expand

April 18, 2023

1 expand

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

We construct a simple MX expression

[2]:
```
x = MX.sym("x",2,2)
y = MX.sym("y",2,1)
```

[3]:
```
z = mtimes(x,y)
```

Let's construct an MXfunction

[4]:
```
f = Function("f", [x,y],[z])
```

We expand the MX expression into an SX expression

[5]:
```
fSX = f.expand('fSX')
```

[6]:
```
print("Expanded expression = ", fSX.str(True))
```

```
Expanded expression =  fSX:(i0[2x2],i1[2])->(o0[2]) SXFunction
Algorithm:
@0 = input[0][0];
@1 = input[1][0];
@0 = (@0*@1);
@2 = input[0][2];
@3 = input[1][1];
@2 = (@2*@3);
@0 = (@0+@2);
output[0][0] = @0;
@0 = input[0][1];
@0 = (@0*@1);
@1 = input[0][3];
@1 = (@1*@3);
@0 = (@0+@1);
output[0][1] = @0;
```

2 Limitations

Not all MX graphs can be expanded. Here is an example of a situation where it will not work.

[7]:
```
linear_solver = Linsol("linear_solver", "csparse", x.sparsity())
g = linear_solver.solve(x, y)
G = Function("G", [x,y], [g])
```

This function cannot be expanded.

[8]:
```
try:
  G.expand('G_sx')
except Exception as e:
  print(e)
```

```
Error in Function::call for 'G' [MXFunction] at
…/casadi/core/function.cpp:339:
…/casadi/core/linsol_internal.cpp:65: eval_sx not defined for CsparseInterface
```