PATC – Elmer FEM

The original case

- Apply 3 changes (faster convergence):
  - **`Nonlinear System Convergence Tolerance`** should be smaller than the **`Linear System Convergence Tolerance`**: **`1.0e-08`** → **`1.0e-06`**
  - The Material parameters for heat transfer are constant. Hence this is a <u>linear</u> problem in terms of the variable Temperature. **`Nonlinear System Max Iterations = 20`** → **`1`**
  - For restart: **`Output File = case.result`**

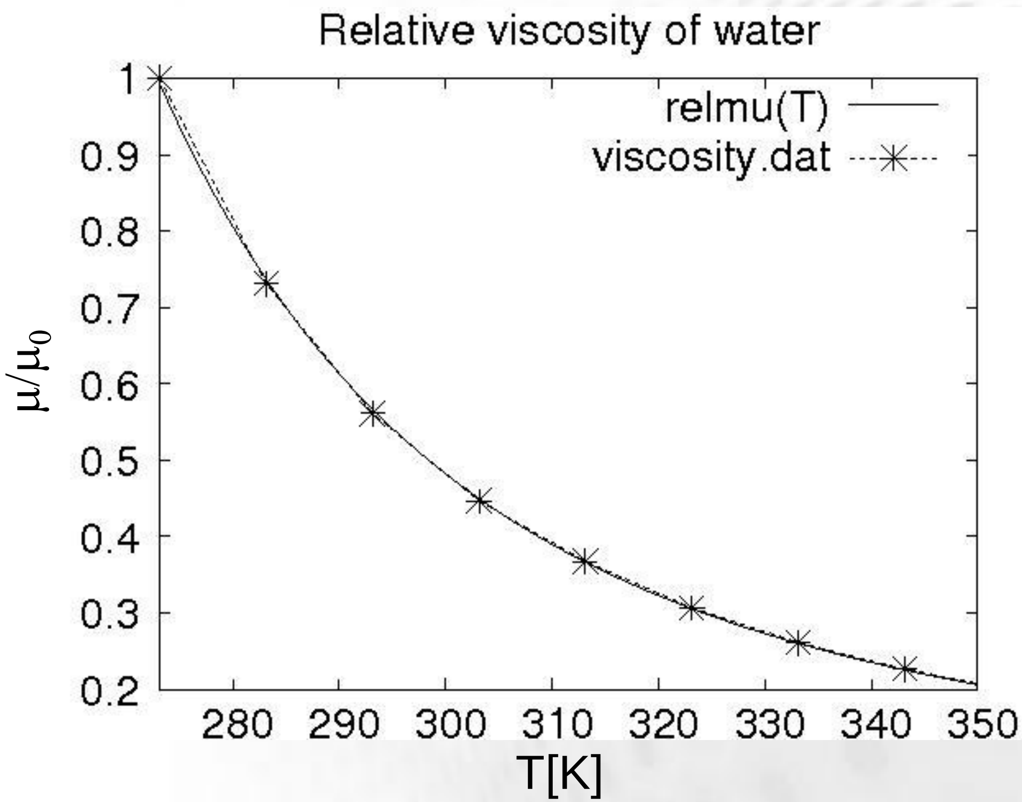
Variations – 2 way coupling

- Temperature dependence of the viscosity for liquid water

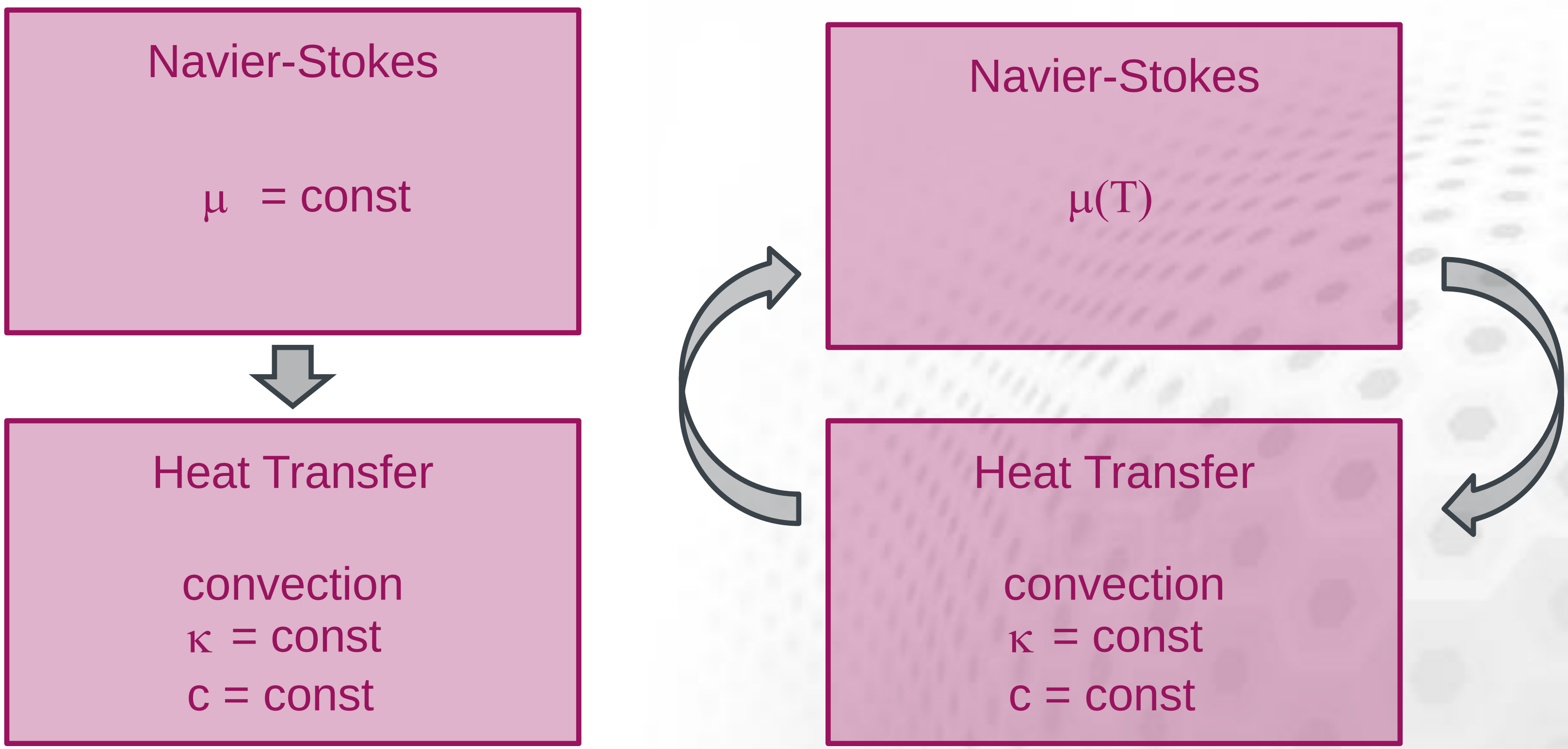
$$\mu/\mu_0 = \exp(-1.704 \;\; -5.306 \;\; 273.15/T + 7.003\ (273.15/T)^2)$$

viscosity.dat

```
273.15  1.788e-3
283.15  1.307e-3
293.15  1.003e-3
303.15  0.799e-3
313.15  0.657e-3
323.15  0.548e-3
333.15  0.467e-3
343.15  0.405e-3
353.15  0.355e-3
363.15  0.316e-3
373.15  0.283e-3
```



Variations – 2 way coupling



`Steady State Max Iterations = 1 → 50`

Variations – 2 way coupling

- Copy the original solver input file (SIF)
- Open in editor of your choice (e.g., gedit)
  - apply the changes as suggested
  - change names of output files!
  - Include restart from earlier case:

```
Restart File = case.result
Restart Position = 0
```

  - The last line restarts from the last entry it found in **case.result**

Variations – 2 way coupling

- Run the case in serial:

  ```
  ElmerSolver name.sif > name.out
  ```

  – Replace ***name.sif*** with the name of the input file

  – Redirect output (good for checking perfomance)

Array 1

- Piecewise linear interpolation
- Alternative:

  `Real cubic`

  interpolates using cubic splines
- See SIF:

