water flow modelling in the subterranean layer by the finite volumes methods in Fortran.

Olivier DOBEK
Frederic GODET

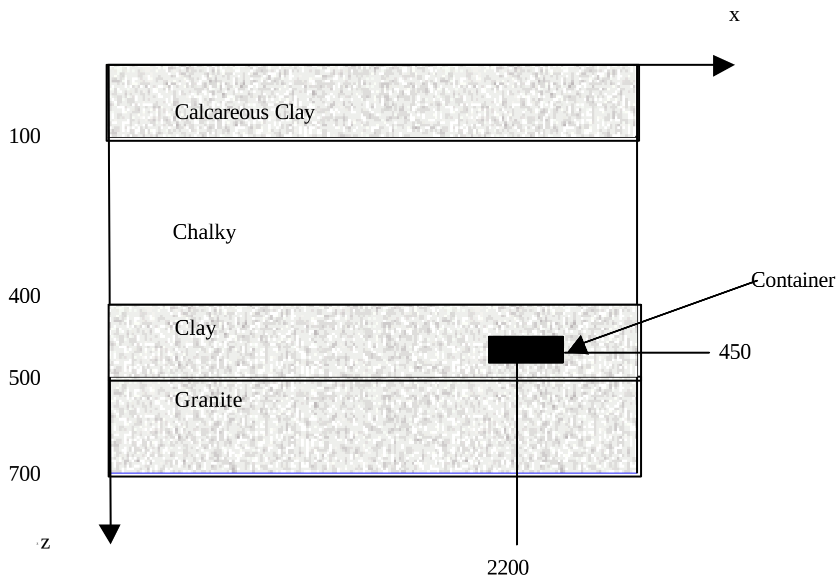
---

1)Objective:

A ground, initially clean, is supposed. A firm wants to bury polluting waste shut up in a container. A preliminary risk study simulates a leak in the container and the rain flow impact.

Firstly, calculation module is created to determinate, with the finite volumes method, the hydrodynamic charge in the inside ground. Then the speed field is deducted.

Secondly, we take into account of the ground morphology.



2)Equations:

We consider that all the rocky layers are saturated with water and that the hydraulic charge boundaries conditions are constant and known. The flow is stationary. The Darcy low allows us to determine the speed (u, w) according to the hydraulic charge.

We have: (1) $\boxed{U = -K\ \nabla H}$ and U components are u,w ($u = -K\dfrac{\partial H}{\partial x}$; $w = -K\dfrac{\partial H}{\partial z}$)

K, the permeability tensor, is constant in every rocky layer.

The mass keeping low, supposing that the voluminal mass is constant, gives us the relation as follows:

(2) $\boxed{\nabla(K\ \nabla H) = 0 \quad ou \quad \dfrac{\partial}{\partial x}(K\dfrac{\partial H}{\partial x}) + \dfrac{\partial}{\partial z}(K\dfrac{\partial H}{\partial z}) = 0}$ in all the field.

3)Data:

Rock	Calcareous Clay	Chalky	Clay	Granite
K	$3.153\ 10^{-5}$	6.3072	$3.1536\ 10^{-6}$	25.288

The boundaries are :

Right face : (x =2500)
H=310 for 100 < y < 400
H=289 for 500 < y < 700

Left Face : (x =0)
H=200 for 100 < y < 400
H=216 for 500 < y < 700

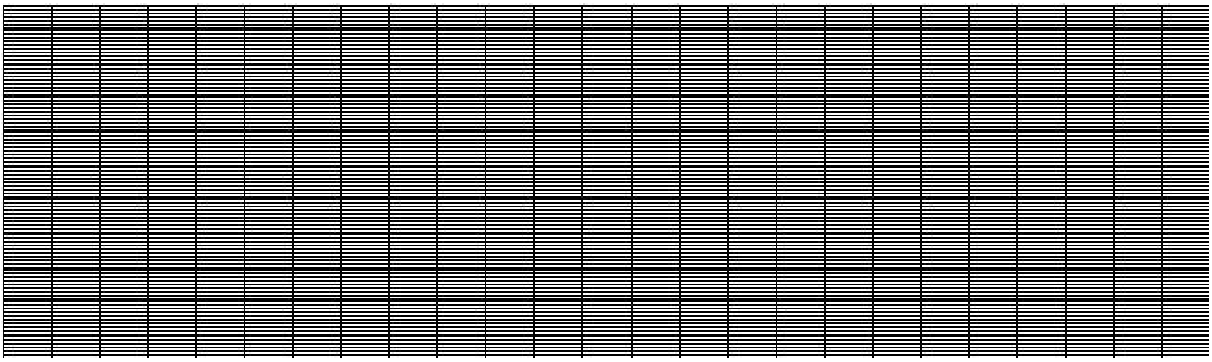
North Face (z=0)
H= 180 + 160.(x / 2500) for 0 < x < 2500

Waterproof everywhere . (dH/dn = 0)

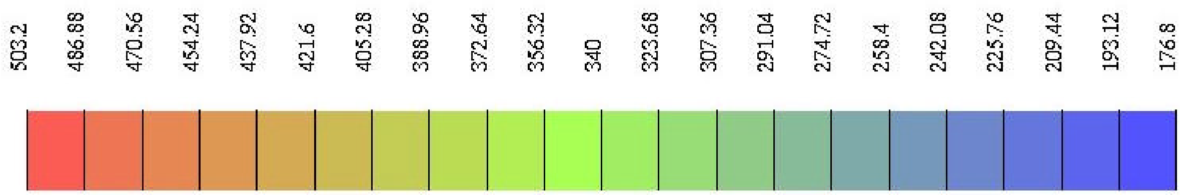
4)Different stages of the programming:

