FFIBM/766/130

Approved
Kjeller October 19, 2000

Bjarne Haugstad
Director of Research

MATLAB TOOLBOX FOR PROCESSING OF GREAC CELL DATA

RUSÅS Per–Olav, TELAND Jan Arild

FFI/RAPPORT–2000/02032

FORSVARETS FORSKNINGSINSTITUTT
Norwegian Defence Research Establishment
Postboks 25, 2027 Kjeller, Norge

FORSVARETS FORSKNINGSINSTITUTT (FFI)
Norwegian Defence Research Establishment
P O BOX 25
NO–2027 KJELLER, NORWAY

REPORT DOCUMENTATION PAGE

1) PUBL/REPORT NUMBER	2) SECURITY CLASSIFICATION	3) NUMBER OF PAGES
FFI/RAPPORT–2000/02032	UNCLASSIFIED	31
1a) PROJECT REFERENCE	**2a) DECLASSIFICATION/DOWNGRADING SCHEDULE**	
FFIBM/766/130		

4) TITLE

MATLAB TOOLBOX FOR PROCESSING OF GREAC CELL DATA

(MATLAB TOOLBOX FOR TOLKNING AV GREAC CELL DATA)

5) NAMES OF AUTHOR(S) IN FULL (surname first)

RUSÅS Per–Olav, TELAND Jan Arild

6) DISTRIBUTION STATEMENT

Approved for public release. Distribution unlimited
(Offentlig tilgjengelig)

7) INDEXING TERMS

IN ENGLISH:	IN NORWEGIAN:
a) Concrete testing	a) Betongtesting
b) GREAC cell	b) GREAC celle
c) Matlab toolbox	c) Matlab toolbox
d) Finite element code	d) Endelig elementkode
e) Material data	e) Materialdata

THESAURUS REFERENCE:

8) ABSTRACT

This report provides the documention for a Matlab toolbox that has been developed at FFI. The purpose of the toolbox is to process and analyse raw data from a GREAC cell test, in order to extract material models and parameters for the concrete being tested. It is optional whether to do the calculations with the thick cylinder theory approximation, or find a finite element solution. A brief introduction to the theory of finite element codes is also given.

9) DATE	AUTHORIZED BY This page only	POSITION
October 19, 2000	Bjarne Haugstad	Director of Research

ISBN-82-464-0464-4

CONTENTS

1 INTRODUCTION

FFI Project 766 has as a main goal to establish reliable methods for describing penetration into concrete and rock, with emphasis on studies and use of hydrocodes. The obtained results and skills will be used by FFI's vulnerability studies group, contributing to future studies of military installations.

In hydrocode simulations, as well as with other prediction tools, it is vital to use physically realistic mathematical descriptions of the materials involved. Values for the different material parameters must then be found by performing suitable experiments.

One such experiment is the GREAC (Gauged Reactive Confinement) cell test. This is a tri–axial compression test, in which a cylindrical test specimen confined by a steel (or similar) jacket is compressed in a hydraulic testing machine. Depending on the sample size, the machine and the properties of the confining jacket, hydrostatic pressures in the GPa range can be achieved. The GREAC cell setup at FFI is based on the methods and equipment developed at Imperial College in London .

The material parameters are not measured directly in this test, and hence the output data must be processed in order to produce the relevant information. In this report, a computer program (GREAC cell toolbox) which performs this task will be described. The program requires a version of Matlab (1) installed on the system.

Basically, measurements of external strains on the confining cylinder, together with axial compression and force are used for calculating the stresses and strains in the material sample. Based on this, an equation of state and a pressure dependent yield (failure) surface can be calculated. These curves constitute the basic data sets in common material models.

When processing the raw data from the GREAC cell, the behaviour of the confining cylinder must be modelled. The toolbox includes the option of choosing between two different methods for this task, either the standard analytical thick cylinder theory which is a planar description with circular symmetry, or a 3D finite element calculation which also includes the ”barreling” which occurs when the material sample becomes shorter than the confining jacket.

For completeness, a brief description of finite element theory (2) has been included in Appendix A. Further detail about the GREAC test method, interpretation of the results and their application in material models can be found in the references (3)–(5).

2 SHORT OVERVIEW OF THE GREAC CELL TOOLBOX

The GREAC cell toolbox is divided into two separate programs. The first part computes stresses and strains inside a steel jacket that is loaded by a given inner pressure or shear stress. The second part is a menu based program which processes raw experimental data and thereby determines the material properties of the (concrete) sample.

- The first part requires no input from experimental data. The calculation is purely numerical/analytical and the user only needs to specify values for the initial and boundary conditions of the problem in the file `setpar.m`. The results for the various elements are calculated and stored in the variable `eleme8tres` by executing the comand `eleme8tres = greacfem(x,y,z),` where the input parameters are explained in Chapter 4.

- The second part requires an input file containing the measured values for results of force, strains and displacements. (It is here assumed that some basic data processing has been performed to convert the logged voltages to physical quantities). The format and content of this file is described in Chapters 6 and 7.

- A file describing the geometry and material of the confining jacket is also needed. How to create such a file is described in detail in Chapters 6 and 7.

- The program is executed by the command `greacme8u`, which will cause a menu to appear. From the menu one can select which data files to process, as well as the method for processing (thick cylinder theory or finite element calculations). After having finished the processing, the results are plotted automatically and (if chosen) estimates for the elastic constants are given. On exiting, the option of saving the results in a file is given. Complete details and examples are given in Chapter 5.

3 DOCUMENTATION OF THE GREAC CELL TOOLBOX

The Matlab functions for analysing GREAC cell results have been collected in a socalled “toolbox”, which is a typical way of gathering useful Matlab functions. This GREAC cell toolbox is available from FFI.

The toolbox is platform independent and should run without modifications under both Unix and Windows. However, it requires that Matlab 5.0 or higher is installed on your computer.

3.1 Installation

To install the toolbox, you run Matlab and change to the directory where the toolbox is located. On a Unix workstation connected to the internal computer network at FFI, this is done in the following way:

cd /net/pilot/ar14/grupper/p766/ml/greac

The installation procedure is now found by typing:

```
help Readme
```

3.2 Directory structure of the Matlab functions

The Matlab functions have been organised in the following directory structure:

- greac – Directory with the main user functions.
- greac/femsubfun – Directory with help functions for the FEM–calculations of the GREAC cell jacket.
- greac/GREACmenusubfun – Directory with help functions for the menu based processing of the GREAC cell data.
- greac/data – Directory containing data for steel cylinders along with various other experimental data.

A summary of the main user functions are found by typing `help greac`. The following text will then appear:

```
GREAC cell toolbox - May 11, 1999
Per-Olav Rusås

Debugged by Jan Arild Teland, Marius Halsør and Lars Kvifte - February 2000


GREAC cell calculations and interpretation of experimental data from GREAC
cell tests

FEM calculations of the GREAC cell
  makesteelcylinderdata    - Script that produces data for GREAC-cell processing
  greacfem                 - Calculates stresses and strains in the steel jacket
  setpar                   - Defines the parameters describing the jacket and the
                             calculation method
  resinnside               - Calculates strains and displacements on the inside
                             of the steel jacket
  plotgrid                 - Creates a deformation plot of the steel jacket
  plotgridres              - Creates a contour plot of stresses and strains inside
                             the steel jacket
  plotstrainoutside        - Creates a plot of the stresses on the outside of the
                             jacket

Help functions for the FEM-calculations are located in the directory femsubfun/
```

```
Thick cylinder theory
  thickcyl                - Uses thick cylinder theory to calculate stresses and
                            strains in the steel jacket

Menu driven processing of GREAC cell data.
  greacmenu               - Activates the menu for processing GREAC cell data

Help functions for greacmenu is located in the directory GREACmenusubfun/
```

A summary of the help functions for the FEM–code is found by typing:

`help femsubfun:`

```
% Help functions for FEM-calculations of the GREAC cell test
%
% Functions that are used in FEM-calculations of the GREAC cell test
%  Nshape        - Shape functions (interpolation functions) for 2D elements (N)
%  Emat          - Elasticity matrix for 2D axial symmetry (E)
%  Bstraindisp   - Strain-displacement matrix (B)
%  elstiff       - Element stiffness matrix
%  assemble      - Assembles, i.e. creates a structural stiffness matrix
%                      from the element stiffness matrices (K)
%  loadvector    - The contribution of applied stresses to the load vector (R)
%  displacecond  - Changes K and R to satisfy the displacement boundary conditions
%  calcstress    - Calculates stress from strains
%
% Help functions for post processing
%  findelement   - Finds an element containing a given coordinate
%  plotstrain    - Plots strains and stresses as function of the radial coordinate
```

Again a description of various functions can be found by using the Matlab command `help`.

