are changed by putting $A(1) = 1$, $A(l) = 0 \quad \text{for} \quad l = 2, ..., 42$. Also vector R is modificated so that $R(k) = 0$ for all these three k.
2. If the k-th row of matrix corresponds to DOF where displacement is not prescribed but l-th position in the stencil correspond to DOF with index j for which the displacement is prescribed. Then

   $R(j) = R(j) - A(l)U(j)$,

   $A(l) = 0$.

3. If the k-th row of matrix corresponds to DOF where displacement is prescribed. Then

   $R(j(l)) = R(j(l)) - A(l) * U(k) \;\; \text{for} \quad l = 2, \cdots, 42$,

   $R(k) = U(k)$,

   $A(1) = 1$,

   $A(l) = 0 \quad \text{for} \quad l = 2, ..., 42$

   where $j(l)$, $l = 2, ..., 42$, are column numbers for the given regular stencil. The modificated matrix is stored in the file FKBC.G32, the modificated right hand side vector is stored in the file FRBC.G32

3 User Guide

After the using of above mentioned "*make files*", we have codes TGRID, FEMSYS, PCG–1 a COMPFL prepared for work. The codes are started as follows:

`name of code` `< directory >`

The correponding files are read and written to directory `< directory >`. If the name of directory misses, the files are supposed to be in the same directory as started code.

a) **TGRID**

The code TGRID generates data files fully describing 3D FE linear elasticity problem. The all data can be read from keyboard or can be read from `DATA.REP` file. The DATA.REP file is generated when we read data from keyboard. It's enable to prepare the data in one place and to send corresponding DATA.REP file to all users which want to test solver on the same problem (they want to solve the same system of linear equations.) How to prepare input data for TGRID preprocessor can be seen from the following examples (input data for the described benchmarks).

a) **Square footing problem** (Fig. 2, see also [1]

Q0 (question 0):
INPUT DATA IN DATA.REP? Y/N −− >>

R0 (remark 0):
It is possible to prepare all input data in the file DATA.REP
and read data from this file. When we read data from keyboard,
they are also subsequently stored into DATA.REP file.
Answer will be "Y" ("y") for YES or "N" ("n") for NO.

A0 (answer 0):
N

Q1 TASK NAME –»

R1 name of task with maximal length 50 characters

A1 SQUARE FOOTING PROBLEM

Q2: DATE –»

R2: max. length 8 characters, recommended format dd.mm.yy.

A2: 15.06.95

Q3: NX=? NY=?, NZ=? (MAX. 64, INCLIN.GRID – ODD) –»

R3: We shall work with regular grids (deformed rectangular grids) which enable to identify any node with three serial numbers (indexes) i, j, k. The grid contains the nodes $(i, j, k) : 1 \leq i \leq NX$, $1 \leq j \leq NY$, $1 \leq k \leq NZ$ and we suppose that for rectangular grids the change of the index i represents the movement of the node in the X – direction, etc.
Note that this version of preprocessor has the limit 64 for all numbers NX, NY, NZ (this limit serves only for dimensioning of working arrays of TGRID). Thus we can generate FE systems with up to 786432 equations. Note also that the numbers NX, NY, NZ must be odd if we want to prepare inclined grids.

For our benchmark we use 40x40x40 grid (192 000 equations). Thus, we answer

A3: 40 40 40

Q4: DOMAIN <XMIN,XMAX> x <YMIN,YMAX> x
x < ZMIN, ZMAX >
$XMIN$ =? $XMAX$ =?

R4: Coordinates of vertices of the rectangular hexahedron domain are entered (in order X,Y,Z) .

A4: 0 50

Q5: YMIN = ? YMAX = ?

A5: 0 50

Q6: ZMIN = ? ZMAX = ?

