SEPRAN

SEPRA ANALYSIS

EXTERNAL PACKAGES

GUUS SEGAL

EXTERNAL PACKAGES

August 2014

Ingenieursbureau SEPRA
Park Nabij 3
2491 EG Den Haag
The Netherlands
Tel. 31 - 70 3871309

Contents

1 Introduction

In this manual it is described how you can couple certain external packages with SEPRAN.
At this moment SEPRAN contains interfaces with the linear solver packages PETSC (2) and MUMPS (3). PETSC is an extensive package that provides many linear solvers and allows also the use of other external packages as MUMPS, ML and HYPRE. PETSC can be used both in serial as in parallel programs.
MUMPS can also be used directly without PETSC. It provides a direct linear solver that can be used as serial solver, but also as a parallel one in a serial program.

2 PETSC

In this chapter we describe the use of the PETSC linear solvers within SEPRAN. PETSC can be used to solve linear problems both serially as well as in parallel. PETSC offers interfaces to a number of packages.

2.1 Installation of PETSC

We recommend installing PETSC with the packages MUMPS, HYPRE, ML, METIS and PARMETIS. Use the MPI system that is on your computer, if not, install OPENMPI.

After you have downloaded PETSC in a directory for example `$HOME/petsc` you may install PETSC including these packages.

The following installation script assumes the presence of MPI on your computer

```
cd $HOME/petsc
./configure --with-cc=mpicc --with-cxx=mpicxx --with-fc=mpif90 \
            --download-fblaslapack --download-ml \
            --download-scalapack --download-metis --download-parmetis \
            --download-mumps --download-hypre
```

This takes some time (approximately 30 minutes). After that use:

```
make all test
```

2.2 Installing SEPRAN with a PETSC interface

Once you have installed PETSC (see 2) you may activate the interface between SEPRAN and PETSC by the following steps:

```
cd $SPHOME/bin/update
touch petsc
touch mpi
touch mumps
sepupdate
```

In this step it is assumed that you have already installed SEPRAN. If not the sequence would be:

```
cd $SPHOME/bin/update
touch petsc
touch mpi
touch mumps
sepinstall linux64
```

Furthermore you have to update your `.profile` file in your home directory, with the following statements:

```
export PETSC_DIR=$HOME/petsc
export PETSC_ARCH=arch-linux2-c-debug
export MPI_DIR=/usr/lib/openmpi
export LD_LIBRARY_PATH=${SPHOME}/lib:${MPI_DIR}/lib:${PETSC_DIR}/${PETSC_ARCH}/lib:${LD_LIBRARY_PATH}
```

or if you used other directories the ones you have used.
Presumably the part with `MPI_DIR` may be skipped.
The `export LD_LIBRARY_PATH=...` is necessary to run programs that use PETSC.

2.3 Using PETSC within SEPRAN

Calling of PETSC in SEPRAN is completely determined by the input file. So for sequenctial runs one can use the standard `sepcomp` program.
In case of parallel computing one has to use the option `parallel` in the `sepmesh` input file as described in the Users Manual Section 2.2.
Furthermore to run the parallel program one uses `sepmpi`.

```
sepmpi main_program inputfile
```

Here `main_program` is the standard main program you are using. If you use `sepcomp` in the serial case then use `sepcompmpi` for the parallel case:

```
sepcompmpi inputfile
```

`inputfile` is the name of the input file you are using.
The use of PETSC requires some special options in the input file. The input for the matrix storage must be changed as well as input in the solve block must be adapted.

2.3.1 structure input block

The matrix storage may be given in the structure input block in the following way

```
matrix_structure, options
```

This statement may be used repeatedly.
The following options are available for PETSC

```
petsc
print_level = p
no_sepran_matrix
problem = p
```

Meaning of these options

petsc This option is of course essential to indicate that PETSC should be used.

print_level $= p$ defines the amount of output to be given by this statement.
Default: $p = 0$.

problem $= p$ Defines the problem number corresponding to this definition.
Default value: $p = 1$.

no_sepran_matrix There are two options to build the matrix.
One is to build the matrix by SEPRAN. Afterwards this matrix is mapped into PETSC format.
This is the default option.
The other possibility is to compute the element matrices by SEPRAN and build the matrix directly by PETSC. This option saves memory. Time measurements suggest that this option takes more time in the serial case, but is less time consuming in the parallel case. However, the difference in time is quite small.
This option is currently only available for time-independent problems.

