

# SXFunction\_constr\_py

May 17, 2023

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 Function constructors

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

```
[2]: x = SX.sym("x")      # A scalar (1-by-1 matrix) symbolic primitive  
     y = SX.sym("y",2)    # A vector (n-by-1 matrix) symbolic primitive  
     z = SX.sym("z",2,3)  # An n-by-m matrix symbolic primitive
```

```
[3]: ins = [x,y] # function inputs  
     outs = [x,y,vertcat(x,y),y*x,0]
```

```
[4]: print(outs)
```

```
[SX(x), SX([y_0, y_1]), SX([x, y_0, y_1]), SX([(y_0*x), (y_1*x)]), 0]
```

```
[5]: f = Function("f", ins, outs)
```

f now has two inputs and a 4 outputs:

```
[6]: print(f.n_in())  
      print(f.n_out())
```

2

5

The outputs has the following string representation. Note how all elements of out have been converted to SX by automatic typecasting functionality

```
[7]: f_out = f(*f.sx_in())  
      for i in range(3):  
          print(f_out[i])
```

x

[y\_0, y\_1]

[x, y\_0, y\_1]