A6: 0 40

Q7: NUMBER OF MATERIALS NMAT = ? –»

R7: The answer is the highest number of material which we will use.
For holes we use the material number 0 (void space).
In our problem the domain is materially homogeneous, so we answer

A7: 1

R: Now the requirements for parameters of materials follow
(material number from 1 up to NMAT) :

Q8: MATERIAL NUMBER 1
E(YOUNG)=? P(POISSON)=? GAMMA (SPEC. WEIGHT)=? –»

R8: We input Young. modulus E in MPa, Poisson's ratio ν
and specific weight in g/cm^3 for material number 1.
In the same manner we input the parameters for the subsequent materials.

A8: 14 0.3 2.5

$Q8a$
$A8a$
$Q8b$
$A8b$

questions for the subsequent materials are not necessary here.

Q9: SET FILE FX.G32 ? Y/N – »

R9: we can prepare new grid (answer "Y") or we can work with some previous grid (answer "N") and change only material parameters, distribution of materials etc. Characters different from "N" are understood equally as "Y".

A9: Y

R: if the answer is "N", then the next question is Q24

Q10: UNIFORM GRID ? Y/N – »

R10: We can use uniform grid ("Y"), but also there exist possibility ("N") to work with general rectangular grid. For this model problem we want

to refine the grid. If the answer would have been "Y" , the next question would be Q14.

A10: N

R if the answer is "N" (or any character different from "Y"), we can describe a general rectangular grid. This grid will be described by the sets of coordinates in the directions X,Y,Z. Moreover, we do not have to input all these coordinates. Firstly, the first and last coordinates are implicitly taken from the previous limits, i.e. $X(1) = XMIN$ and $X(NX) = XMAX$, etc. Secondly, if we skip some coordinates, then they are finally computed by interpolation, i.e. . if $X(i+1), ..., X(j-1)$ are skipped then $X(k) = X(i) + (k-1)*$ $(X(j) - X(i))/(j-i)$ for $k = i+1, ..., j-1$.

Thus for all directions we input the pairs of the serial number of coordinate and the value of coordinate. If we made a mistake, we can input the same serial number and the value $1E9(10^9)$ for clearing up the wrong record.

Q11 DIRECTION X
I=? – »

R11: the number of the coordinate. By inputing 0, we continue for the other direction.

A11: 20

Q11a: X=? –»

A11a: 10.0

Q11b: I=? –»

A11b: 30

Q11c: X=? –»

A11c: 25.0

Q11d: I=? –»

A11d: 0

Q12: DIRECTION Y
J=? –»

A12: 20

Q12a: Y=? –»

A12a: 10.0

Q12b: J=? –»

A12b: 30

Q12c: Y=? –»

A12c: 25.0

Q12d: J=? –»

A12d: 0

Q13: DIRECTION Z
K=? –»

A13: 20

Q13a: Z=? –»

A13a: 10.0

Q13b: K=? –»

A13b: 30

Q13c: Z=? –»

A13c: 20.0

Q13d: K=? –»

A13d: 0

R: Now the rectangular grid is described. The projection on XZ plane can be seen from Fig.5. The projection on YZ plane looks equally.

Q14: RECTANGULAR GRID ? Y/N –»

R14: Rectangular grid can be slightly modified to nonrectangular grid (inclined grid). It is only for the possibility to test solvers on nonrectangular grids. The inclination is fixed, the user can't change it. All answers except "Y" (or "y") make the inclination.

A14: Y

R: Now we can proceed to the description of the distribution of materials in the considered domain. In the beginning the whole domain is supposed as materially homogeneous, with material 1 in all elements. This initial material distribution can be changed by input of another material number in "hexahedral" subdomains which are blocks of eight-node bricks. It will be explained in more detail for the second benchmark because it is not necessary for the present problem.

Q15: NEW MATERIAL DISTRIBUTION ? Y/N –»

R15: Any answer except "N" cause reading data for change of material distribution.

A15: N

R: Generally Q15 is repeated until the whole distribution of material is defined.

Q16: PRESSURE/FORCE PZ=? –»

R16: Input of the value of uniform pressure on the "rectangle" area on the top side (in MPa).

A16: 2.4

Q17: 1 <= I <= ? and 1 < = J < = ?