4.1)With only one layer

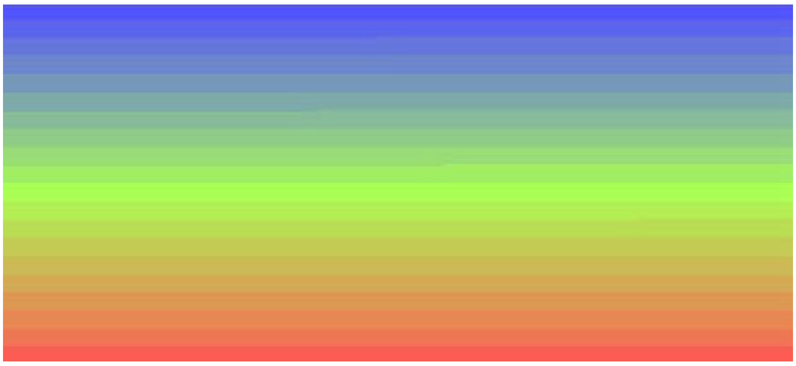
The mesh is as follows:



The colourful hydraulic charge scale is as follows:

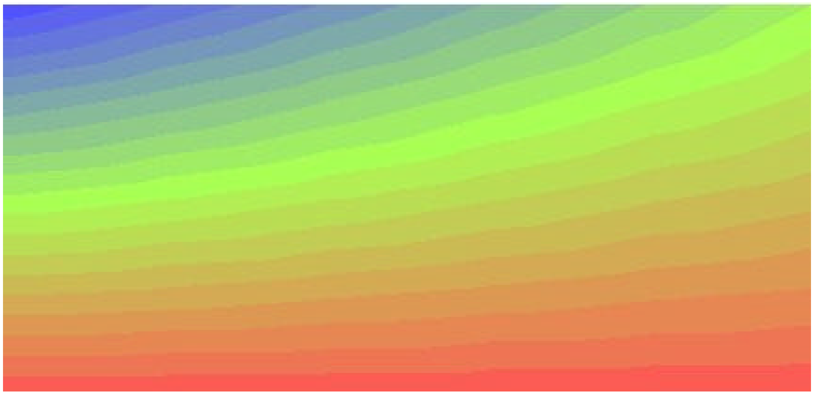


1) Only one layer
Compulsory charge for z=0 H=180 and z=zmax H=500
Adiabatic Condition for x=0 and x=2500



Hydraulic charge is linear and correct.

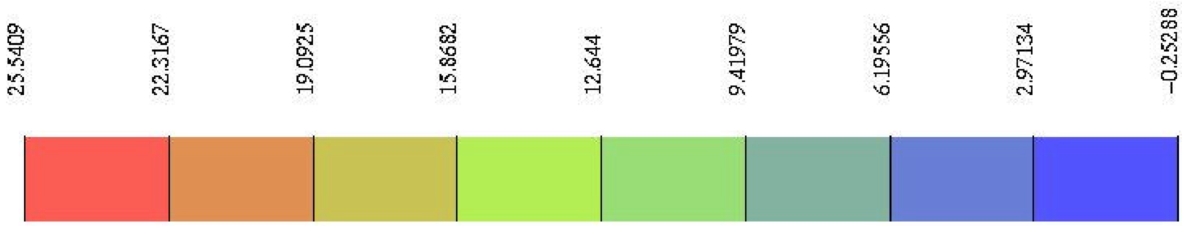
2) Only one layer
Compulsory charge for z=0 H=180+160*(x/2500) and z=zmax H=500
Adiabatic Condition for x=0 et x=2500



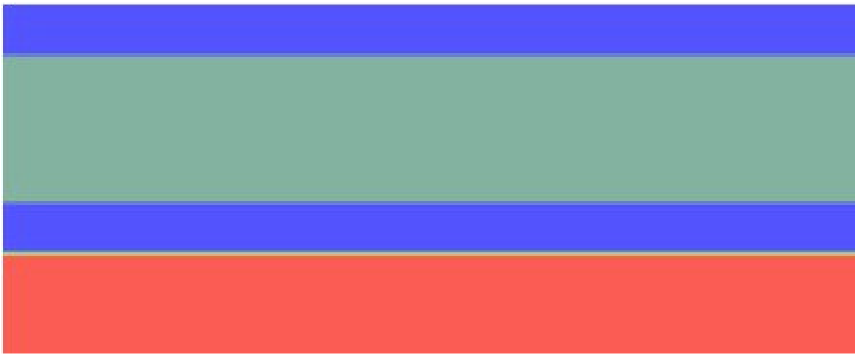
For z=0, the charge increases linearly from 180 to 340.

4.2)Hydraulic charges and speed calculations with linear layers :

<u>4.2.1)Permeability</u>

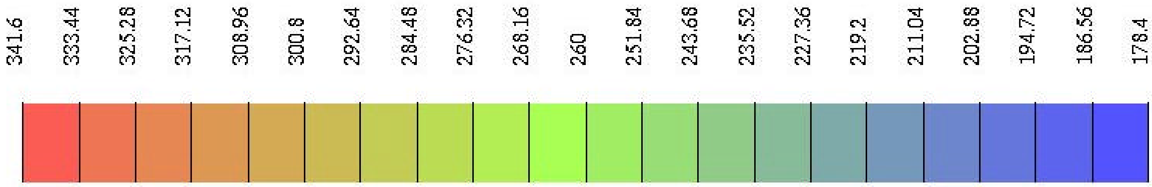


Permeability scale

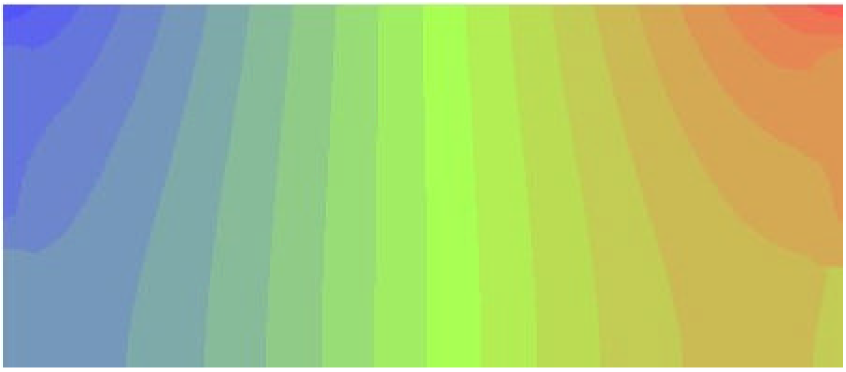


Permeability profile

4.2.2) Hydraulic charge :