4 COMPUTING STRESSES AND STRAINS IN THE GREAC-CELL JACKET

Here we give an example of how to use the toolbox to calculate stresses and strains:

1. Create a symbolic link to the directory where the toolbox is located. On a Unix workstation at FFI this is achieved in the following way:
   ```
   cd ~
   ln -s /net/pilot/user/p766 p766
   ```
2. Run Matlab (To ensure that the paths in startup.m are enabled, you must run it from your home directory.)

3. Create your own directory for storing results and change to this directory:
   ```
   >> mkdir mygreac
   >> cd mygreac
   ```

4. Copy `setpar.m` from the p766-directory (or wherever the toolbox is located):
   ```
   >> !cp ~/p766/ml/greac/setpar.m .
   ```

5. Use a text editor to edit `setpar.m`, and define the variables you desire. An example is given below:

```
function [physpar,numpar] = setpar
% SETPAR defines the parameters describing the steel cylinder and calculation method
%   [fyspar,numpar] = setpar gives value to the parameters
%   Make a copy of this file and place it in your directory.  Only make
%   changes to the copy.

% ************************************
% Parameters for materials and geometry
% ************************************

% geometry
% Large cylinder diameter 76.2mm
physpar.inner_r = 1;        % 38.1 mm inner radius
physpar.outer_r = 50.8/38.1; % 50.8 mm outer radius
physpar.h = 75/38.1;        % 75.0 mm half of the cylinder height

% symmetric about y=0
physpar.sym = 1;           % 1 (symmetric) or 0 (not symmetric)

% axis symmetry (plane strain if no axis symmetry)
physpar.axissym = 1;

% material properties
physpar.E = 1;             %206 GPa
physpar.nu = 0.288;

% ******************
% boundary conditions
% ******************
%
% inner pressure
physpar.p0 = 1e-3;         % 0.001*E

% inner shear stress
```

```
physpar.tau0 = 0;          % positive in pos. y-direction

% works up to the following (y-value)
physpar.yp0 = physpar.h*0.9;


% constrain u at outer radius (equal to zero for normal GREAC cell calculations)
physpar.constrainu = 0;     % 0 eller 1

% factor for plotting of displacements
numpar.plotfactor = 10;
```

From this version of `setpar.m`, we see that stresses are calculated in units of E (Young's modulus) when the cylinder is loaded by an internal pressure of magnitude 0.001**E*. The internal pressure is applied between the symmetry plane y=0 (whose normal vector points in the axial direction) and 0.9 times the cylinder height. The inner radius is defined as unit length.

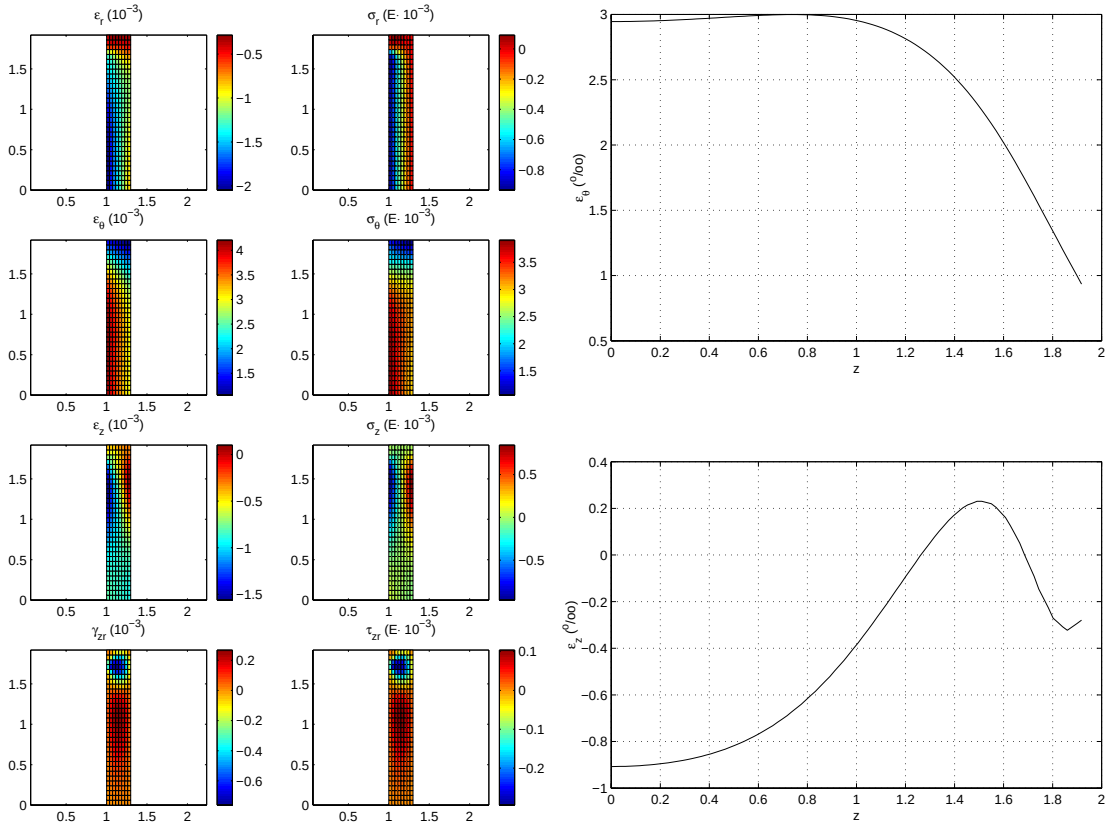

Figure 4.1 Examples of plots created by `greacfem`.

6. Calculations are run and results are stored in the structured variable `elementres`:

```
>> elementres = greacfem(9,8,32)
```

(See `help greacfem` for an explanation of the parameters associated with `greacfem`). Figure 3.1 shows the results which now appear in new windows that pop up on the screen.