R17: Here the upper limits Imax, Jmax are read which define a "rectangle" area on the top side of the domain. This area contains nodes with indexes $(i, j, 1) : 1 \leq i \leq Imax$, $1 \leq j \leq Jmax$.

A17: 10 10

Now the description of the problem is finished and the text "TGRID FINISHED" appears on the screen. Note that zero normal displacement and zero tangential stresses are assumed on all sides different from the top side of the domain.

Stope stability problem (Fig. 3, see also [1]

Now, we shall show the generation of the grid more briefly than in the previous problem, only the distribution of the material will be shown in more detail.

A0: N
A1: STOPE STABILITY PROBLEM
A2: 15.06.95
R: specify basic data
A3: 40 40 40
A4: 0 150
A5: 0 150
A6: 0 150
R: specify material parameters
A7: 3
A8: 8000 0.2 2.5
A8a: 800 0.3 1.3
A8b: 200 0.4 2.2
R: specify grid coordinates
A9: Y
A10: N
A11: 5
A11a: 37.0
A11bc: 10
A11c: 57.0
A11d: 30

A11e: 97.0

A11f: 36

A11g: 121.0

A11h: 0

A12: 3

A12a: 20.0

A12b: 6

A12c: 38.0

A12d: 8

A12e: 44.0

A12f: 28

A12g: 74.0

A12h: 32

A12i: 86.0

A12j: 36

A12k: 110

A12l: 0

A13: 5

A13a: 40.0

A13b: 9

A13c: 64.0

A13d: 13

A13e: 76.0

A13f: 33

A13g: 106.0

A13h: 35

A13i: 112.0

A13j: 38

A13k: 130.0

A13l: 0

A14: Y

Q15: NEW MATERIAL DISTRIBUTION ? Y/N –»

A15: Y

R: Now questions and answers for defining the material distribution will follow. The initial material distribution (material number 1 in all elements) or the previous material distribution is changed by defining new material number in the subdomain containing nodes $(i, j, k) : I1 \leq i \leq I2$, $J1 \leq j \leq J2$, $K1 \leq k \leq K2$.
This subdomain composed from eight-node bricks defined by the grid, can be called generalized hexahedron or block of eight-node bricks .

Q15a: THE NUMBER OF MATERIAL ? –»

R15a: For this model problem we shall use the following materials

0 – corridors (void space)
1 – origin rock
2 – coal
3 – broken rock (goaf material)

A15a: 2

Q15b: DIRECTION X, FROM NODE I1 TO I2? –»

R15b: We shall read indexes of nodes which form generalized hexahedron
< I1,I2 > x < J1,J2 > x < K1,K2 >

A15b: 1 40

Q15c: DIRECTION Y, FROM NODE J1 TO J2? –»

A15c: 1 40

Q15d: DIRECTION Z, FROM NODE K1 TO K2 ? –»

A15d: 27 29

R: The same questions and answers follow for further changes of material distribution.

A15e: Y

A15f: 0

A15g: 19 21

A15h: 1 40

A15i: 27 29

A15j: Y

A15k: 0

A15l: 1 19

A15m: 16 18

A15n: 27 29

A15o: Y

A15p: 3

A15q: 1 19

A15r: 26 40

A15s: 17 20

A15t: Y

A15u: 3

A15v: 1 20

A15w: 24 40

A15x: 19 22

A15y: Y

A15z: 3

A15a1: 1 20

A15b1: 22 40

A15c1: 21 224

A15d1: Y

A15e1: 3

A15f1: 1 20

A15g1: 20 40

A15h1: 23 26

A15i1: Y

A15j1: 3

A15k1: 1 20

A15l1: 18 40

A15m1:25 30

A15n1: N

Now answers for introducing pressure on the top side of the domain follow. We suppose uniform pressure on the whole top side to simulate the depth of investigated domain under surface (the weight of overburden).