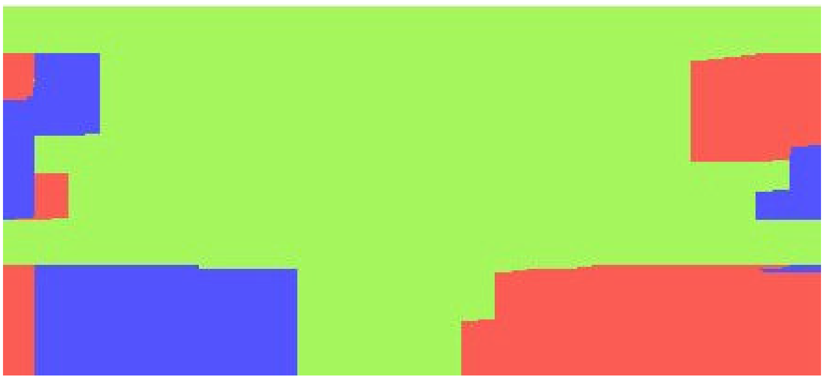


hydraulic charge scale

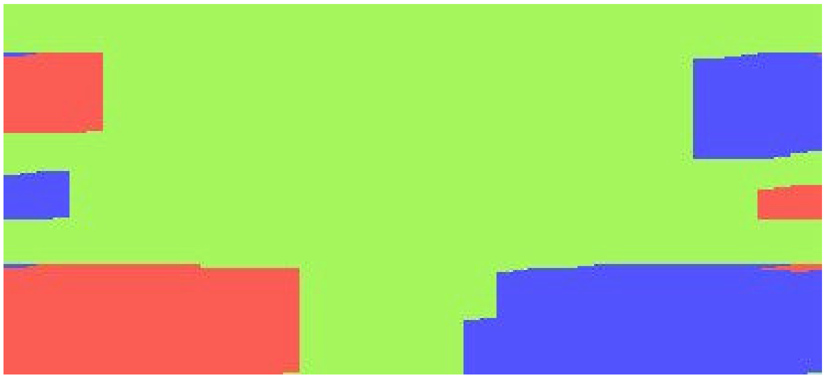


hydraulic charge profile

4.2.3) Speed profile and conclusion:



component u



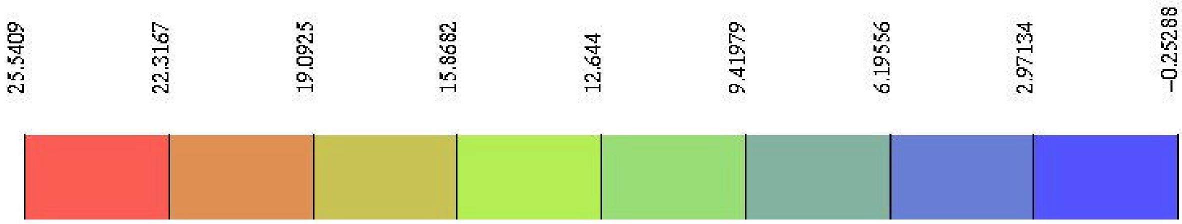
component w

To see the speed, we selected only three colours from –1^{e}-06 to 1^{e}-06 and speed higher in absolute value than 3.6 mm/ hour is only visible. We think that speed in the green area (|V|< 3.6 mm/h) is low if there is a leak in the container. The red and blue areas must be excluded.

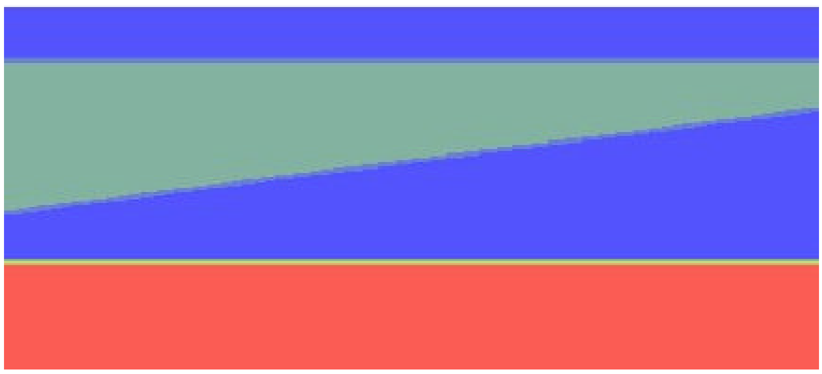
By component u and w superposition, we can selected the best position of the container which is in the green area, as far as possible of the red and blue areas.
It's the same conclusion for the non-linear layers.

4.3)Hydraulic charges and speed calculations with non-linear layers :

4.3.1)Permeability

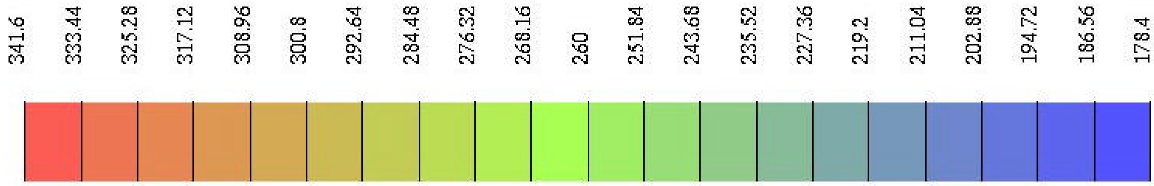


Permeability scale

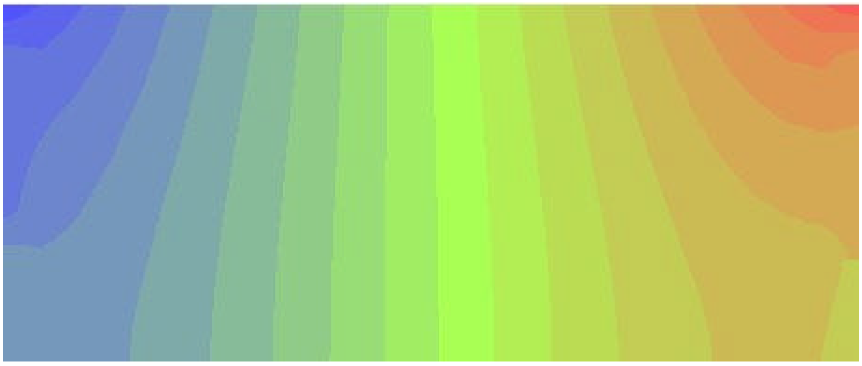


Permeability profile

4.3.2) hydraulic charge :

