The NAG Library *for Python — Technical Document*

The NAG Python bindings are available from http://www.nag.com/python.asp and are compatible with Python 2.7 and 3.x.

Bindings are available for Windows (64 bit), Linux (64 bit), and Mac (64 bit).

The package contains an embedded NAG Library.

The nag4py bindings strive to provide interfaces that match those in the NAG C Library as closely as possible. Below is a chart that describes the relationship between objects in the NAG C Library and their equivalents in nag4py.

NAG / C Types	Python Example
NAG Enums and NagError Structure— available in the nag4py.util module	*from nag4py.util import noisy_fail, Nag_TRUE*
NAG Structures — available in the associated module for the chapter of interest	*from nag4py.e04 import Nag_E04_Opt* *myoptions = Nag_E04_Opt()* *myoptions.list = Nag_FALSE*
Input Integers/Doubles — input directly into Python	*from nag4py.s import nag_cumul_normal* *nag_cumul_normal(1.1)*
Input/Output Arguments — these must first be set to numpy arrays with the correct data type. TypeError will be raised by the nag4py wrapper whenever an argument is incorrectly typed.	*from numpy import array, empty* *from nag4py.util import nag_int_type* *i_arr = array([1]*10, dtype=nag_int_type)* *d_arr = empty(n)*
Callback Functions — available in the associated chapter	*from nag4py.e04 import NAG_E04UCC_FUN* *c_callback = NAG_E04UCC_FUN(py_callback)*

Other Notes

- **Arguments listed as Input/Output or Output by the NAG C Library will, on output from a nag4py Python call, be modified in place (on the Python function's right-hand side).**
- **A nag_int_type is provided in the nag4py.util module. This type matches the integer type of the NAG C Library.**
- **The nag4py util module provides two convenience functions ('quiet_fail' and 'noisy_fail') to create a NagError instance with printing of messages disabled or enabled respectively.**

Contact us

NAG Ltd—Oxford UK
www.nag.co.uk
+44 1865 511245

NAG Inc—Chicago, USA
www.nag.com
+1 630 971 2337

Nihon NAG—Tokyo, Japan
www.nag-j.co.jp
+81 3 5542 6311

Quick Test of nag4py to show implementation details and check NAG licence

```
>>> from nag4py.a00 import nag_implementation_details, nag_licence_query
>>> nag_implementation_details()
>>> nag_licence_query()
```

The Python 'help' mechanism can be used for more information on chapters/functions

```
>>> import nag4py.c05
>>> help(nag4py.c05)


>>> from nag4py.c05 import c05auc
>>> help(c05auc)
```

Python example calling a NAG optimization function (nag_opt_one_var_deriv/e04bbc)

```
"""
Example for nag_opt_one_var_deriv (e04bbc) using nag4py.
Finds the minimum of sin(x)/x in the interval [3.5,5].
NAG Copyright 2014.
"""

def py_obj_func(xc, fc, gc, comm):
      from math import sin, cos
      fc[0] = sin(xc) / xc
      gc[0] = (cos(xc) - fc[0]) / xc


from nag4py.e04 import nag_opt_one_var_deriv, NAG_E04BBC_FUN
from nag4py.util import Nag_Comm, noisy_fail
from numpy import array, empty

# Create the C callback function:
c_func = NAG_E04BBC_FUN(py_obj_func)

# Input data:
e1 = 0.0
e2 = 0.0
max_fun = 30

# Initial bounding interval, will be modified on exit:
a = array(3.5)
b = array(5.0)
# Output data, initialized for the NAG call:
x = empty(1)
f = empty(1)
g = empty(1)

comm = Nag_Comm()
fail = noisy_fail()

print("nag_opt_one_var_deriv Python Example Script Results.")
print("Finding the minimum of sin(x) / x.")
print("Starting interval is [%.1f, %.1f]." % (a, b))

nag_opt_one_var_deriv(c_func, e1, e2, a, b, max_fun, x, f, g, comm, fail)

print("Solution is within the interval [%.12f, %.12f]." % (a, b))
print("Solution is %.12f." % x)
print("At the solution the function value is %.12f" % f)
print(" and the gradient is %.11e." % g)
print("There were %i function evaluations required." % comm.nf)
```

- **Note the Python callback function has the same signature as the C callback.**
- **Data from callback functions has to be returned in the C-like by-reference fashion, with scalar data having to appear as an array of length 1 to facilitate this.**
- **Set the callback to the Python function**
- **Print any error messages**
- **Call the NAG function**