A16: 12.5

A17: 40 40

Remark:

Who knows how the DATA.REP file is formed, can form new test problem by suitable modification od DATA.REP file. (DATA.REP file is FORMATTED ASCII file).

b) **FEMSYS**

Code FEMSYS generates files containing information about the stiffness matrix and the load vector problem. After starting the code doesn't require any data from keyboard.

c) **PCG–1**

Code PCG–1 solves system of linear equations by the preconditioned conjugate gradient method. From keyboard are read accuracy of solution of system of equations, type of preconditioning and max. number of iterations for the solving of equations without reaching required accuracy). The following choices of the preconditioning are available:

ipc = 0	=>	no preconditioning
ipc = 1	=>	diagonal preconditioning
ipc = 2	=>	SSOR preconditioning
ipc = 3	=>	DD–IF preconditioning
ipc = 11	=>	automatic choice: switch DD–IF to SSOR, in the case of instability of DD–IF incomplete factorization

d) **COMPFL**

This code enables the comparison of FKBC.G32 (stiffness matrix), FRBC.G32 (rhs vector), FV.G32 (solution – displacement vector) files with FKBC1.G32, FRBC1.G32, FV1.G32 by the maximum norm ($\| u - u_1 \|_\infty$). This comparison serves for testing new solvers and watching the errors.

4 Tests, benchmarks

The solvers can be tests on modelled benchmarks. The presented benchmarks are motivated by problems of soil and rock mechanics and lay between pure academic model problems [1] and real-life problems of geomechanics [2].

- **Square footing problem** (Fig. 2)
  is the problem of soil mechanics. The square area (10x10 m) on the top side of the considered domain is loaded by uniform load (2.4 MPa).

  The whole domain is parallelepiped of the size 100x100x40 m. Due to the symmetry of the problem, we restrict ourselves to the domain 50x50x40 m with the load situated on the square area (5x5) in the corner.

  Body forces are formed by the weight of the material.

Table 1: Solving of the square footing problem by PCG1

preconditioning	number of iterations	time on IBM RS 6000/320
Diagonal	409	305
SSOR	84	169
DD – IF	49	95

Boundary conditions represent zero normal displacements and zero tangential stresses on all sides except the top one where the pressure is imposed.

The grid (40x40x40 nodes) is rectangular with a grid refinement under the footing.
The material is homogeneous and isotropic. Preparing of input data was demonstrated in the previous section.

- **Stope stability problem** (Fig.3)
  is the problem concerning mining in the coal seam. The whole domain is parallelepiped of the size 300x150x150 m which is situated in the depth of 500 m under surface. The depth is simulated by loading 12.5 MPa on the whole top side of the domain. Due to the symmetry, we restrict ourselves to the domain 150 x 150 x 150 m. Body forces are formed by the weight of the material.

  We suppose zero normal displacements and zero tangential stresses on all sides except the top one where the pressure is imposed.

  The grid (40x40x40 nodes) is rectangular with a grid refinement in the area of mining. The domain consists of three materials: coal, surrounding rock and broken material (goaf) behind mining. Roads are modelled by "zero" material. All materials are isotropic.

The benchmark problems were solved by PCG1 code with using various type of preconditioning. The results can be seen from Tables 1 to 4. In Tables 3 and 4, RES = $\sqrt{< r, r >}$ denotes l_2 norm of the residual, CONV.F. and A.CONV.F are local and global convergence factors, PC.RES = $\sqrt{< Cr, r >}$ where C is the preconditioner and RES/RHS is the norm of the residual

Table 2: Solving of the stope stability problem by PCG1

preconditioning	number of iterations	time on IBM RS 6000/320
Diagonal	358	268
SSOR	68	140
DD – IF	63	122

divided by the norm of the rhs vector. The presented results can serve for comparison with different solvers. Concerning computational time, we must note that the PCG1 program can be substantially speeded up by merging some loops, use more compact storage of the stiffness matrix and increase the record in the matrix storage file.