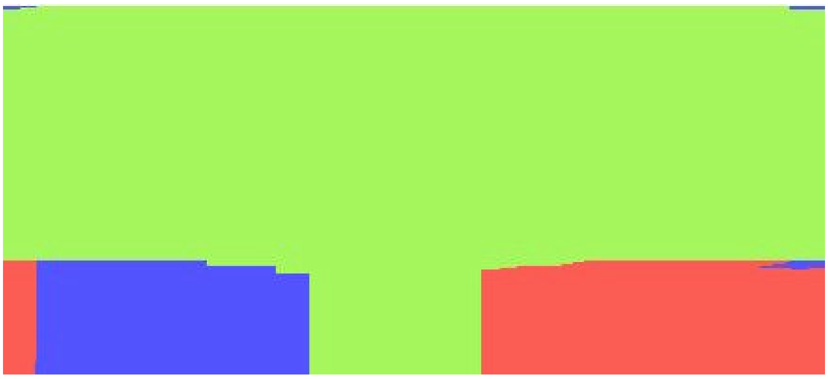


hydraulic charge scale

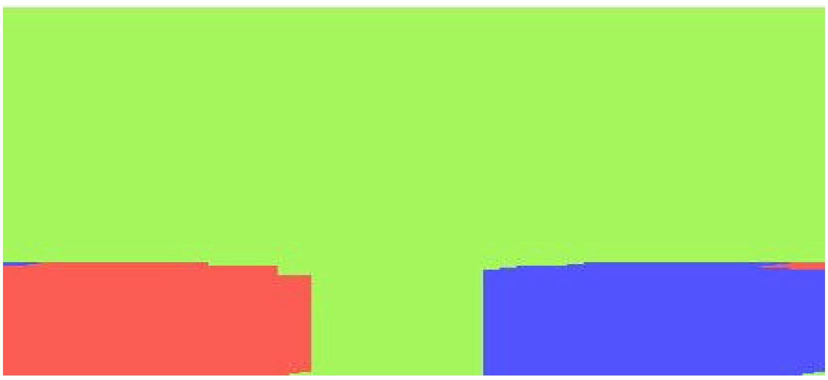


hydraulic charge profile

4.3.3) Speed profile



component u



component w

5)Programming listing :

```
MODULE DONNEES
IMPLICIT NONE
DOUBLE PRECISION,DIMENSION(:,:),ALLOCATABLE::Ae,Aw,An,As,Ap,B,H,U,W,KK,HH,UU,&
                     WW,KKK
DOUBLE PRECISION,DIMENSION(:),ALLOCATABLE::X,Y

DOUBLE PRECISION::Kmarne,Kcalcaire,Kargile,Kgranit,Xmin,Xmax,Zmin,&
            Zmarne,Zcalcaireg,Zcalcaired,Zargile,Zgranit,pasx,pasz

character (len=30) ::finh,fink,finu,finw
character (len=30) ::name

INTEGER::Nx,Nz
END MODULE DONNEES

module Vigie
 interface
   subroutine  fvigie ( name, n, m, x, y, temp)
    implicit none
    integer , intent (in) :: n , m
    double precision  , intent(in) :: x(n) , y(m) , temp(n,m)
    character  (len =*) , intent (in) :: name
  end subroutine fvigie
 end interface
end module Vigie

MODULE INTERFACE


  INTERFACE
    SUBROUTINE RESOLUTIONH(Nx,Nz,Aw,Ae,An,As,B,Ap,HH)

      INTEGER,INTENT(IN)::Nx,Nz
      DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(IN)::Ae,Aw,An,As,Ap,B
      DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::HH

    END SUBROUTINE

  END INTERFACE
END MODULE INTERFACE

MODULE INTERAFFICHE

   INTERFACE
    SUBROUTINE AFFICHE(Nx,Nz,HH,UU,WW)

      INTEGER::Nx,Nz
      DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::HH,UU,WW

    END SUBROUTINE
   END INTERFACE

END MODULE INTERAFFICHE


MODULE INTERVITESSE

   INTERFACE
     SUBROUTINE VITESSE(Nx,Nz,Nmarne,Ncalcaire,Nargile,Ngranit,pasx,pasz,&
               Kmarne,Kcalcaire,Kargile,Kgranit,Kmarca,&
```

```
                  Kcalargile,Kargilgranit,H,UU,WW)

              INTEGER,INTENT(IN)::Nx,Nz,Nmarne,Ncalcaire,Nargile,Ngranit
              DOUBLE PRECISION,INTENT(IN)::pasx,pasz,Kmarne,&
                          Kcalcaire,Kargile,Kgranit,&
                          Kmarca, Kcalargile,Kargilgranit
              DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(IN)::H
              DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::UU,WW

           END SUBROUTINE VITESSE
         END INTERFACE

END MODULE INTERVITESSE


MODULE INTERFACEK

 INTERFACE
 SUBROUTINE CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J,Kmarne,&
Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)

 INTEGER,INTENT(IN)::Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J
 DOUBLE PRECISION,INTENT(IN)::Kmarne,Kmarca,Kcalcaire,&
                  Kcalargile,Kargile,Kargilgranit,Kgranit
 DOUBLE PRECISION,INTENT(OUT)::K

  END SUBROUTINE CALCULK
 END INTERFACE
END MODULE INTERFACEK

MODULE INTERDEMARRE
        INTERFACE
SUBROUTINE DEMARRE(Xmax,Xmin,Zgranit,Zmin,Zmarne,Zcalcaired,Zcalcaireg,Zargile&
       ,pasx,pasz,Nx,Nz,Nmarne,Ncalcaired,Ncalcaireg,Nargile,Kmarne&
       ,Kcalcaire,Kargile,Kgranit,Kmarca,Kcalargile,Kargilgranit)


DOUBLE PRECISION,INTENT(IN)::Xmax,Xmin,Zgranit,Zmin,Zmarne,Zcalcaired,&
                  Zcalcaireg,Zargile,Kmarne,Kcalcaire,Kargile,Kgranit
DOUBLE PRECISION,INTENT(OUT)::pasz,Kmarca,Kcalargile,Kargilgranit
DOUBLE PRECISION,INTENT(INOUT)::pasx
INTEGER,INTENT(INOUT)::Nx
INTEGER,INTENT(OUT)::Nz,Ncalcaired,Ncalcaireg,Nmarne,Nargile

      END SUBROUTINE DEMARRE
   END INTERFACE

END MODULE INTERDEMARRE


MODULE TABVIGI

  INTERFACE
  SUBROUTINE TABLEAUVIGIE(Nx,Nz,pasx,pasz,H,U,W,KK,HH,UU,WW,KKK,X,Y)

     INTEGER,INTENT(IN)::Nx,Nz
     DOUBLE PRECISION,INTENT(IN)::pasx,pasz
     DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(IN)::H,U,W,KK
     DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::HH,UU,WW,KKK
