Taller de Python Científico

Llamando Fortran subroutinas

Departamento de
Ciencias de la Atmósfera y los Océanos
Facultad de Ciencias Exactas y Naturales - Universidad de Buenos Aires

Franco N. Bellomo - fnbellomo@gmail.com

¿Por que llamar a Fortran/C?

```
# file: add_numbers.py
total = 10000000
for i in range(10):
    avg = 0.0
    for  j in range(total):
        avg += j
    avg = avg/total
print("Average is
{0}".format(avg))
```

```
/* file: add_numbers.c */
#include <stdio.h>
int main(int argc, char **argv) {
    int i, j, total;
    double avg;
    total = 10000000;
    for (i = 0; i < 10; i++) {
        avg = 0;
        for (j = 0; j <  total; j++) {
            avg += j;
        }
    avg = avg/total;
    }
    printf("Average is %f\n", avg);
}
```

¿Por que llamar a Fortran/C?

```
$ time python add_numbers.py
Average is 4999999.5

real 0m8.047s
user 0m7.884s
sys 0m0.012s

$ time ./add_numbers.e
Average is 4999999.500000

real 0m0.284s
user 0m0.283s
sys 0m0.001s
```

¿Por que llamar a Fortran/C?

```
$ time python add_numbers.py
Average is 4999999.5

real 0m8.047s
user 0m7.884s
sys 0m0.012s

$ time ./add_numbers.e
Average is 4999999.500000

real 0m0.284s
user 0m0.283s
sys 0m0.001s
```

28x: una simulación de
1 hora es 1 día

¿Y NumPy?

```
# file: add_numbers_fast.py
from numpy import  mean, arange
total = 10000000
a = arange(total)
for  i in range(10):
     avg = mean(a)
print("Average is {0}".format(avg))
```

¿Y NumPy?

```
# file: add_numbers_fast.py
from numpy import  mean, arange
total = 10000000
a = arange(total)
for  i in range(10):
     avg = mean(a)
print("Average is {0}".format(avg))
```

```
$ time python add_numbers_fast.py
Average is 4999999.5

real 0m0.266s
user 0m0.189s
sys 0m0.075s# file
```

Casi igual a C

Opciones para llamar C/Fortran

NumPy es potente pero, solo si mi operaciones son vectoriales.

Opciones para llamar C/Fortran

NumPy es potente pero, solo si mi operaciones son vectoriales.

* Ctypes
  Llamar a funciones de C definiendo los tipos en Python

* Cython
  Autogenerador de código de C desde Python

* F2PY
  Sólo para FORTRAN, fácil de usar, algunos problemas con memoria

F2PY

* Parte del paquete de numpy
* Gran parte de numpy está portado con f2py
* Genera una librería en FORTRAN directamente compatible con Python
* Crea documentación automática bastante buena
* Código creado por máquina, pero mucho mejor que Cython

F2Py

```
! file: fib1.f90
subroutine fib(a,n)
   !
   !     calculate first n fibonacci numbers
   !
   integer n
   real*8 a(n)
   do i=1,n
      if (i.eq.1) then
         a(i) = 0.0d0
      elseif (i.eq.2) then
         a(i) = 1.0d0
      else
         a(i) = a(i-1) + a(i-2)
      endif
   enddo
end subroutine fib
```

F2Py: la forma fácil

```
$ f2py -c fib1.f90 -m fib1
$ python
>>> import fib1
>>> import numpy
>>> a = numpy.zeros(8, dtype=numpy.float64)
>>> fib1(a, 8)
>>> print a
[  0.   1.   1.   2.   3.   5.   8.  13.]
```

F2Py: la forma fácil

```
$ f2py -c fib1.f90 -m fib1
$ python
>>> import fib1
>>> import numpy
>>> a = numpy.zeros(8, dtype=numpy.float64)
>>> fib1(a, 8)
>>> print a
[ 0.  1.  1.  2.  3.  5.  8. 13.]
```

Es fácil en serio!!

F2Py: la forma inteligente

$ f2py fib1.f90 -m fib2 fib2.pyf

```
!    -*- f90 -*-
! Note: the context of this file is case sensitive.

python module fib2 ! in
    interface  ! in :fib1
        subroutine fib(a,n) ! in :fib1:fib1.f90
            real*8 dimension(n) :: a
            integer, optional,check(len(a)>=n),depend(a) :: n=len(a)
        end subroutine fib
    end interface
end python module fib1

! This file was auto-generated with f2py (version:2).
! See http://cens.ioc.ee/projects/f2py2e/
```

F2Py: la forma inteligente

$ cat fib2.pyf

```
!    -*- f90 -*-
! Note: the context of this file is case sensitive.

python module fib2 ! in
    interface  ! in :fib1
        subroutine fib(a,n) ! in :fib1:fib1.f90
            real*8 dimension(n), intent(out), depend(n) :: a
            integer intent(in) :: n
        end subroutine fib
    end interface
end python module fib1

! This file was auto-generated with f2py (version:2).
! See http://cens.ioc.ee/projects/f2py2e/
```

F2Py: la forma inteligente

```
$ f2py -c fib1.f90 fib2.pyf
$ python
>>> import fib2
>>> import numpy
>>> fibonacci = fib2.fib1(8)
>>> print(fibonacci)
[ 0.  1.  1.  2.  3.  5.  8. 13.]
```

F2Py: la forma inteligente y fácil

```
! file: fib3.f90
subroutine fib(a,n)
  !
  !     calculate first n fibonacci numbers
  !
  integer n
  real*8 a(n)
  !f2py intent(in) n
  !f2py intent(out) a
  !f2py depend(n) a
  do i=1,n
     if (i.eq.1) then
        a(i) = 0.0d0
     elseif (i.eq.2) then
        a(i) = 1.0d0
     else
        a(i) = a(i-1) + a(i-2)
     endif
  enddo
end subroutine fib
```

F2Py: la forma inteligente y fácil

```
$ f2py -c fib3.f90 -m fib3
$ python
>>> import numpy
>>> import fib3
>>> a = fib3.fib(8)
>>> print a
[ 0.  1.  1.  2.  3.  5.  8. 13.]
```