7. Display the contents of `elementres`:

```
>> elementres

elementres =

        element: {1x256 cell}
          nodes: [1x1 struct]
        physpar: [1x1 struct]
         numpar: [1x1 struct]
              K: [2227x2227 sparse]
              D: [2210x1 double]
             Em: [4x4 double]
    cyloutside: [1x1 struct]
     cylinside: [1x1 struct]
```

8. Description of `elementres.element`.
   We can now extract results for each element. As an example, we display the results for element nr. 12

```
>> element12=elementres.element{12}

element12 =

             x: [9x1 double]
             y: [9x1 double]
          Geps: [4x9 double]
        Gsigma: [4x9 double]
            Gu: [1x9 double]
            Gv: [1x9 double]
            Gx: [1x9 double]
            Gy: [1x9 double]
          psig: [3x1 double]
            pn: [3x3 double]
   Gsigma_mises: [1x9 double]
           eps: [4x9 double]
         sigma: [4x9 double]
```

```
    u: [1x9 double]
    v: [1x9 double]
```

Table 4.1 Description of the variables inside the structured variable `elementres.element{n}`*:*

Variable	**Description**
`x: [9x1 double]`	`x(j)` is radial coordinate for node `j` of the element.
`y: [9x1 double]`	`y(j)` is axial coordinate for node `j` of the element.
`Gx: [1x9 double]`	`Gx(j)` is radial coordinate for the quadrature point of element `j`.
`Gy: [1x9 double]`	`Gy(j)` is axial coordinate for the quadrature point of element `j`.
`u: [1x9 double]`	`u(j)` is radial coordinate for node `j` of the element.
`v: [1x9 double]`	`v(j)` is axial coordinate for node `j` of the element.
`eps: [4x9 double]`	Matrix containing the strains of the element nodes. Rows 1–4 contain radial, tangential, axial and shear strain. Column `j` gives the strain of node `j`.
`sigma: [4x9 double]`	Matrix with the stresses of the element nodes. Rows 1–4 contain radial, tangential, axial and shear stress. Column `j` gives the stress of node `j`.
`Geps: [4x9 double]`	Similar to `eps` defined above, but for the quadrature points of the element.
`Gsigma: [4x9 double]`	Similar to `sigma` defined above, but for the quadrature points of the element.
`Gu: [1x9 double]`	Similar to `u` defined above, but for the quadrature points of the element.
`Gv: [1x9 double]`	Similar to `v` defined above, but for the quadrature points of the element.
`psig: [3x1 double]`	Principal stresses in the middle point of the element.
`pn: [3x1 double]`	The principal stress directions are the columns of `pn`.
`Gsigma_mises: [1x9 double]`	`Gsigma_mises(j)` is the von Mises stress in quadrature point `j`.

The values of a variable in `elementres.element{n}` can be retrieved in the following way:

```
>> format short e

>> sigma=elementres.element{15}.sigma

sigma =

  Columns 1 through 6
```

```
-1.8174e-04  -8.6630e-05  -8.6703e-05  -1.8184e-04  -1.3485e-04  -8.6568e-05
 3.1235e-03   3.0218e-03   3.0247e-03   3.1262e-03   3.0709e-03   3.0231e-03
-2.8007e-05  -4.4693e-05  -4.1655e-05  -2.5977e-05  -3.6875e-05  -4.3186e-05
 2.6620e-06   1.5431e-06   3.0923e-06   5.3565e-06   2.1097e-06   2.2483e-06

Columns 7 through 9

-1.3492e-04  -1.8172e-04  -1.3480e-04
 3.0737e-03   3.1247e-03   3.0721e-03
-3.4323e-05  -2.7020e-05  -3.5620e-05
 4.2368e-06   3.9310e-06   3.0995e-06
```

9. Other results are also stored in elementres:

```
>> elementres.cylinside

ans =

           y: [1x65 double]
       eps_r: [1x65 double]
    eps_teta: [1x65 double]
       eps_y: [1x65 double]
           u: [1x65 double]
           v: [1x65 double]

>> elementres.cyloutside

ans =

          y: [1x100 double]
    epsteta: [1x100 double]
       epsy: [1x100 double]
```

   These variables contain the strains on the inside and outside boundary of the cylinder for the points with axial coordinate *y*.

10. Display the results graphically:

    The deformed cylinder with displacements magnified 10 times (Figure 3.2):

```
>> plotgrid(elementres,1,10)
```

    The strain on the outside boundary of the cylinder:

```
>> plotstrainoutside(elementres,100)
```

This gives the plot to the right in Figure 3.2.

Create plots that show stresses and strains, like in Figure 3.3:

```
>> plotgridres(elementres,1,elementres.physpar.yp0)
```

11. Store `elementres` as the file '`femresults.mat`'

```
>> save femresults.mat elementres
```

The results can later be retrieved with the following command:

```
>> load femresults.mat
```

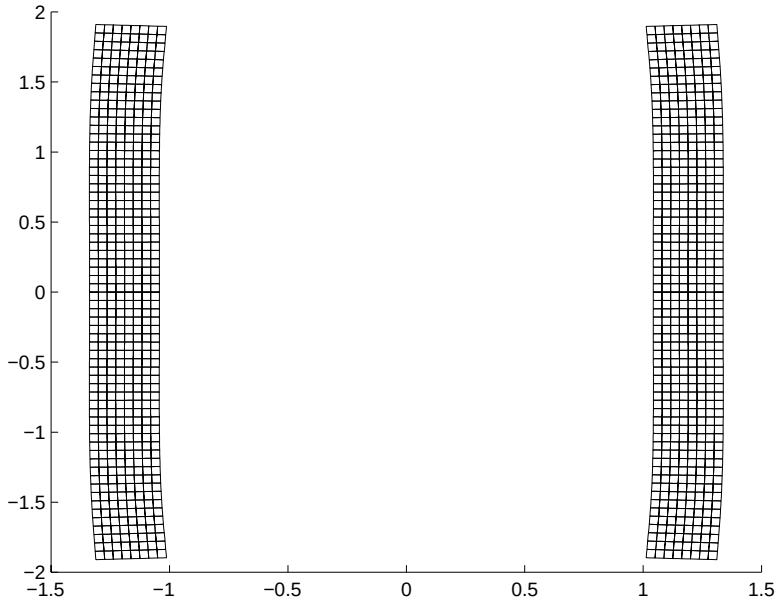

Figure 4.2 Deformed cylinder with displacements magnified by a factor 10. Created by `plotgrid`.

5 MENU BASED METHOD FOR PROCESSING OF GREAC CELL DATA

1. Follow points 1–3 of the example given in Chapter 3, if this has not already been done.

2. Copy the files containing GREAC cell data into your own directory:
   ```
   >> cd ~/mygreac
   >> !cp ~/p766/ml/greac/data/* .
   ```

3. Now the program which processes the GREAC cell data should be run:

   ```
   >> greacmenu
   ```

   Figure 4.1 shows the the menu that will now be displayed. The program searches the directory from which it was started and displays all the files (with the relevant extension) in the upper left corner. By default all these files will be analysed, unless the "Remove file" option is used to deselect it. (This option does not delete the file, but only removes it from the list of files to be analysed.) Use the "Refresh" option to search the directory for new files.

4. After the relevant options have been selected, you must push the button marked "Process data" in the lower left corner. This will run the calculations and display the relevant results extracted from the data. As an exercise, you should try processing with the same options as selected in Figure 5.2. The final results should then be as shown in Figure 5.3.

   Note that the unit for stress in this example is *Pa* since the Young's modulues of steel is given as '206e11', i.e with unit *Pa.* The units of variables with units of stress (f.ex. pressure, bulk modulus, shear modulus) are thus indirectly given by the unit which is chosen for the Young's modulus of the steel (By putting a '1' in the menu, one can also use the Young's modulus of the steel as a unit, which is sometimes convenient.)

5. The results are stored in the global variables `GREACthickcylres` and `GREACFEMres.` In our example all the results are stored in `GREACthickcylres,` since we only selected 'Thick Cylinder theory'. By selecting 'FEM–calculations', we can make the program store results in `GREACFEMres` as well.