2.3.2 matrix input block

The alternative for the input in the structure input block is input via the matrix input block. In the matrix input block one may give the same input as in the structure input block, however, to indicate the use of PETSC one uses

```
storage_scheme = petsc
```

2.3.3 solve input block

The type of solver to be used by PETSC is defined in the solve block. However, one may also provide an input file `petscrc` in which the PETSC options are defined. If this file is present it is used instead of the options in the solve input block.

```
iterative_solver = m, options
```

The following possibilities for `m` are available

```
cg
cgs
gmres
gcr
bcgs
richardson
chebyshev
fgmres
dgmres
bicg
tfqmr
cr
lsqr
preonly
```

For a description of these iteration methods the user is referred to the PETSC manual.
The choice `preonly` is meant for the case that the preconditioner is a direct solver, like for example `mumps`.
The following options are available

```
preconditioner = p
print_level = p
max_iter = m
accuracy = eps
abs_accuracy = eps
rel_accuracy = eps
div_tolerance = eps
damping_factor = d
emin = e1
emax = e2
ninner = n
num_levels = i
```

The options `print_level = p`, `max_iter = m`, `accuracy = eps` have the usual meaning as described in the SEPRAN users manual.
With respect to the other options the reader is referred to the PETSC manual.
The following preconditioners are available (see PETSC manual).

```
none
ilu
eisenstat
jacobi
bjacobi
sor
icc
asm
gamg
mumps
ml
hypre
```

Not all combinations of iteration method and preconditioner can be used. Consult the PETSC manual.

2.4 Examples of the use of PETSC

In this Section we give some examples of the use of PETSC.
All files mentioned can be found in the directory `$SPHOME/sourceexam/petsc`

2.4.1 A one-dimensional potential problem

The first example is a simple one-dimensional example just to show the use of PETSC.
To get this example in your local directory use

```
sepgetex potential1d
```

To run this example use:

```
sepmesh potential1d.msh
sepcomp potential1d.prb > out
```

The output is stored in the file out.

2.4.2 A two-dimensional potential problem

This example is the two-dimensional equivalent of the previous 1d example. The multigrid method BOOMERANG of HYPRE is used as preconditioner.
To get this example in your local directory use

```
sepgetex potential2dpetsc
```

To run this example use:

```
sepmesh potential2dpetsc.msh
sepcomp potential2dpetsc.prb > out
```

The output is stored in the file `out`.

2.4.3 A parallel two-dimensional potential problem

In this example (2.4.3) we solve a potential problem in parallel on 4 processors.
To get this example in your local directory use

```
sepgetex potential2dpar4
```

To run this example use:

```
sepmesh potential2dpar4.msh
sepcompmpi potential2dpar4.prb
```

The output is stored in the files `sepran_out.001` to `sepran_out.004`.
If you want to inspect the complete solution you have to combine the solutions of all processors and use program `seppost` afterwards:

```
sepcombineout
seppost potential2dpar4.pst
```

2.4.4 An example of the use of MUMPS in PETSC

This example shows the use of MUMPS in PETSC.
To get this example in your local directory use

```
sepgetex potential2dmumps
```

To run this example use:

```
sepmesh potential2dmumps.msh
sepcomp potential2dmumps.prb > out
```

The output is stored in `out`.

2.4.5 A couple of examples of a Stokes equation solved with PETSC

In this series of examples we solve the Stokes equation by Taylor-Hood elements. It concerns the examples

```
stokespetsc1
stokespetsc2
stokespetsc3
stokespetsc4
```

which you can get and run in the same way as the previous examples.
The first example uses linear elements and the second one quadratic elements. The third and fourth examples are the same, but now parallel.

2.4.6 An example of a time-dependent heat equation coupled with Stokes

This example shows the use of PETSC for two sets of equations. The temperature satisfies a time dependent heat equation, driven by the velocity. The velocity is solved by a Stokes equation using the temperature as driving force. In each time step the Stokes equation is solved. Both equations are solved with PETSC.
To get this example in your local directory use

```
sepgetex stokes_heat2dpetsc
```

To run this example use:

```
sepmesh stokes_heat2dpetsc.msh
sepcomp stokes_heat2dpetsc.prb > out
```

The output is stored in `out`.

There are 2 parallel equivalents for this example:

```
stokes_heat2d1par4petsc
stokes_heat2d2par4petsc
```

The first one uses linear Taylor-Hood elements and second one quadratic elements. Mark that we use `reorder levels` in the second case to assure convergence.

2.4.7 Examples of petscrc files

In the directory `$SPHOME/sourceexam/petsc` one can find three examples of `petscrc` files called

```
petscrc.cg
petscrc.mumps
petscrc.hypre
```

If you want to use one of them, copy them into you local directory in a file `petscrc`. Using this file overwrites the options given in the input file.
The file `petscrc.cg` for example has the following contents:

```
-ksp_type cg
-pc_type ilu
```

For a description see the PETSC manual.

3 MUMPS

In this chapter we describe the use of the MUMPS linear solver within SEPRAN. MUMPS is a direct solver using a sophisticated method to reduce the amount of storage and work considerably. An important aspect is the use of a clever renumbering scheme. MUMPS contains renumbering schemes itself but it is recommended to use the METIS renumbering. So install MUMPS including METIS and PARMETIS.
MUMPS can be used as a serial solver but also as a parallel solver but only in a serial program. If you want to use MUMPS as part of a parallel program, use MUMPS through PETSC calls.
Since MUMPS can be stored as part of the PETSC package (2.2) we recommend you to install PETSC with MUMPS, METIS etcetera and not MUMPS separately.

