f2py: Python's interface to the world of number crunching

CodeJam08

Moritz Helias

introduction

- f2py: part of numpy
- wrapping fortran (or C) code into a python module
- frequently needed to make special numerical functions available
- overview:
  - step0: the fortran (or C) source
  - step1: creating an interface
  - step2: adapting the interface
  - step3: compiling the module
  - step4: using the module in python

step 0: the source

fortran77 source file: chgm.f

```
SUBROUTINE CHGM(A,B,X,HG)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
...
RETURN
END
```

- A, B, X: double input valiables
- HG: double output variable

step 1: creating the signature file

```
f2py chgm.f -m add_special -h chgm.pyf
```

creates file: chgm.pyf

```
python module add_special
    interface
        subroutine chgm(a,b,x,hg)
            double precision :: a
            double precision :: b
            double precision :: x
            double precision :: hg
        end subroutine chgm
    end interface
end python module add_special
```

step 2: adapting the signature file

```
python module chgm
    interface
        subroutine chgm(a,b,x,hg)
            double precision, intent(in) :: a
            double precision, intent(in) :: b
            double precision, intent(in) :: x
            double precision, intent(out) :: hg
        end subroutine chgm
    end interface
end python module chgm
```

step 3: building the module

```
f2py -c chgm.pyf -m add_special chgm.f
```

creates

```
add_special.so
```

step 4: using the module in python

python:

```
>>> import add_special
>>> add_special.chgm(1.0, 1.0, 1.0)
2.7182818284590451
```

summary

- f2py: tool to create python modules from fortran / C-code
- numpy.array supported datatype
- uses distutils
- supports many Fortran 77/90/95 compilers (Gnu, Intel, Sun Fortre, SGI MIPSpro, Absoft, NAG, Compaq).
- takes 5 minutes to make fortran code usable in python
- no modification of Fortran code needed
- for details see

  `http://www.scipy.org/F2py`