   ```
   >> GREACthickcylres

   GREACthickcylres =

       [1x1 struct]    [1x1 struct]    [1x1 struct]    [1x1 struct]
   ```

   The results for the raw data stored in `R´k250a05.dat´:`
   ```
   >> GREACthickcylres{2}

   ans =
   ```

```
    concrete: [1x1 struct]
       steel: [1x1 struct]
```

To only retrieve results for the concrete:

```
>> GREACthickcylres{2}.concrete

ans =

              p: [2521x1 double]
          eps_v: [2521x1 double]
          eps_r: [2521x1 double]
          eps_z: [2521x1 double]
        epsdiff: [2521x1 double]
        sigma_z: [2521x1 double]
        sigma_r: [2521x1 double]
      sigmadiff: [2521x1 double]
              S: [2521x1 double]
              G: [2.3418e+09 2.5031e+09]
              K: [3.3573e+09 3.3922e+09]
              E: [5.7001e+09 6.0269e+09]
             nu: [0.2170 0.2039]
     distenergy: [1.7449e+05 4.7369e+05]
```

Most of these variables should be self–explanatory, perhaps with an exception for these ones:

`S` The total shear force from the steel on the concrete for the upper half of the concrete.

`distenergy` The distortion energy that is lost (converted to irreversible forms of energy) during the loading and unloading cycles.

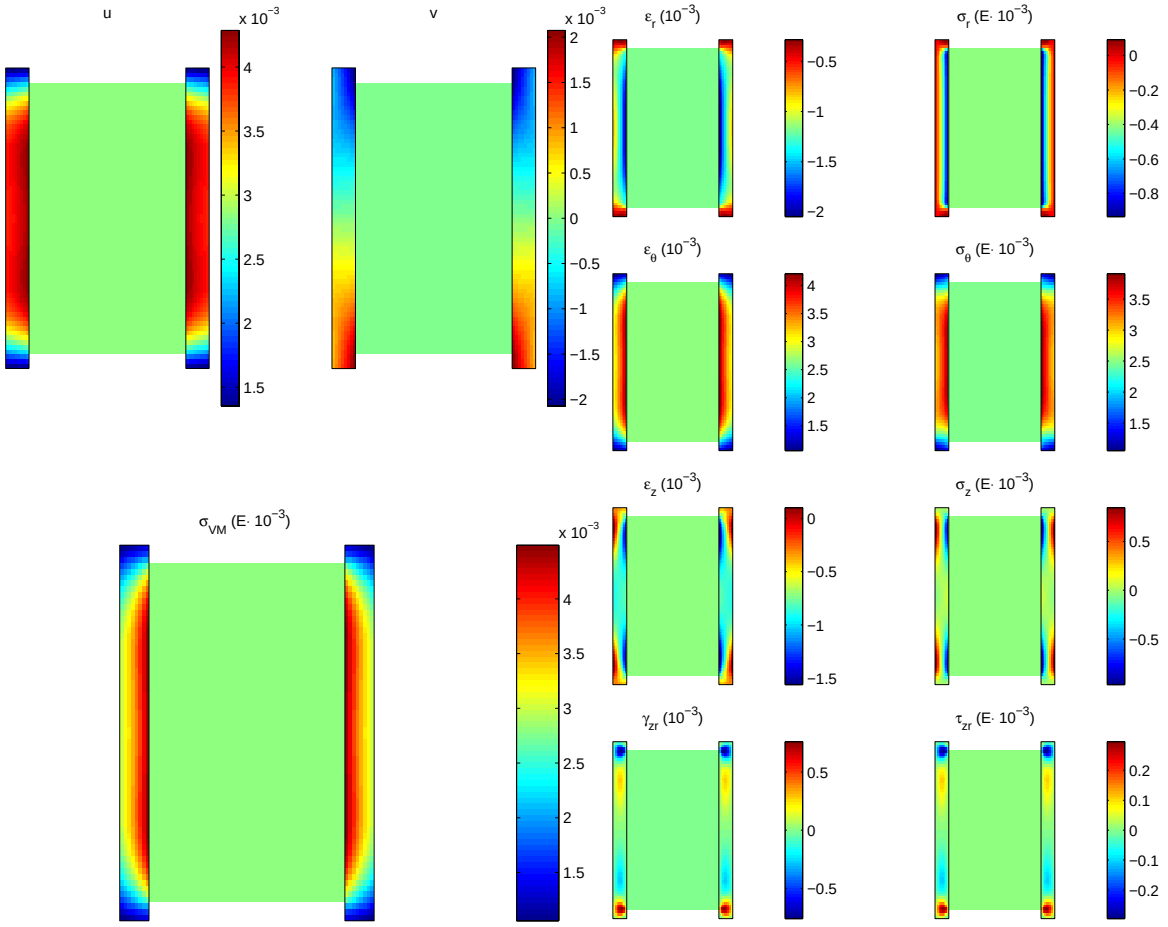

Figure 5.1 Illustration of stresses and strains produced by `plotgridres`.

6. It is now very simple to extract a plot of the material properties, say the equation of state:

```
>> p = GREACthickcylres{2}.concrete.p;
>> eps_v = GREACthickcylres{2}.concrete.eps_v;
>> figure
>> plot(eps_v,p*1e-6)
>> xlabel('Volumetric strain')
>> ylabel('Pressure (MPa)')
```

Note that sometimes the data happens to contain various types of noise. The program should be robust to handle this most of the time, but if a calculated slope turns out to be obviously incorrect (this will always be easy to spot), you have the option of "smoothing the data" by defining a number of interpolation points from the menu. However, usually this should not be necessary.

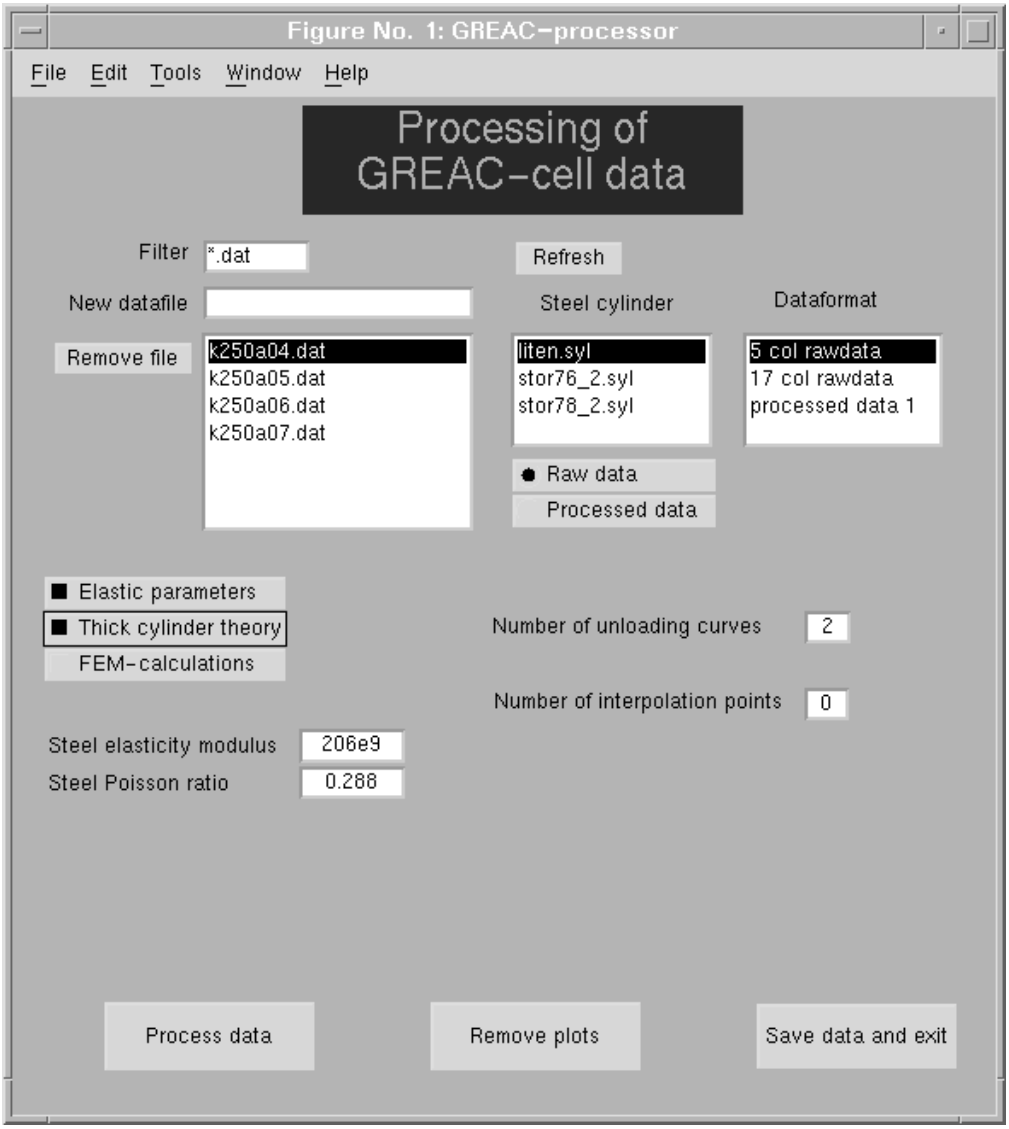

Figure 5.2 The menu based program for processing GREAC–cell data.