Table 3:

```
NAME= SQUARE FOOTING PROBLEM
DATE= 15.06.95

DIAGONAL SCALING
 IT   :    RES    :    CONV.F.   :  A.CONV.F.  :   PC.RES :     RES/RHS

    40  .54143E+02  .10194E+01   .10312E+01   .61704E+01  .32407E+01
    80  .11960E+02  .96756E+00   .99647E+00   .13934E+01  .71584E+00
   120  .53358E+01  .10088E+01   .99096E+00   .62172E+00  .31936E+00
   160  .31861E+01  .98531E+00   .99001E+00   .35823E+00  .19070E+00
   200  .18351E+01  .97143E+00   .98927E+00   .20562E+00  .10984E+00
   240  .76626E+00  .96316E+00   .98745E+00   .85139E-01  .45863E-01
   280  .37401E+00  .97714E+00   .98670E+00   .42044E-01  .22386E-01
   320  .17036E+00  .97853E+00   .98593E+00   .18570E-01  .10196E-01
   360  .79101E-01  .96511E+00   .98538E+00   .87592E-02  .47345E-02
   400  .21239E-01  .98088E+00   .98359E+00   .23058E-02  .12712E-02
   409  .16197E-01  .96241E+00   .98330E+00   .17499E-02  .96944E-03
COMPUTING TIME= 18326.689

SSOR PRECONDITIONER, X=.67*D
```

IT :	RES :	CONV.F. :	A.CONV.F. :	PC.RES :	RES/RHS
20	.86684E+01	.90467E+00	.95682E+00	.15436E+01	.51883E+00
40	.23045E+01	.91856E+00	.94630E+00	.38278E+00	.13793E+00
60	.35830E+00	.91669E+00	.93443E+00	.57490E-01	.21445E-01
80	.27416E-01	.82815E+00	.92036E+00	.43384E-02	.16410E-02
84	.14484E-01	.85255E+00	.91701E+00	.22574E-02	.86693E-03

COMPUTING TIME= 10157.390

DD-IF PRECONDITIONING

IT :	RES :	CONV.F. :	A.CONV.F. :	PC.RES :	RES/RHS
20	.39690E+01	.87907E+00	.92376E+00	.15190E+01	.23756E+00
40	.15398E+00	.77803E+00	.88614E+00	.46988E-01	.92162E-02
49	.13952E-01	.78359E+00	.86270E+00	.45352E-02	.83510E-03

COMPUTING TIME= 5680.160

Table 4:

NAME= STOPE STABILITY PROBLEM
DATE= 15.06.95

DIAGONAL SCALING

IT :	RES :	CONV.F. :	A.CONV.F. :	PC.RES :	RES/RHS
40	.14418E+05	.99289E+00	.10144E+01	.31401E+02	.13862E+01
80	.11635E+05	.99014E+00	.10045E+01	.27758E+02	11186E+01
120	.30125E+04	.96854E+00	.99177E+00	.70411E+01	.28962E+00
160	.11547E+04	.97481E+00	.98788E+00	.28133E+01	.11102E+00
200	.57208E+03	.99398E+00	.98682E+00	.14198E+01	.55000E-01
240	.32132E+03	.97649E+00	.98663E+00	.80783E+00	.30892E-01
280	.98762E+02	.97939E+00	.98437E+00	.25059E+00	.94951E-02
320	.40822E+02	.96563E+00	.98359E+00	.10399E+00	.39247E-02
358	.99269E+01	.95500E+00	.98144E+00	.25662E-01	.95439E-03

COMPUTING TIME= 16083.050

SSOR PRECONDITIONER, X=.67*D

```
 IT :    RES    :    CONV.F.   :  A.CONV.F.   :   PC.RES   :    RES/RHS

   20  .47280E+04  .83080E+00   .94055E+00   .16753E+02  .45455E+00
   40  .52351E+03  .95096E+00   .91791E+00   .18333E+01  .50331E-01
   60  .37057E+02  .89307E+00   .90371E+00   .12439E+00  .35627E-02
   68  .92031E+01  .83699E+00   .89600E+00   .29990E-01  .88480E-03
COMPUTING TIME= 8371.360

DD-IF PRECONDITIONING
 IT :    RES    :    CONV.F.  :  A.CONV.F.   :   PC.RES  :    RES/RHS

   20  .35066E+04  .88310E+00   .94395E+00   .29522E+02  .33713E+00
   40  .51041E+03  .85870E+00   .92587E+00   .37813E+01  .49072E-01
   60  .16497E+02  .85763E+00   .89714E+00   .11154E+00  .15860E-02
   63  .10371E+02  .85667E+00   .89517E+00   .72764E-01  .99711E-03
COMPUTING TIME= 7348.740
```