Section 3.1 describes how to install the SEPRAN-MUMPS interface.
In Section 3.2 you can find the steps that have to be performed to use MUMPS in a SEPRAN program.
Some examples of the use of MUMPS are given in Section 3.3.
It is recommended to use MUMPS directly and not via PETSC.

3.1 Installing SEPRAN with a MUMPS interface

There are two ways to install MUMPS. One is installing PETSC with MUMPS as part of the installation. This is the recommended one. If you follow that approach you can skip this Section and use the procedure as described in Section 2.2.
The other option is to install MUMPS directly including the METIS package. See the MUMPS site for the details.
Once you have installed MUMPS by itself, without PETSC, you may activate the interface between SEPRAN and MUMPS by the following steps:

```
cd $SPHOME/bin/update
touch mumps
sepupdate
```

In this step it is assumed that you have already installed SEPRAN. If not the sequence would be:

```
cd $SPHOME/bin/update
touch mumps
sepinstall linux64
```

If you want to run the parallel version (recommended), also use `touch mpi` before running `sepupdate` or `sepinstall`.

Furthermore you have to update your `.profile` file in your home directory, with the following statements:

```
export MUMPS_DIR=...
export METIS_DIR=...
export SCALAPACK_DIR=...
export MPI_DIR=...
export MPICH_LIB=...
export ATLAS_DIR=...
```

Here `...` stands for the directory where these packages can be found. This is of course dependent on your local installation of MUMPS and MPI.

3.2 Using MUMPS within SEPRAN

There are two ways to call MUMPS in SEPRAN.
The first one is the direct way which uses only mumps subroutines.
The other one is calling MUMPS through PETSC, which is described in Section 2.3.

In general the first approach is recommended, even if you installed MUMPS together with PETSC.

3.2.1 Calling a program that uses MUMPS

If MUMPS is used in a serial program the use of MUMPS is only visible in the input part of the computational program.

If you have a serial program, but want to use the MUMPS solver as a parallel solver you have to use the command `sepmumps`. This command should be used as follows:

```
sepmumps main_program inputfile number_of_processors
```

Here `main_program` is the standard main program you are using. If you use the standard sepcomp use sepcompmumps.

```
sepcompmumps inputfile number_of_processors
```

`inputfile` is the name of the input file you are using.
`number_of_processors` defines the number of processors.
For example:

```
sepcompmumps potential.prb 4
```

runs the program sepcomp with input file potential.prb on 4 processors.

If you want to use MUMPS in a parallel program you have to call MUMPS through PETSC.

3.2.2 Input files

The use of MUMPS requires some special options in the input file. The only thing that has to be changed is the input for the matrix storage. This can be done either in the matrix input block or the structure input block. No further input is required, although one might give some extra information in the solve input block.

3.2.3 structure input block

The matrix storage may be given in the structure input block in the following way

```
matrix_structure, options
```

This statement may be used repeatedly.
The following options are available for MUMPS

```
mumps
print_level = p
symmetric
positive_definite
unsymmetric
problem = p
```

Meaning of these options

mumps This option is of course essential to indicate that MUMPS should be used.

print_level = p defines the amount of output to be given by this statement.
Default: $p = 0$.

symmetric indicates that the matrix is symmetric. This reduces the amount of storage required.
Default: unsymmetric.

unsymmetric indicates that the matrix is unsymmetric.
This is the default value.

positive_definite indicates that the matrix is positive definite.
Default: not positive definite.

problem = p Defines the problem number corresponding to this definition.
Default value: $p = 1$.

3.2.4 matrix input block

The alternative for the input in the structure block is input via the matrix input block. In the matrix input block one may give the same input as in the structure input block. In this case to indicate the use of MUMPS one uses

```
storage_scheme = mumps
```

3.2.5 solve input block

No extra input for the solve input block is necessary, but one may indicate what kind of renumbering scheme is used. The following input is available

```
direct_solver = mumps, options
```

The only option available is

```
renum_mumps = v
```

v defines the renumbering sequence number as defined in the MUMPS manual. The default value is 7, which means MUMPS chooses the best one.

3.3 Examples of the direct use of MUMPS

In this Section we give one example of the use of MUMPS, where we have a time-dependent problem (convection-diffusion), coupled with a stationary problem (Stokes).
The example itself is not relevant at this moment.
To get this example into your local directory use:

```
sepgetex rb0mumps
```

To run this example serially use:

```
sepmesh rb0mumps.msh
sepcomp rb0mumps.prb > out
```

The output is stored in the file `out`
To run it in parallel with 4 processors use

```
sepmesh rb0mumps.msh
sepcompmumps rb0mumps.prb 4
```

The output is stored in the files `sepran_out.001` to `sepran_out.004`, where `sepran_out.001` is the only important one.
The files are stored in the directory `$SPHOME/sourceexam/mumps`