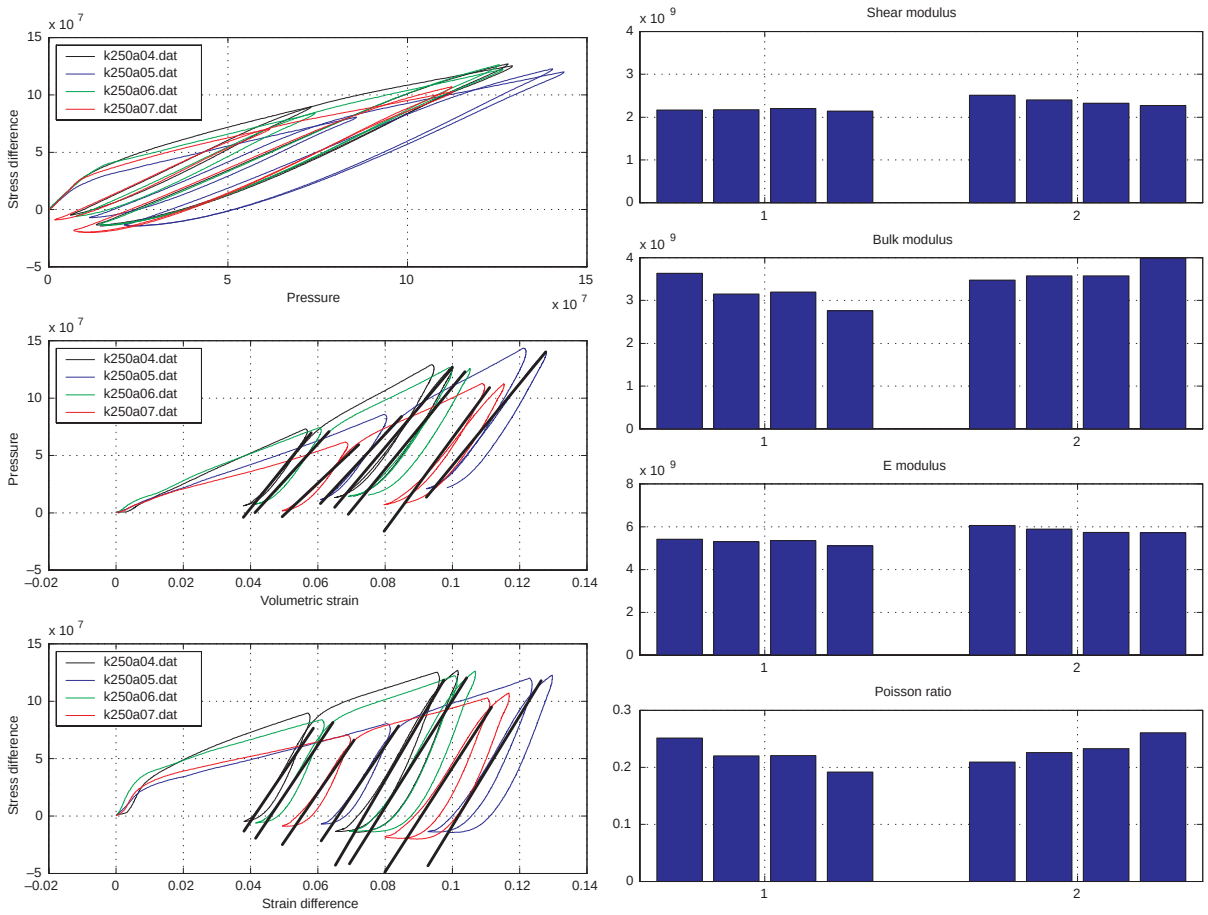

Figure 5.3 Plots of the various results shown on screen by the GREAC–processor.

6 CREATING NEW DATA FOR THE MENU BASED GREAC PROCESSOR.

A result file from the GREAC cell experiments must contain the data in columns separated by a space. Each column contains data for a specific quantity. Comments can be included in the file by putting a '%' in front. As an example, we're using the result file 'k250a04.dat', where the first few lines look like this:

```
%Test Control                Started                    Time:     0.029799998
%          Calendar Time:    18/11/1998 10:05:45
%          File Name:        C:\TS2\TWSX\KALD250A.000
%          Procedure Name:   KALD250A Default Procedure
%          Operator:         MTS
%          Teststar Config:  C:\TS2\CONFIG\KALDH01.tcc
%
%DRP Process Data             data_fagst                 Time:     252.56218
%Time        Length 1    Force 1     Slot 13      Slot 14
%Sec         mm          kN          microstrain  microstrain
0.29899997 -0.013440995 -0.67604387 0.23352112 0.21420741
0.39899999 -0.023777625 -1.1669232 -0.2488136 0.28310552
```

```
0.49899998 -0.054787517 -1.4683404 -0.2488136 0.3520036
0.59899998 -0.13058947 -1.9075483 -0.31771857 0.28310552
0.69899994 -0.19260924 -2.5189946 -0.3866235 -0.061385009
0.79899997 -0.23395576 -3.1821125 -0.17990862 -0.33697739
0.89899993 -0.24773793 -3.6988275 -0.11100368 -0.40587553
0.99899995 -0.25807458 -4.1983185 -0.17990862 -0.47477362
1.0991999 -0.26841122 -4.715034 -0.11100368 -0.68146795
```

These datapoints are read into the following Matlab function:
`greac/GREACmenusubfun/readGREACdata.m`

If one is using another data format, it will become necessary to create a new block in this file, to be able to read the new format correctly. This block can be found in the function `readGREACdata.m`:

```
% ************************************************************
% Make a copy of this block and edit it to add
% a new file format.

% The name of the file format is stored here.  You must also add
% the name in the variable "formatlist" in GRmeny_init.m

FILEFORMAT_NAME = ´Insert name of the file format here´;

 if rawdata & strcmp(dataformat,FILEFORMAT_NAME)
   % Puts the inverse spring constant of the machine to zero by default.
   if isfield(syldata,´k´)
      k = syldata.k;
   else
      k = 0;
   end

%  Quantity                                          Unit        Description
%  ------------------------------------------------------------------------------
   GREACdata.t = data(:,1);                        % s          time
   GREACdata.fz = -data(:,3)*1e3;                  % N          axial force
   GREACdata.uz = data(:,2)*1e-3 + k*GREACdata.fz; % m          axial displacement
   GREACdata.epsteta = data(:,4)*1e-6;             % strain     tangential strain
   GREACdata.epsz = data(:,5)*1e-6;                % strain     axial strain
%  ------------------------------------------------------------------------------
end
% ************************************************************
```

Follow the above description. The data from the result file has been put into the matrix `data`, with rows and colums as in the result file. Now the last part of the block has to be

edited to ensure that data from each column is put into the correct variable. It is very important that the data is given in the units specified above. (For example if the displacement is given in millimeters, it has to be converted to meters in the variable `GREACdata.uz`).