     DOUBLE PRECISION,DIMENSION(1:Nx),INTENT(OUT)::X
     DOUBLE PRECISION,DIMENSION(1:Nz),INTENT(OUT)::Y
```

```
   END SUBROUTINE
  END INTERFACE

END MODULE TABVIGI

PROGRAM MARNE

USE DONNEES
USE INTERFACE
USE vigie
USE INTERAFFICHE
USE INTERVITESSE
USE INTERFACEK
USE INTERDEMARRE
USE TABVIGI

INTEGER::Jligne,Icolonne,Nmarne,Ncalcaire,Nargile,I,J
DOUBLE PRECISION::Kmarca,Kcalargile,Kargilgranit,K


Jligne=0
Icolonne=0


OPEN(16,FILE='donnees3.dat',STATUS='OLD',FORM='FORMATTED',ACCESS='SEQUENTIAL')
READ(16,*) pasx
READ(16,*) Xmin
READ(16,*) Xmax
READ(16,*) Zmin
READ(16,*) Zmarne
READ(16,*) Zcalcaireg
READ(16,*) Zcalcaired
READ(16,*) Zargile
READ(16,*) Zgranit
READ(16,*) Kmarne
READ(16,*) Kcalcaire
READ(16,*) Kargile
READ(16,*) Kgranit

CLOSE (UNIT=16,STATUS='KEEP')

CALL DEMARRE(Xmax,Xmin,Zgranit,Zmin,Zmarne,Zcalcaired,Zcalcaireg,Zargile&
        ,pasx,pasz,Nx,Nz,Nmarne,Ncalcaired,Ncalcaireg,Nargile,Kmarne&
        ,Kcalcaire,Kargile,Kgranit,Kmarca,Kcalargile,Kargilgranit)


ALLOCATE(Ae(1:Nx,1:Nz))
ALLOCATE(Aw(1:Nx,1:Nz))
ALLOCATE(An(1:Nx,1:Nz))
ALLOCATE(As(1:Nx,1:Nz))
ALLOCATE(Ap(1:Nx,1:Nz))
ALLOCATE(B(1:Nx,1:Nz))
ALLOCATE(H(1:Nx,1:Nz))
ALLOCATE(U(1:Nx,1:Nz))
ALLOCATE(W(1:Nx,1:Nz))
ALLOCATE(KK(1:Nx,1:Nz))
ALLOCATE(HH(1:Nx,1:Nz))
ALLOCATE(UU(1:Nx,1:Nz))
ALLOCATE(WW(1:Nx,1:Nz))
ALLOCATE(KKK(1:Nx,1:Nz))
ALLOCATE(X(1:Nx))
```

```
ALLOCATE(Y(1:Nz))


DO Icolonne=2,(Nx-1)
  DO Jligne=2,(Nz-1)

CALL CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,Icolonne,Jligne,Kmarne,&
 Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)


      Ae(Icolonne,Jligne)= K*pasz*2/pasx
      Aw(Icolonne,Jligne)= K*pasz*2/pasx
      An(Icolonne,Jligne)= K*pasx*2/pasz
      As(Icolonne,Jligne)= K*pasx*2/pasz
      B(Icolonne,Jligne)=0
      Ap(Icolonne,Jligne)=  Ae(Icolonne,Jligne)+ Aw(Icolonne,Jligne)+&
                       An(Icolonne,Jligne)+ As(Icolonne,Jligne)+&
                       B(Icolonne,Jligne)
  END DO
END DO


Jligne=0
Icolonne=0

DO Icolonne=1,Nx
 Ap(Icolonne,1)=1
 Ae(Icolonne,1)=0
 Aw(Icolonne,1)=0
 An(Icolonne,1)=0
 As(Icolonne,1)=0
 B(Icolonne,1)=180+160*(Icolonne-1)*pasx/Xmax
END DO


DO Jligne=2,Nmarne
  Ae(1,Jligne)=Kmarne*pasz*2/pasx
  Aw(1,Jligne)=0
  An(1,Jligne)=Kmarne*pasx*2/pasz
  As(1,Jligne)=Kmarne*pasx*2/pasz
  B(1,Jligne)=0  !flux nul
  Ap(1,Jligne)= Ae(1,Jligne)+ An(1,Jligne)+ As(1,Jligne)
END DO

Jligne=Nmarne+1
  Ae(1,Jligne)=Kmarca*pasz*2/pasx
  Aw(1,Jligne)=0
  An(1,Jligne)=Kmarca*pasx*2/pasz
  As(1,Jligne)=Kmarca*pasx*2/pasz
  B(1,Jligne)=0  !flux nul
  Ap(1,Jligne)= Ae(1,Jligne)+ An(1,Jligne)+ As(1,Jligne)


DO Jligne=2,Nmame
  Aw(Nx,Jligne)=Kmarne*pasz*2/pasx
  Ae(Nx,Jligne)=0
  An(Nx,Jligne)=Kmarne*pasx*2/pasz
  As(Nx,Jligne)=Kmarne*pasx*2/pasz
  B(Nx,Jligne)=0  !flux nul
  Ap(Nx,Jligne)= Aw(Nx,Jligne)+ An(Nx,Jligne)+ As(Nx,Jligne)
END DO
```

```
 Jligne=Nmarne+1
   Aw(Nx,Jligne)=Kmarca*pasz*2/pasx
   Ae(Nx,Jligne)=0
   An(Nx,Jligne)=Kmarca*pasx*2/pasz
   As(Nx,Jligne)=Kmarca*pasx*2/pasz
   B(Nx,Jligne)=0  !flux nul
   Ap(Nx,Jligne)= Aw(Nx,Jligne)+ An(Nx,Jligne)+ As(Nx,Jligne)


DO Jligne=(Nmarne+2),Ncalcaired+1
   Aw(Nx,Jligne)=0
   Ae(Nx,Jligne)=0
   An(Nx,Jligne)=0
   As(Nx,Jligne)=0
   B(Nx,Jligne)=310.
   Ap(Nx,Jligne)= 1.
END DO

DO Jligne=(Nmarne+2),Ncalcaireg+1
   Ae(1,Jligne)=0
   Aw(1,Jligne)=0
   An(1,Jligne)=0
   As(1,Jligne)=0
   B(1,Jligne)=200.
   Ap(1,Jligne)= 1
END DO


DO Jligne=Ncalcaireg+2,Nargile
   Ae(1,Jligne)=Kargile*pasz*2/pasx
   Aw(1,Jligne)=0
   An(1,Jligne)=Kargile*pasx*2/pasz
   As(1,Jligne)=Kargile*pasx*2/pasz
   B(1,Jligne)=0  !flux nul
