f2py: volání fortranských procedur z Pythonu

f2py/f2py3 extrahuje z fortranských zdrojových kódů zvolené funkce, podprogramy a moduly, zpracuje jejich rozhraní, zajistí překlad kompatibilním překladačem (gfortran, ifort aj.) a vytvoří pythonský modul. f2py je součástí NumPy, volat lze i z příkazového řádku, pracuje v Linuxu i ve Windows. V Linuxu nečiní obtíže použití OpenMP paralelizace nebo volání externích knihoven. Nelze-li upřesnit popisy rozhraní ve fortranských kódech (zejména atributem intent), mohou se potřebné údaje dodat pomocí souboru s popisy rozhraní (signature file). f2py fortranská rozhraní „pythonizuje": fortranské funkce i podprogramy přetváří na pythonské funkce, výstupní argumenty podprogramů mění na návratové hodnoty funkcí, skaláry udávající velikost polí činí volitelnými a zajišťuje automatické typové konverze argumentů. Vstupní argumenty mohou být pole předpokládaného tvaru, např. a(:). Velikost výstupních polí musí být dána explicitně, např. z(n), z(size(a)), nikoliv z(:), z(*). Složitější datové struktury v argumentech použít nelze.

Ukázka překladu funkce a podprogramu a jejich volání z Pythonu (zdroj a.f90)

```
function f(n)                 subroutine s(a,n,z)          program a
integer,intent(in) :: n       real,intent(in) :: a(n)      real(8) z(2)
real f                        real(8),intent(out) :: z(n)  print *,f(1)
f=n                           z=a                          call s([1.,2.],2,z); print *,z
end function                  end subroutine               end program
```

Překlad rozšiřujícího modulu f90.*.so (Win: *.pyd a f90\.libs*.dll) s volitelným nastavením překladače a jeho voleb:

```
gfortran (Linux, Win):  f2py -c -m f90 a.f90      nebo   python -m numpy.f2py -c -m f90 a.f90
gfortran-9 (Linux):     f2py -c -m f90 --fcompiler=gnu95 --f90exec=gfortran-9 --opt=-Ofast a.f90
ifort (Win):            f2py -c -m f90 --fcompiler=intelvem --opt=/O3 a.f90
ifort (Linux):          f2py -c -m f90 --fcompiler=intelem --opt=-Ofast a.f90
výpis voleb:            f2py; f2py -c --help-fcompiler (pro seznam názvů kompatibilních překladačů)
volání z Pythonu:  import f90; print(dir(f90));print(f90.__doc__);print(f90.s.__doc__); print(f90.f(1));print(f90.s([1,2]))
```

Python volá fortranský podprogram přetvořený na funkci vracející intent(out) argument, argument udávající velikost pole je volitelným, provedou se automatické typové konverze argumentů. Ve Windows je třeba zpřístupnit dll soubor vytvořený gfortranem, např. kopírováním ke skriptu nebo vytvořením linku: md f90; cd f90; mklink /d .libs ..

Ukázka pro modulový podprogram s polem předpokládaného tvaru (zdroj b.f90)

```
module m; contains; subroutine s(a,z); real,intent(in) :: a(:); real,intent(out) :: z(size(a)); z=a; end subroutine; end
program b; use m; real z(2); call s([1.,2.],z); print *,z; end program
```

volání z Pythonu: `import f90; print(f90.m.s([1,2]))`

Ukázka včetně OpenMP paralelizace (zdroj p.f90, funkční v Linuxu, ve Windows jen s ifort)

```
real(8) function f(nmax)                    program p
f=0.                                        real(8) f
!$OMP PARALLEL DO REDUCTION (+:f)           print *,f(2000000000)
do n=1,nmax; f=f+1._8/n; enddo              end program
!$OMP END PARALLEL DO
end function
```

```
gfortran:          f2py -c -m f90 --f90exec=gfortran-9 --f90flags=-fopenmp --opt=-Ofast -lgomp p.f90
ifort:             f2py -c -m f90 --fcompiler=intelem --f90flags=-qopenmp --opt=-Ofast -liomp5 p.f90
volání z Pythonu:  import f90; print(f90.f(2000000000))
```

Ukázka včetně knihoven BLAS/MKL (zdroj z.f90, funkční v Linuxu, ve Windows jen s ifort)

```
subroutine my_matmul(a,b,c,m,n,k)                    program z
real(8),intent(in) :: a(m,k),b(k,n)                  integer,parameter :: n=5000
real(8),intent(out) :: c(m,n)                        real(8),dimension(n,n) :: a,b,c
real(8) alpha,beta                                   a=1.; b=1.
alpha=1.; beta=0.                                    call my_matmul(a,b,c,n,n,n)
call dgemm('n','n',m,n,k,alpha,a,m,b,k,beta,c,m)     print *,n,c(1,1)
end subroutine                                       end program
```

```
gfortran s BLAS:  f2py -c -m f90 --opt=-Ofast -lblas z.f90
ifort s MKL:      f2py -c -m f90 --fcompiler=intelem --opt=-Ofast …
                       -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5 -lpthread -lm -ldl z.f90
```

Lze také použít -lblas s ifortem a volby knihovny MKL s gfortranem. Win: --fcompiler=intelvem --opt="/O3 /Qmkl".

```
Volání z Pythonu:  import f90; import numpy as np; import time; n=5000; a=np.ones((n,n)); b=np.ones((n,n))
                   t1=time.time(); c=np.matmul(a,b); t2=time.time(); print(n,c[0,0],t2-t1) # NumPy
                   t1=time.time(); c=f90.my_matmul(a,b,n,n,n); t2=time.time(); print(n,c[0,0],t2-t1) # MKL
```

Počet vláken řídí proměnná prostředí OMP_NUM_THREADS, při volání knihovny MKL proměnná MKL_NUM_THREADS.