The steel cylinder data has to be stored in a file with the extension `.syl`. The filename itself can be chosen arbitrarily. The following data must be stored:

`L`	Length (m)
`a`	Inner radius (m)
`aob`	Ratio between inner and outer radius (1)
`k`	The inverse of the machine's spring constant (m/N). Default value is 0.

The applied units are given in the parantheses.

Use the following command to store the variables:

```
>> save filnavn.syl a aob L k -mat
```

If these are the only variables stored, only thick cylinder theory can be used to process the data. To use FEM–calculations, follow the instructions in the next chapter.

7 CALCULATIONS OF STRESSES AND STRAINS FOR USE IN THE FEM–BASED PROCESSING OF THE GREAC–CELL RESULTS

Here we describe how to create data for use in a FEM–based processing of the the GREAC–cell test.

The first task is to copy the file `p766/ml/greac/makesteelcylinderdata.m` to your directory. Then use a text editor to edit the file and insert data for the steel cylinder. The file may look something like this:

```
% MAKESTEELCYLINDERDATA Script which produces data for GREAC-cell processing
% Calculates results for various heights of the concrete
% Copy makesteelcylinderdata.m to your directory and make necessary
% changes as described in the file.
% Copy setpar.m to your directory and edit it.

% Adapt the data inside this block
% ************************************

% Filename for stored data (must be on the form *.syl)
filename = 'stor76_2B.syl';

% Inner radius of the steel cylinder (meters)
```

```
a = 0.0762/2;

% Outer radius of the steel cylinder (meters)
b = 0.0508;

% Half the length of the steel cylinder (meters)
L = 0.075;

% Inverse spring constant for the machine (m/N)
%k = 4.1660e-09; % 250kN for the machine at Fellesverkstedet/FFI
k = 0;
% ************************************

% Height of the concrete compared with the steel cylinder
y0 = L*(1-[0:0.02:.5])/a    % in units of a
m = length(y0);

% Pressure normal to the the inside of the steel cylinder
p0 = 1e-3*ones(1,m);         % enhet E

% Shear stress normal to the inside of the steel in positive y-direction
tau0 = -1e-3*ones(1,m);      % enhet E

elementres=[];

% Results with pressure only on the inside of the cylinder
for ii=1:m
   disp(ii)
   elementres{end+1}=greacfem(9,8,32,y0(ii),p0(ii),0,0);

   epsz_p(ii) = elementres{end}.cyloutside.epsy(1);
   epsteta_p(ii) = elementres{end}.cyloutside.epsteta(1);
   uinside_p(ii) = elementres{end}.element{1}.u(1);
end

% Results with shear stress tau0 only on the inside of the cylinder
for ii=1:m
   disp(ii)
   elementres{end+1}=greacfem(9,8,32,y0(ii),0,tau0(ii),0);

   epsz_t(ii) = elementres{end}.cyloutside.epsy(1);
   epsteta_t(ii) = elementres{end}.cyloutside.epsteta(1);
   uinside_t(ii) = elementres{end}.element{1}.u(1);
end
```

```
tau0 = tau0(1);
sigr0 = -p0(1);
aob = a/b;
lengthratio = y0*a/L;

save(filename,'tau0','sigr0','L','a','aob','epsteta_p','epste-
ta_t','epsz_p','epsz_t','lengthratio','uinside_p','uinside_t','k','-mat')
```

This script creates a file with the extension .syl, containing the variables described in Table 7.1. It is very important to use the units that are given in the table. (Here *E* is the elasticity modulus of the steel, while *a* is the inner radius of the cylinder.)

The files with extension `.syl` are read by the program which processes the GREAC–cell results based on a FEM–analysis, and have to contain the variables described in Table 7.1 However, if only `L, a` and `aob` are stored, thick cylinder theory can still be used.

*Table 7.1 Variables stored in *.syl*

Variable	**Description**	**Unit**
`tau0`	The shear stress on the inside boundary of the cylinder .	E
`sigr0`	The radial stress on the inside boundary of the cylinder.	E
`L`	The length of the steel cylinder.	m
`a`	Inner radius of the steel cylinder.	m
`aob`	Ratio between inner and outer radius of the steel cylinder.	1
`lengthratio`	Vector containing the length of concrete over the length of steel.	1
`epsteta_p, epsz_p`	Tangential and axial strain in the middle of the outside boundary of the steel cylinder, when only the inner pressure is applied. These vectors have the same length as `lengthratio.`	1
`epsteta_t, epsz_t`	Same as `epsteta_p` and `apsz_p,` but with applied inner shear stress.	1
`uinside_p`	Radial displacement with only applied inner pressure .	`a`
`uinside_t`	Radial displacement with only applied inner shear stress.	`a`
`k`	The inverse spring constant for the test machine. If not given, it is by default assumed to be zero.	m/N

7.1 List of registered results and data formats

Results from GREAC cell tests performed at FFI, or at Imperial College have been adapted to the GREAC–cell processor. In Table 7.2 result files with corresponding steel cylinder data and data format are given. On running the GREAC–cell processor, the correct combinations have to be given. Notice that the files densit*.prn contain already interpreted data from Imperial College, which means that no data about the steel cylinder is necessary. To enable the use of these data it is also necessary to push the 'Processed data' button.

Table 7.2 Corresponding result files, steel cylinder data and data formats.

Result file	**Steel cylinder file**	**Data format**
k250a*.dat	liten.syl	5 col rawdata
pullen0399_1.prn	stor76_2.syl	17 col rawdata
pullen0399_1a.prn	stor78_2.syl	17 col rawdata
pullen0399_2.prn	stor78_2.syl	17 col rawdata
densit*.prn	———	processed data 1

References

(1) Using Matlab, MathWorks Inc.

(2) Cook R D, Malkus D S, Plesha M E, Concepts and Applications of Finite Element Analysis, Wiley, New York, 1989

(3) Autodyn User Manual, Century Dynamic Inc.

(4) Pullen A D, Testing of fibre–reinforced concrete materials using the GREAC cell, CRIC Client report CRIC/98/FFI/1, September 1998

(5) Chen W F, Plasticity in Reinforced Concrete, McGraw Hill, 1982

APPENDIX

A THE FINITE ELEMENT METHOD FOR STRESS CALCULATIONS IN A HOLLOW CYLINDER

The purpose of the finite element method described here is to calculate the stress and strain distribution in a hollow cylinder that is being loaded from the inside by a given stress. This stress, which is often referred to as the traction, is assumed to be symmetric around the cylinder axis, and as a consequence the resulting stresses and strains will have the same property. The FEM–code is based on a displacement formulation, which means that the displacements are the primary unknowns from which strains and stresses are derived.

A.1 The principle of minimum potensial energy.

In a Finite element method, no attempt is made at solving the differential equations of equilibrium. Instead the solution is found from an equivalent variation principle. In this report, we have chosen to calculate the displacements that correspond to minimum potensial energy. In our problem, this potensial energy is the elastic strain energy which is absorbed by the structure when the inside of the cylinder is loaded by a given traction.