   Ap(1,Jligne)= Ae(1,Jligne)+ An(1,Jligne)+ As(1,Jligne)
END DO

Jligne=Nargile+1
   Ae(1,Jligne)=Kargilgranit*pasz*2/pasx
   Aw(1,Jligne)=0
   An(1,Jligne)=Kargilgranit*pasx*2/pasz
   As(1,Jligne)=Kargilgranit*pasx*2/pasz
   B(1,Jligne)=0  !flux nul
   Ap(1,Jligne)= Ae(1,Jligne)+ An(1,Jligne)+ As(1,Jligne)


DO Jligne=Ncalcaired+2,Nargile
   Aw(Nx,Jligne)=Kargile*pasz*2/pasx
   Ae(Nx,Jligne)=0
   An(Nx,Jligne)=Kargile*pasx*2/pasz
   As(Nx,Jligne)=Kargile*pasx*2/pasz
   B(Nx,Jligne)=0  !flux nul
   Ap(Nx,Jligne)= Aw(Nx,Jligne)+ An(Nx,Jligne)+ As(Nx,Jligne)
END DO

 Jligne=Nargile+1
   Aw(Nx,Jligne)=Kargilgranit*pasz*2/pasx
   Ae(Nx,Jligne)=0
   An(Nx,Jligne)=Kargilegranit*pasx*2/pasz
   As(Nx,Jligne)=Kargilgranit*pasx*2/pasz
   B(Nx,Jligne)=0  !flux nul
   Ap(Nx,Jligne)= Aw(Nx,Jligne)+ An(Nx,Jligne)+ As(Nx,Jligne)
```

```
DO Jligne=(Nargile+2),Nz
  Aw(Nx,Jligne)=0
  Ae(Nx,Jligne)=0
  An(Nx,Jligne)=0
  As(Nx,Jligne)=0
  B(Nx,Jligne)=289.
  Ap(Nx,Jligne)= 1.
END DO

DO Jligne=(Nargile+2),Nz
  Ae(1,Jligne)=0
  Aw(1,Jligne)=0
  An(1,Jligne)=0
  As(1,Jligne)=0
  B(1,Jligne)=216.
  Ap(1,Jligne)= 1
END DO

DO I=2,Nx-1
 Aw(I,Nz)=Kgranit*pasz*2/pasx
 Ae(I,Nz)=Kgranit*pasz*2/pasx
 An(I,Nz)=0
 As(I,Nz)=Kgranit*pasx*2/pasz
 B(I,Nz)=0  !flux nul
 Ap(I,Nz)=Aw(I,Nz)+Ae(I,Nz)+As(I,Nz)
END DO


CALL RESOLUTIONH(Nx,Nz,Aw,Ae,An,As,B,Ap,H)

CALL VITESSE(Nx,Nz,Nmarne,Ncalcaire,Nargile,Ngranit,pasx,pasz,&
          Kmarne,Kcalcaire,Kargile,Kgranit,Kmarca,&
          Kcalargile,Kargilgranit,H,U,W)

DO J=1,Nz
  DO I=1,Nx
 CALL CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J,Kmarne,&
          Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)

   KK(I,J)=K
 END DO
END DO


CALL TABLEAUVIGIE(Nx,Nz,pasx,pasz,H,U,W,KK,HH,UU,WW,KKK,X,Y)

name="finh"
CALL fvigie(name,Nx,Nz,X,Y,HH)

name="finu"
CALL fvigie(name,Nx,Nz,X,Y,UU)

name="finw"
CALL fvigie(name,Nx,Nz,X,Y,WW)

name="fink"
CALL fvigie(name,Nx,Nz,X,Y,KKK)


CALL AFFICHE(Nx,Nz,H,U,W)
```

```
END PROGRAM

SUBROUTINE RESOLUTIONH(Nx,Nz,Aw,Ae,An,As,B,Ap,HH)

INTEGER,INTENT(IN)::Nx,Nz
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(IN)::Ae,Aw,An,As,Ap,B
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::HH

DOUBLE PRECISION,DIMENSION(1:Nx)::Pligne,Qligne
DOUBLE PRECISION,DIMENSION(1:Nz)::Pcol,Qcol
DOUBLE PRECISION::RES
DOUBLE PRECISION,PARAMETER::prec=1E-5
INTEGER::I,J,Ic,Jl



I=0
J=0

DO I=1,Nx

  DO J=1,Nz
      HH(I,J)=0
  END DO

END DO


DO
I=0
J=0
RES=0.

   DO J=1,Nz

      Pligne(1)=Ae(1,J)/Ap(1,J)
      Qligne(1)=(An(1,J)*HH(1,J+1)+As(1,J)*HH(1,J-1)+B(1,J))/Ap(1,J)


         DO I=2,Nx
           Pligne(I)=Ae(I,J)/(Ap(I,J)-Aw(I,J)*Pligne(I-1))
           Qligne(I)=(An(I,J)*HH(I,J+1)+As(I,J)*HH(I,J-1)+B(I,J)+Aw(I,J)*&
                Qligne(I-1))/(Ap(I,J)-Aw(I,J)*Pligne(I-1))
         END DO


      HH(Nx,J)=Qligne(Nx)


         DO I=Nx-1,1,-1

          HH(I,J)=Pligne(I)*HH(I+1,J)+Qligne(I)


       END DO

    END DO


I=0
J=0
```

```
  DO I=1,Nx

    Pcol(1)=An(I,1)/Ap(I,1)
    Qcol(1)=(Ae(I,1)*HH(I+1,1)+Aw(I,1)*HH(I-1,1)+B(I,1))/Ap(I,1)

      DO J=2,Nz

        Pcol(J)=An(I,J)/(Ap(I,J)-As(I,J)*Pcol(J-1))
        Qcol(J)=(Ae(I,J)*HH(I+1,J)+Aw(I,J)*HH(I-1,J)+B(I,J)+As(I,J)*&
              Qcol(J-1))/(Ap(I,J)-As(I,J)*Pcol(J-1))

      END DO

    HH(I,Nz)=Qcol(Nz)

     DO J=Nz-1,1,-1

       HH(I,J)=Pcol(J)*HH(I,J+1)+ Qcol(J)

     END DO

   END DO



  DO J=1,Nz

     DO I=1,Nx

    RES = RES + ABS(Ae(I,J)*HH(I+1,J)+Aw(I,J)*HH(I-1,J)+An(I,J)*HH(I,J+1)+&
              As(I,J)*HH(I,J-1)+B(I,J)-Ap(I,J)*HH(I,J))
     END DO

  END DO

IF (ABS(RES)<prec) EXIT

END DO

END SUBROUTINE


