

callback

September 30, 2024

This file is part of CasADi.

CasADi -- A symbolic framework for dynamic optimization.
Copyright (C) 2010–2023 Joel Andersson, Joris Gillis, Moritz Diehl,
KU Leuven. All rights reserved.
Copyright (C) 2011–2014 Greg Horn

CasADi is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 3 of the License, or (at your option) any later version.

CasADi is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with CasADi; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

```
[1]: from casadi import *
```

1 A simple case of Callback

Callback allows the user to create functions that can be embedded into CasADi expressions. The user creates a class that inherits from this class and implements a subset of the virtual methods. Although Callback itself is implemented in C++, the virtual methods can be implemented in Python or MATLAB thanks to cross-language polymorphism as supported by the SWIG framework.

```
[2]: class Fac(Callback):  
    def __init__(self, name, opts={}):  
        Callback.__init__(self)  
        self.construct(name, opts)  
  
    def get_n_in(self): return 1  
    def get_n_out(self): return 1
```

```
def eval(self, arg):
    x = arg[0]
    y = 1
    for i in range(int(x)):
        y*=(i+1)
    return [y]
```

```
[3]: fac = Fac('fac')
```

```
[4]: # Evaluate numerically
y = fac(4)
```

```
[5]: print("4! = ", y)
```

4! = 24

2 Using the function in a graph

```
[6]: x = MX.sym("x")
y = fac(x)
```

```
[7]: f = Function('f', [x],[y])
```

```
[8]: y = f(5)
```

```
[9]: print("5! = ", y)
```

5! = 120