Because of the cylindrical symmetry of the problem, it is convenient to use cylindrical coordinates (r,θ, z) defined in the conventional manner shown in Figure 8.1. We note that due to symmetry requirements there will be no displacements in the angular direction, thus enabling us to only consider the displacement in the r– and z–direction. These components of the displacement vector $\mathbf{u}$ are denoted by u and v.

$$\mathbf{u} + \begin{bmatrix} u \\ v \end{bmatrix}. \tag{A .1}$$

We now define the strain matrix, $\boldsymbol{\epsilon}$, in the following way:

$$\boldsymbol{\epsilon} + \begin{bmatrix} \epsilon_r \\ \epsilon_\theta \\ \epsilon_z \\ \gamma_{zr} \end{bmatrix} + \begin{bmatrix} \frac{\partial}{\partial r} & 0 \\ \frac{1}{r} & 0 \\ 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial z} & \frac{\partial}{\partial r} \end{bmatrix} \mathbf{u}. \tag{A .2}$$

By introducing a more compact notation, we can write Equation (A .2) as:

$$\boldsymbol{\epsilon} + \partial \boldsymbol{u}. \tag{A .3}$$

Similarly, we define a stress matrix, $\boldsymbol{\sigma}$, by

$$\boldsymbol{\sigma} + \begin{bmatrix} \sigma_r \\ \sigma_\theta \\ \sigma_z \\ \tau_{zr} \end{bmatrix}. \tag{A .4}$$

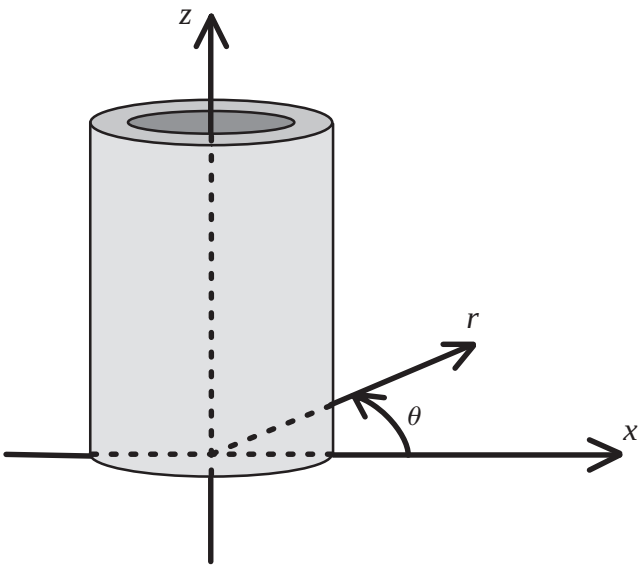

Figure 8.1 Hollow cylinder in a cylindrical coordinate system.

This enables us to express Hooke's law in the following familiar form:

$$\boldsymbol{\sigma} = \mathbf{E}\boldsymbol{\epsilon} = \mathbf{E}\partial\mathbf{u}, \tag{A .5}$$

where the elasticity matrix, **E,** is defined by

$$\mathbf{E} = \frac{(1-\nu)E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1 & f & f & 0 \\ f & 1 & f & 0 \\ f & f & 1 & 0 \\ 0 & 0 & 0 & g \end{bmatrix},$$

and

$$f = \frac{\nu}{1-\nu}\ ,\ g = \frac{1-2\nu}{2(1-\nu)} \tag{A .6}$$

where E is Young's modulus and ν the Poisson ratio.

Our main objective is to calculate the displacement field which gives minimum potensial energy when a traction $\boldsymbol{\Phi}$ is applied to the the inner surface of the hollow cylinder. The potensial energy is known to be given by:

$$\Pi = \int_V \frac{1}{2}\boldsymbol{\epsilon}^T\mathbf{E}\boldsymbol{\epsilon}\mathrm{d}V - \int_S \mathbf{u}^T\boldsymbol{\Phi}\mathrm{d}S, \tag{A .7}$$

When Π is minimal, a small variation of the displacement field will give zero variation of the potensial energy:

$$\delta\Pi = 0. \tag{A .8}$$

This condition will be used to determine the displacement field **u**.

A.2 Formulation of the element method

To apply the finite element method, it is necessary to divide the volume into elements. In our problem we are going to use elements with eight degrees of freedom, as is shown in Figure 9.1. The four nodes are the easily seen to have two degrees of freedom each, namely the displacement in the r - and z- directions, denoted by u_i, and v_i for i=1,2,3,4 respectively.

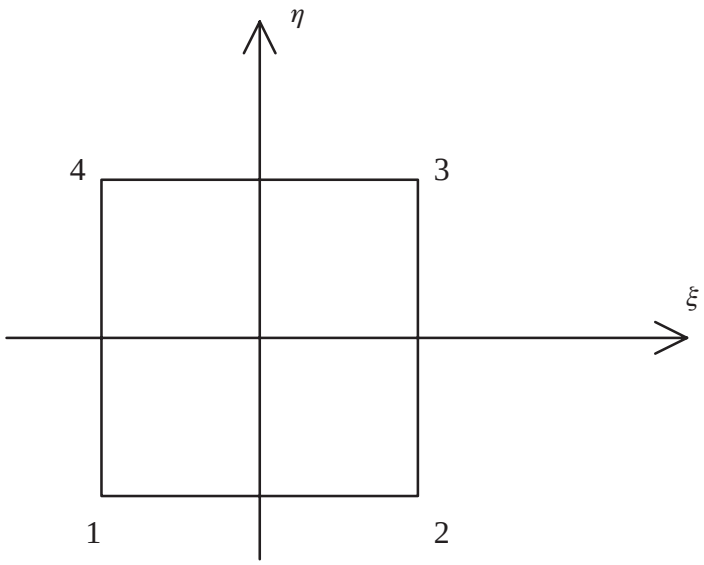

Figure 9.1 Element with four nodes, one in each corner.

A local coordinate system (ξ, η) is defined for each element, with the elements being quadratic and stretching from –1 to 1 in the ξ and η directions. Inside every element we are going to approximate the displacement field with a linear combination of socalled shape functions. These are defined as follows:

$$N_1 = \frac{1}{4}(1 - \xi)(1 - \eta) \qquad \text{(A .9)}$$

$$N_2 = \frac{1}{4}(1 + \xi)(1 - \eta) \qquad \text{(A .10)}$$

$$N_3 = \frac{1}{4}(1 + \xi)(1 + \eta) \qquad \text{(A .11)}$$

$$N_4 = \frac{1}{4}(1 - \xi)(1 + \eta). \qquad \text{(A .12)}$$

We see that the shape functions N_j are defined to have the value 1 in node j and 0 in all other nodes. In addition they are zero at all sides of the element which do not contain node j. The last property implies that the displacement field (which is created by a linear combination of the form functions) at one element side only depends on the values of the nodes at that side. Since displacements on a common boundary of two elements are found by a linear interpolation between the values in the end points of the boundary, we are then guaranteed continuity in the displacement field.

When the displacement field is continuous over the element boundaries and the displacement field inside the elements is unique, we say that the finite element method formulation is compatible. This implies that no holes appear and that the unphysical situation of overlapping materials is not possible.

The displacement field is created by a linear combination of the four shape functions:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} N_1 & N_2 & N_3 & N_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & N_1 & N_2 & N_3 & N_4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}. \tag{A .13}$$

It is easily seen to have the correct values in the node points of the four element corners. This displacement field can also be considered as a bilinear interpolation between the displacement values of the four node points. In a more compact notation, we can write the interpolation as

$$\mathbf{u} = \begin{bmatrix} \mathbf{N}_{0[1\times 4]} & \mathbf{0}_{[1\times 4]} \\ \mathbf{0}_{[1\times 4]} & \mathbf{N}_{0[1\times 4]} \end{bmatrix} \begin{bmatrix} \mathbf{v}_{u[4\times 1]} \\ \mathbf{v}_{v[4\times 1} \end{bmatrix} = \mathbf{N}_{[2\times 8]}\mathbf{v}_{[8\times 1]}, \tag{A .14}$$

In the column vector **v**, the eight degrees of freedom of the element are gathered, and it is therefore called the element degree of freedom vector. The matrix **N** is called the shape function matrix.