SUBROUTINE AFFICHE(Nx,Nz,HH,UU,WW)

INTEGER::Nx,Nz
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::HH,UU,WW

INTEGER::I,J

OPEN(17,FILE='chargeH.dat')
DO J=1,Nz
WRITE(17,170) (HH(I,J),I=1,Nx)
170 FORMAT(100(f6.3,1x))
END DO
CLOSE(UNIT=17, STATUS='KEEP')

OPEN(18,FILE='U.dat')
DO J=1,Nz
WRITE(18,180) (UU(I,J),I=1,Nx)
180 FORMAT(100(E7.3,1x))
END DO
CLOSE(UNIT=18, STATUS='KEEP')
```

```
OPEN(19,FILE='W.dat')
DO J=1,Nz
WRITE(19,190) (WW(I,J),I=1,Nx)
190 FORMAT(100(E7.3,1x))
END DO
CLOSE(UNIT=19, STATUS='KEEP')

END SUBROUTINE


SUBROUTINE VITESSE(Nx,Nz,Nmarne,Ncalcaire,Nargile,Ngranit,pasx,pasz,&
            Kmarne,Kcalcaire,Kargile,Kgranit,Kmarca,&
            Kcalargile,Kargilgranit,H,UU,WW)

INTEGER,INTENT(IN)::Nx,Nz,Nmarne,Ncalcaire,Nargile,Ngranit
DOUBLE PRECISION,INTENT(IN)::pasx,pasz,Kmarne,Kcalcaire,Kargile,Kgranit,&
                  Kmarca, Kcalargile,Kargilgranit
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(IN)::H
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::UU,WW

INTEGER::I,J
DOUBLE PRECISION::K
I=0
J=0
K=Kmarne


DO J=1,Nz

 CALL CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J,Kmarne,&
      Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)


   UU(1,J)=-K*(H(2,J)-H(1,J))/pasx


   UU(Nx,J)=-K*(H(Nx,J)-H(Nx-1,J))/pasx

   DO I=2,Nx-1

CALL CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J,Kmarne,&
      Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)


      UU(I,J)=-K*(H(I-1,J)-2*H(I,J)+H(I+1,J))/(pasx**2)

   END DO


END DO


K=Kmarne

DO I=1,Nx

   WW(I,1)=-Kmarne*(H(I,2)-H(I,1))/pasz
   WW(I,Nz)=-Kargile*(H(I,Nz)-H(I,Nz-1))/pasz

     DO J=2,Nz-1
```

```
  CALL  CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J,Kmarne,&
        Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)


          WW(I,J)=-K*(H(I,J-1)-2*H(I,J)+H(I,J+1))/(pasz**2)


        END DO
END DO

END SUBROUTINE VITESSE

  SUBROUTINE CALCULK(Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J,Kmarne,&
Kmarca,Kcalcaire,Kcalargile,Kargile,Kargilgranit,Kgranit,K)

  INTEGER,INTENT(IN)::Nx,Nmarne,Ncalcaireg,Ncalcaired,Nargile,I,J
  DOUBLE PRECISION,INTENT(IN)::Kmarne,Kmarca,Kcalcaire,&
                        Kcalargile,Kargile,Kargilgranit,Kgranit
  DOUBLE PRECISION,INTENT(OUT)::K

IF (J<=Nmarne) THEN
K=Kmarne
END IF

IF (J==Nmarne+1) THEN
K=Kmarca
END IF

IF ((J>=Nmarne+2) .AND. (J<=Ncalcaired)) THEN
K=Kcalcaire
END IF

IF ((J>=Ncalcaired+1) .AND. (J<=Ncalcaireg+1)) THEN

    IF ((Nx-I+1)>(J-Ncalcaired)) THEN
        K=Kcalcaire
     END IF

     IF ((Nx-I+1)<(J-Ncalcaired)) THEN
        K=Kargile
     END IF

    IF ((Nx-I+1)==(J-Ncalcaired)) THEN
        K=Kcalargile
     END IF

 END IF


IF ((J>=Ncalcaireg+2) .AND. (J<=Nargile)) THEN
K=Kargile
END IF

IF (J==Nargile+1) THEN
K=Kargilgranit
END IF

IF (J>=Nargile+2) THEN
K=Kgranit
END IF
!print*,J
END SUBROUTINE CALCULK
```

```
SUBROUTINE DEMARRE(Xmax,Xmin,Zgranit,Zmin,Zmarne,Zcalcaired,Zcalcaireg,Zargile&
               ,pasx,pasz,Nx,Nz,Nmarne,Ncalcaired,Ncalcaireg,Nargile,Kmarne&
               ,Kcalcaire,Kargile,Kgranit,Kmarca,Kcalargile,Kargilgranit)


DOUBLE PRECISION,INTENT(IN)::Xmax,Xmin,Zgranit,Zmin,Zmarne,Zcalcaired,&
                         Zcalcaireg,Zargile,Kmarne,Kcalcaire,Kargile,Kgranit
DOUBLE PRECISION,INTENT(OUT)::pasz,Kmarca,Kcalargile,Kargilgranit
DOUBLE PRECISION,INTENT(INOUT)::pasx
INTEGER,INTENT(INOUT)::Nx
INTEGER,INTENT(OUT)::Nz,Ncalcaired,Ncalcaireg,Nmarne,Nargile

INTEGER::kx,kz

kx=INT((Xmax-Xmin)/(pasx*2))
Nx=2*kx+1
pasx=(Xmax-Xmin)/(Nx-1)

pasz=(Zcalcaireg-Zcalcaired)/(Nx-1)
Nz=(Zgranit-Zmin)/pasz+1

Nmarne=Zmarne/pasz
