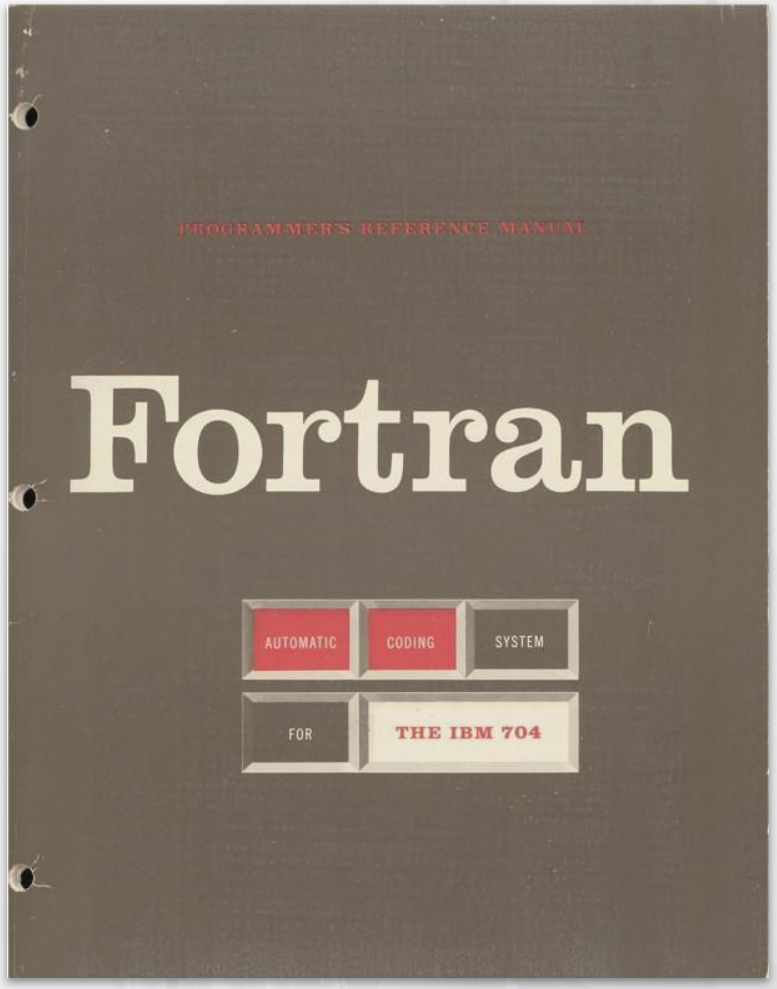
F2PY: Bringing fast code into the future

Melissa Mendonça
Pearu Peterson

FORTRAN/Fortran

"I don’t know what the programming language of the year 2000 will look like, but I know it will be called FORTRAN.”

- Tony Hoare, in 1982



Python

- Interactive, dynamic, “batteries included”
- Large (and growing!) scientific libraries stack
- Community
  - Documentation
  - Online Forums
  - Conferences
  - Locally organized user groups
  - Governance

NumPy/SciPy

NumPy

Base N-dimensional array package

SciPy library

Fundamental library for scientific computing

Matplotlib

Comprehensive 2-D plotting

IP[y]:
IPython

IPython

Enhanced interactive console

SymPy

Symbolic mathematics

pandas

Data structures & analysis

And much more...

F2PY

F2PY is a part of NumPy, and creates extension modules that can be imported in Python.

- Can be used as a command line tool or as a module
- Dependency for many important projects including SciPy
- Can be used to wrap Fortran or C code

F2PY - How does it work?

```
$ f2py -c source.f90 -m fortran
```

```
module fmodule
    implicit none
contains
    subroutine fast_reverse(a, n)

        ! Reverses the first n
        ! elements of the array a

        integer, intent(in) :: n
        real, intent(inout) :: a(:)

        a(1:n) = a(n:1:-1)

    end subroutine fast_reverse
end module fmodule
```

source.f90

```
In [1]: import fortran

In [2]: import numpy as np

In [3]: a = np.array([1, 2, 3], np.float32)

In [4]: fortran.fmodule.fast_reverse(a, 2)

In [5]: a
Out[5]: array([2., 1., 3.], dtype=float32)
```

F2PY - How does it work?

```
$ f2py -c source_alt.f90 -m fortran
```

```
module fmodule
    implicit none
contains
    subroutine fast_reverse(a, n)

        ! Reverses the first n
        ! elements of the array a

        integer, intent(in) :: n
        real, intent(inout) :: a(:)
!f2py   check(len(a)>=n) :: a

        a(1:n) = a(n:1:-1)

    end subroutine fast_reverse
end module fmodule
```

source_alt.f90

```
In [1]: import fortran

In [2]: import numpy as np

In [3]: a = np.array([1, 2, 3], np.float32)

In [4]: fortran.fmodule.fast_reverse(a, 2)
---------------------------------------------------------------------------
error
Traceback (most recent call last)
<ipython-input-9-dd98e8b257c1> in <module>
----> 1 fortran.fmodule.fast_reverse(a, 4)

error: (len(a)>=n) failed for 1st argument
a
```

F2PY - How does it work?