From Equation (A .3) we now find the strains in the elements as

$$\boldsymbol{\epsilon} = \partial\mathbf{u} = (\partial\mathbf{N})\mathbf{v}. \tag{A .15}$$

This can be written as:

$$\boldsymbol{\epsilon} = \mathbf{B}\mathbf{v}, \tag{A .16}$$

where **B** is called the strain–displacement matrix, which can be found by letting the operators in the matrix ∂ of Equation (A .2) work on **N.**

However, before this is possible we have to relate the local $\xi\eta$ coordinate system to the global *rz* system. Since the geometry in our problem is very simple, we can let the ξ–axis be parallell with the *r*–axis and the η–axis parallell with the *z*–axis for all elements. In addition, all the elements are given the same size, i.e. length *2a* in radial direction and *2b* in axial direction, so that

$$\frac{\partial}{\partial r} = \frac{1}{a}\frac{\partial}{\partial \xi} \quad , \quad \frac{\partial}{\partial z} = \frac{1}{b}\frac{\partial}{\partial \eta} \tag{A .17}$$

If the geometry had been more complicated, we could have applied the theory of isoparametrical elements. See Chapter 6 of (2) for more information. The strain–displacement matrix **B** is now found to be given by:

$$\mathbf{B} = \begin{bmatrix} \frac{1}{a}\frac{\partial}{\partial \xi}\mathbf{N}_0 & 0 \\ \frac{1}{r_m + a\xi}\mathbf{N}_0 & 0 \\ 0 & \frac{1}{b}\frac{\partial}{\partial \eta}\mathbf{N}_0 \\ \frac{1}{b}\frac{\partial}{\partial \eta}\mathbf{N}_0 & \frac{1}{a}\frac{\partial}{\partial \xi}\mathbf{N}_0 \end{bmatrix}_{[4\times 8]}. \tag{A .18}$$

Here r_m is the distance between the symmetry axis and the origin of the local coordinate system of the element. The final expression for **B** is now easily found from the definitions given in Equations (A .9)–(A .12), and is therefore not reproduced here.

We now consider the strain energy (first integral of Equation (A .7)). Using Equation (A .16), the strain energy from an element filling a volume V_e, can then be expressed as:

$$\frac{1}{2}\int_{V_e} \boldsymbol{\epsilon}^T \mathbf{E} \boldsymbol{\epsilon} \mathrm{d}V = \frac{1}{2}\mathbf{v}^T \left(\int_{V_e} \mathbf{B}^\mathbf{T} \mathbf{E} \mathbf{B} \mathrm{d}V \right) \mathbf{v} = \frac{1}{2}\mathbf{v}^T \mathbf{k} \mathbf{v}. \tag{A .19}$$

The new matrix **k** is called the element stiffness matrix and has dimension 8×8, i.e. it has equally many rows and colums as the element has degrees of freedom.

The work which is done by the traction on the element (second integral in Equation (A .7)), can be expressed as:

$$\int_{S_e} \mathbf{u}^T \boldsymbol{\Phi} \mathrm{d}S = \mathbf{v}^T \int_{S_e} \mathbf{N}^T \boldsymbol{\Phi} \mathrm{d}S = \mathbf{v}^T \mathbf{r}_e, \tag{A .20}$$

where $\mathbf{r}_e$ is called the element load vector and $S_e$is the element surface.

We have now found an expression for the potensial energy for each element. The total strain energy is found by summing the contributions of all elements. This is called assembling:

$$\Pi = \frac{1}{2}\sum_n [\mathbf{v}]_n^T [\mathbf{k}]_n [\mathbf{v}]_n - [\mathbf{v}]_n^T [\mathbf{r}_e]_n \tag{A .21}$$

The index n is called the element number, and the summation runs over all the elements of the problem. We are now able to collect all the degrees of freedom of the problem in a vector **D**, all the element stiffness matrices in a structural stiffness matrix **K**, and the element load vectors in a structural load vector **R**. It is beyond the scope of this report to describe how this is done in practice, so we just refer the interested reader to Chapter 2 of (2) for a more thorough treatment of the assembly technique.

Anyway, Equation (A .21) can eventually be written on the following form:

$$\Pi = \frac{1}{2}\mathbf{D}^T\mathbf{K}\mathbf{D} - \mathbf{D}^T\mathbf{R}. \quad \text{(A .22)}$$

We now return to our initial problem of finding the displacements which make the potensial energy stationary. In our new notation this comes down to finding the components D_j of the column vector **D** that obey

$$\frac{\partial \Pi}{\partial D_j} = 0. \quad \text{(A .23)}$$

On differentiating Equation (A .22), we then obtain the following matrix equation:

$$\mathbf{K}\mathbf{D} = \mathbf{R}. \quad \text{(A .24)}$$

Finally, this equation is solved for **D**.

It is now clear why **K** is called the stiffness matrix. Larger values in **K** implies that larger loads (the elements of **R**) must be applied to obtain a given displacement (the elements of **D**).

We have skipped a couple of points in this simplified derivation. First, at least one point of the cylinder has to be restrained from moving in the axial direction. This is because all solutions with constant displacement in the axial direction will satisfy Equation (A .24), and these solutions are of no interest to us. We can achieve this in several ways, and for the toolbox the method of socalled Lagrange multiplicatiors has been chosen. See Chapter 9.2 of (2) for more information.

Another important point is how to calculate the element stiffness matrix **k.** Since our problem is quite simple, we are able to find an explicit expression for **k**, since only integrals of polynoms are involved. It is, however, easier to calculate **k** numericalled by using a 2×2 Gaussian quadrature, and it can be shown that for our problem this is sufficient to calculate **k** exactly. See Chapter 6 of (2) for a thorough introduction to the theory of choosing quadrature for numerical calculations of the stiffness matrix.

DISTRIBUTION LIST

FFIBM **Dato:** 19 oktober 2000

RAPPORT TYPE (KRYSS AV)	RAPPORT NR	REFERANSE	RAPPORTENS DATO
X RAPP ☐ NOTAT ☐ RR	2000/02032	FFIBM/766/130	19 oktober 2000

RAPPORTENS BESKYTTELSESGRAD	ANTALL EKS UTSTEDT	ANTALL SIDER
UNCLASSIFIED	46	31

RAPPORTENS TITTEL	FORFATTER(E)
MATLAB TOOLBOX FOR PROCESSING OF GREAC CELL DATA	RUSÅS Per–Olav, TELAND Jan Arild

FORDELING GODKJENT AV FORSKNINGSSJEF:	FORDELING GODKJENT AV AVDELINGSSJEF:

EKSTERN FORDELING

ANTALL	EKS NR	TIL
1		FBT
1		v/ Helge Langberg
1		Per–Olav Rusås Sydlibakken 6 1747 SKJEBERG
1		Imperial College, London
1		v/ Andy Pullen
1		DERA
1		v/ Jim Sheridan
1		v/ Cathy O'Carroll
1		TNO
1		v/ Jaap Weerheijm
1		v/ Cyril Wentzel
1		FOA
1		v/ Lennart Ågårdh
1		v/ Håkan Hansson
1		v/ Mattias Unosson

INTERN FORDELING

ANTALL	EKS NR	
14		FFIBIBL
1		Adm direktør/stabssjef
1		FFIE
1		FFISYS
6		FFIBM
1		Bjarne Haugstad, FFIBM
1		Henrik Sjøl, FFIBM
2		Eirik Svinsås, FFIBM
1		Haakon Fykse, FFIBM
1		Harald H Soleng, FFIBM
2		Jan Arild Teland, FFIBM
		FFI–veven

FFI–K1 Retningslinjer for fordeling og forsendelse er gitt i Oraklet, Bind I, Bestemmelser om publikasjoner for Forsvarets forskningsinstitutt, pkt 2 og 5. Benytt ny side om nødvendig.