Ncalcaired=Zcalcaired/pasz
Ncalcaireg=Zcalcaireg/pasz
Nargile=Zargile/pasz

Kmarca=(Kmarne+Kcalcaire)/2
Kcalargile=(Kcalcaire+Kargile)/2
Kargilgranit=(Kargile+Kgranit)/2


END SUBROUTINE DEMARRE


SUBROUTINE TABLEAUVIGIE(Nx,Nz,pasx,pasz,H,U,W,KK,HH,UU,WW,KKK,X,Y)

INTEGER,INTENT(IN)::Nx,Nz
DOUBLE PRECISION,INTENT(IN)::pasx,pasz
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(IN)::H,U,W,KK
DOUBLE PRECISION,DIMENSION(1:Nx,1:Nz),INTENT(OUT)::HH,UU,WW,KKK
DOUBLE PRECISION,DIMENSION(1:Nx),INTENT(OUT)::X
DOUBLE PRECISION,DIMENSION(1:Nz),INTENT(OUT)::Y


INTEGER::I,J

DO I=1,Nx
   X(I)=pasx*(I-1)
END DO
DO I=1,Nz
   Y(I)=pasz*(I-1)
END DO

DO I=1,Nx

  DO J=1,Nz
 HH(I,J)=H(I,Nz-J+1)
  END DO
END DO
```

```
DO I=1,Nx
   DO J=1,Nz
  UU(I,J)=U(I,Nz-J+1)
   END DO
END DO

DO I=1,Nx
   DO J=1,Nz
  WW(I,J)=W(I,Nz-J+1)
   END DO
END DO

DO I=1,Nx
   DO J=1,Nz
  KKK(I,J)=KK(I,Nz-J+1)
   END DO
END DO

END SUBROUTINE TABLEAUVIGIE


subroutine  fvigie ( name, n, m, x, y, temp)
 implicit none
 character(len =*), intent (in) :: name
 integer, intent (in) :: n , m
 double precision, intent(in) :: x(n) , y(m) , temp(n,m)
!
! ATTENTION : on suppose ici que les tableaux x et y et surtout la
!             matrice temp sont dimensionnes par rapport a n et m

 integer :: i,j,k,kg, len_trim
 character (len= len_trim (name)+5 ):: nomd, noms

 k = len_trim ( name)
 if ( k == 0) then
   print *,"Erreur vigie: Nom fichier vide"
   return
 endif
 if ( n <=0 .OR. m <=0 ) then
   write (*, "('Erreur vigie nbre points n=',i5,' m=',i5 )") n,m
   return
 endif

! D'abord fichier de parametres
 nomd = name (1:k)//".desc"
 noms = name (1:k)//".sol"
  print *,"Fichier generique pour vigie:",nomd
  open (unit=3,file=nomd, status='unknown')
  write (3,"(a)") "ascii2d"   ! type de fichier 2d en ASCII
  write (3,"(a)") noms                    ! Nom du fichier qui contient les valeurs
  close (unit=3)

!fichier solution pour Vigie

 print *,"Fichier solution:",noms
 open (unit=3,file=noms, status='unknown')
 write (3,"('points ',i4)")n*m           ! Nombre total de valeurs

 do j=1 , m                                       !Coordonnes de chaque noeuds
   do i= 1 , n                                            ! Un (x,y)  par ligne
             write (3,"(2f10.4)")x(i),y(j)
   enddo
```

```
 enddo
              !Table de connectivite
 write (3,"('quadrangles ',i4)") (n-1)*(m-1)   ! Nombre de quadrangles
 do j = 1 , m-1
  do i = 1 , n-1
          ! Pour chaque element numero des noeuds le composant
          ! Dans  le cas d'un maillage stucture  on a
          !          (i,j+1) *--------* (i+1,j+1)
          !                  |        |
          !          (i , j) *--------* (i+1,j)
          !
          ! Sachant que l'on a n points sur x  on reecrit
          !      (i +(j+1)*m) *--------* (i+1+(j+1)*m)
          !                   |        |
          !                   |        |
          !         (i +j*n) *--------* (i+1 +j*n)
          !
    kg = (i-1) +(j-1)*n
    write (3,"(4i5)") kg , kg+1 , kg+1+ n , kg +n
  enddo
 enddo
!    Ecriture solution . Peut etre constituer de plusieurs series
! Format :
!     Mot clef   "scalars T "
!     Puis les valeurs une par ligne
 write (3,"('scalars T ')")
  do j = 1 , m
   do i =1 ,n
    write (3,"(f8.3)")temp(i,j)
   enddo
 enddo
 close(unit=3)

! Si on veut rajouter une autre serie de valeurs il suffit de
! recommencer le bloc ci-dessus en changeant le nom  du scalaire
!   Exemp le:   write (3,"('scalars T1 ')")
!   et mettre les valeurs a la suite


 print *, "Ok vigie creation de ",nomd," et de ",noms
 return
end
```