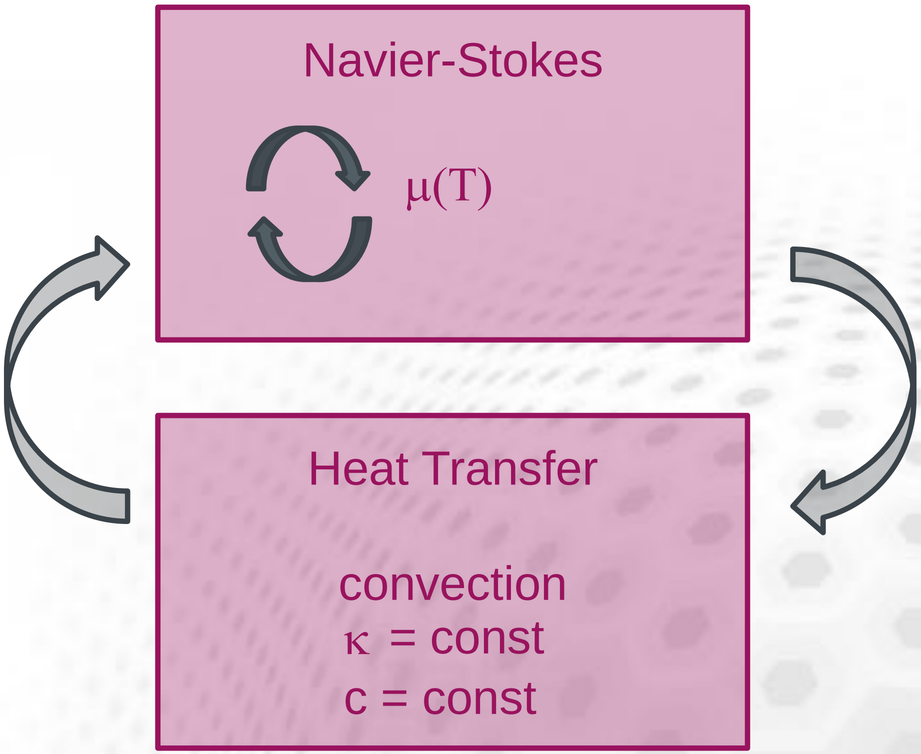
`coupled_array.sif`

```
Material 1
  Name = "Water (room temperature)"
  Viscosity = Variable Temperature
    Real
      273.15 1.788e-3 ! 0 Celsius
      283.15 1.307e-3
      293.15 1.003e-3
      303.15 0.799e-3
      313.15 0.657e-3
      323.15 0.548e-3
      333.15 0.467e-3
      343.15 0.405e-3
      353.15 0.355e-3
      363.15 0.316e-3
      373.15 0.283e-3 ! 100 Celsius
   End
```

Array 2

- Same as before, but now we switch to only one non-linear iteration For Navier-stokes
- See SIF:

`coupled_array_var.sif`



`Nonlinear System Max Iterations = 50` → `1`

MATC function

- Declare outside sections:
  - Constant **`mu0`**
  - Function **`relativevisc`**

```
$ mu0 =  1.788e-3

$ function relativevisc(T){\

  a = -1.704;\

  b = -5.306;\

  c = 7.003;\

  z = 273.15/T;\

  _relativevisc = exp(a + b * z + c *(z^2));\

}
```

- Call both using MATC from within **`Material 1`**

```
Material 1
  Name = "Water (room temperature)"
  Viscosity = Variable Temperature
    Real MATC "mu0 * relativevisc(tx)"
```

User Defined Function (UDF)

- Write a simple UDF in Fortran 90 that returns the value of viscosity from a given value of temperature **`viscosity1.f90`**
  - Pre-defined Header:

```
FUNCTION getWaterViscosity( Model, N, temperature ) &
RESULT(viscosity)
  USE DefUtils
  IMPLICIT NONE
  !------------ external variables ---------------------------
  TYPE(Model_t) :: Model
  INTEGER :: N
  REAL(KIND=dp) :: temperature, viscosity
```

NB for F90: exponential function … exp()   multiplication … *

User Defined Function (UDF)

- Compile it:

```
elmerf90 viscosity1.f90 -o
  viscosity1
```

- Re-write the Material 1 section:

```
Material 1
  Name = "Water (room temperature)"
  Viscosity = Variable Temperature
    Procedure "viscosity1" "getWaterViscosity"
```