5 Availability of codes

The source of the described codes can be provided to the COPERNICUS CP 940820 project partners for enabling them to work with the benchmarks and testing their own solvers. The codes for generating the FE system and iterative solver belongs to the IGN–GEM software which is a property of the Institute of Geonics, Ostrava.

Source codes are written in FORTRAN 77 and can be distributed by e–mail. In similar way can be distributed DATA.REP files for above mentioned benchmarks.

Contact address: KOHUT@UGN.CAS.CZ

References

[1] R. Blaheta, R. Kohut, Benchmarks and programs for testing iterative solvers for 3D problems of geomechanics, Report 9502, Dept. Appl. Math., Institute of Geonics, Ostrava, August 1995.

[2] R. Blaheta and R. Kohut, PCG–1 iterative solver, Report 9501, Dept. Appl. Math., Institute of Geonics, Ostrava, June 1995.

[3] A. Hájek and Z. Rakowski, Mathematical models for development of mining technology. Basic description, COPERNICUS CP 940820 report, D. Rozinka, August 1995.

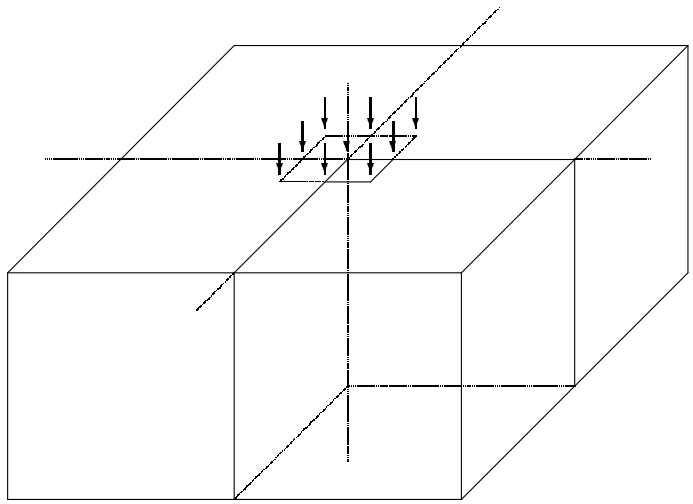

Figure 2: Square footing problem

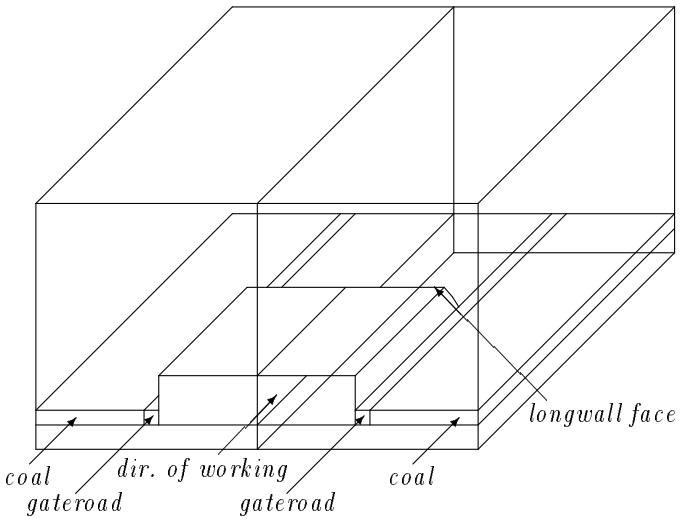

Figure 3: Stope stability problem