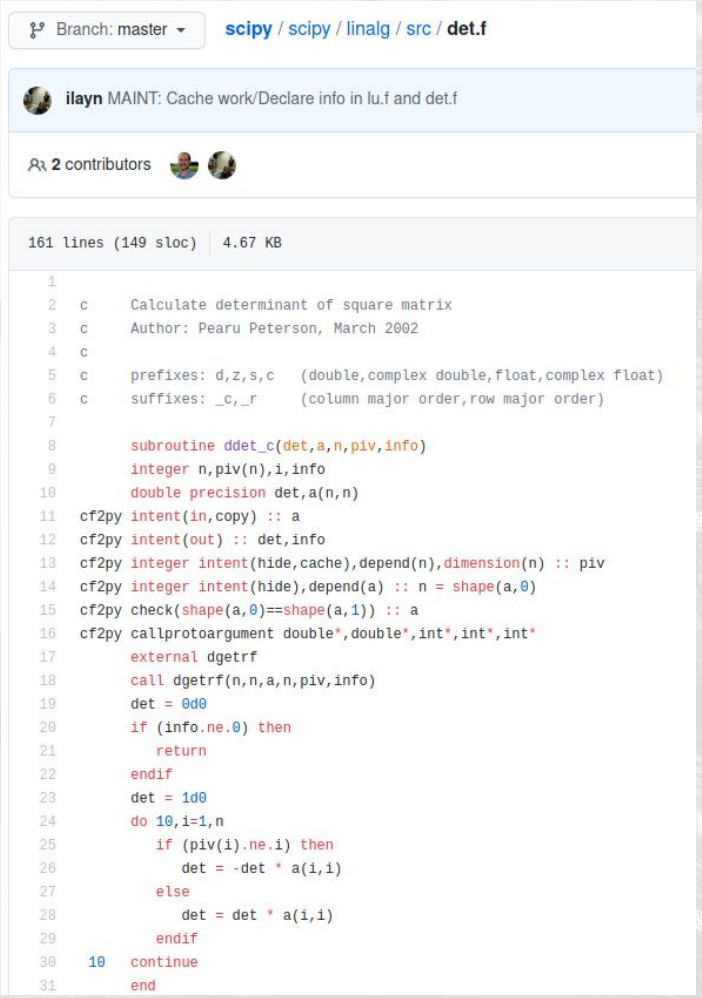
```
$ f2py source.f90 -m fortran -h fortran.pyf
```

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

python module fortran ! in
    interface  ! in :fortran
        module fmodule ! in :fortran:source.f90
            subroutine fast_reverse(a,n) ! in :fortran:source.f90:fmodule
                real dimension(:),intent(inout) :: a
                integer intent(in) :: n
            end subroutine fast_reverse
        end module fmodule
    end interface
end python module fortran

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

fortran.pyf

F2PY - Who uses it

- State Estimation and Analysis in Python (SEAPY)
  - Tools for working with ocean models and data
- wrf-python
  - A collection of diagnostic and interpolation routines for use with output from the Weather Research and Forecasting (WRF-ARW) Model
- Slycot
  - Python wrapper for selected SLICOT routines (systems and control theory)
- pyHyp
  - Used as a preprocessing step in the geometry and mesh-creation process prior to an optimization



Branch: master ▾ scipy / scipy / linalg / src / det.f

ilayn MAINT: Cache work/Declare info in lu.f and det.f

2 contributors

161 lines (149 sloc) | 4.67 KB

```
1
2   c     Calculate determinant of square matrix
3   c     Author: Pearu Peterson, March 2002
4   c
5   c     prefixes: d,z,s,c    (double,complex double,float,complex float)
6   c     suffixes: _c,_r      (column major order,row major order)
7
8         subroutine ddet_c(det,a,n,piv,info)
9         integer n,piv(n),i,info
10        double precision det,a(n,n)
11  cf2py intent(in,copy) :: a
12  cf2py intent(out) :: det,info
13  cf2py integer intent(hide,cache),depend(n),dimension(n) :: piv
14  cf2py integer intent(hide),depend(a) :: n = shape(a,0)
15  cf2py check(shape(a,0)==shape(a,1)) :: a
16  cf2py callprotoargument double*,double*,int*,int*,int*
17        external dgetrf
18        call dgetrf(n,n,a,n,piv,info)
19        det = 0d0
20        if (info.ne.0) then
21           return
22        endif
23        det = 1d0
24        do 10,i=1,n
25           if (piv(i).ne.i) then
26              det = -det * a(i,i)
27           else
28              det = det * a(i,i)
29           endif
30     10 continue
31        end
```

F2PY - Current Features

- Call Fortran 77/90/95 modules and C functions from Python
- Access Fortran 77 COMMON blocks and Fortran 90 module data (also allocatable arrays) from Python
- Call Python functions from Fortran and C (callbacks)
- Generate documentation strings
- Suitable for wrapping C libraries to Python

F2PY - What's missing?

- Support for Fortran 95 and up
- Derived types
- Pointers
- Character/string array support
- Maintenance
- Documentation

Next Steps

- Call for contributions
- Support for Modern Fortran
- Better integration with other tools and compilers
- Support from packages that use f2py as a dependency

Thank You

@melissawm

QUANSIGHT