

NLPImplicitSolver

July 23, 2025

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 NLPImplicitSolver

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

We will investigate the working of rootfinder with the help of the parametrically excited Duffing equation.

$\ddot{u} + \dot{u} - \epsilon(2\mu\dot{u} + \alpha u^3 + 2ku \cos(\Omega t))$ with $\Omega = 2 + \epsilon\sigma$. \

The first order solution is $u(t) = a \cos(\frac{1}{2}\Omega t - \frac{1}{2}\gamma)$ with the modulation equations: \ $\frac{da}{d\epsilon t} = -[\mu a + \frac{1}{2}ka \sin \gamma]$ \ $a \frac{d\gamma}{d\epsilon t} = -[-\sigma a + \frac{3}{4}\alpha a^3 + ka \cos \gamma]$ \

We seek the stationair solution to these modulation equations.

Parameters

```
[2]: eps = SX.sym("eps")
      mu = SX.sym("mu")
      alpha = SX.sym("alpha")
      k = SX.sym("k")
      sigma = SX.sym("sigma")
      params = [eps,mu,alpha,k,sigma]
```

Variables

```
[3]: a = SX.sym("a")
      gamma = SX.sym("gamma")
```

Equations

```
[4]: res0 = mu*a+1.0/2*k*a*sin(gamma)
      res1 = -sigma * a + 3.0/4*alpha*a**3+k*a*cos(gamma)
```

Numerical values

```
[5]: sigma_ = 0.1
      alpha_ = 0.1
      k_ = 0.2
      params_ = [0.1,0.1,alpha_,k_,sigma_]
```

We create a NLPImplicitSolver instance

```
[6]: f=Function("f", [vertcat(a, gamma), vertcat(*params)], [vertcat(res0, res1)])
      opts = {}
      opts["nlpsol"] = "ipopt"
      opts["nlpsol_options"] = {"ipopt.tol":1e-14}
      s=rootfinder("s", "nlpsol", f, opts)
```

Initialize $[a,\gamma]$ with a guess and solve

```
[7]: x_ = s([1,-1], params_)
      print("Solution = ", x_)
```

```
*****
This program contains Ipopt, a library for large-scale nonlinear optimization.
Ipopt is released as open source code under the Eclipse Public License (EPL).
For more information visit https://github.com/coin-or/Ipopt
*****
```

This is Ipopt version 3.14.11, running with linear solver MUMPS 5.4.1.

```
Number of nonzeros in equality constraint Jacobian...:      4
Number of nonzeros in inequality constraint Jacobian.:      0
Number of nonzeros in Lagrangian Hessian...:           3
```

```

Total number of variables...:      2
      variables with only lower bounds:      0
      variables with lower and upper bounds:  0
      variables with only upper bounds:      0
Total number of equality constraints...:      2
Total number of inequality constraints...:      0
      inequality constraints with only lower bounds:      0
      inequality constraints with lower and upper bounds:  0
      inequality constraints with only upper bounds:      0

```

iter	objective	inf_pr	inf_du	lg(mu)	d	lg(rg)	alpha_du	alpha_pr	ls
0	0.0000000e+00	8.31e-02	0.00e+00	-1.0	0.00e+00	-	0.00e+00	0.00e+00h	0
1	0.0000000e+00	1.23e-02	0.00e+00	-2.5	2.40e-01	-	1.00e+00	1.00e+00h	1
2	0.0000000e+00	2.02e-03	0.00e+00	-3.8	2.00e-01	-	1.00e+00	1.00e+00h	1
3	0.0000000e+00	1.28e-03	0.00e+00	-3.8	1.15e-01	-	1.00e+00	1.00e+00h	1
4	0.0000000e+00	4.42e-05	0.00e+00	-5.7	2.96e-02	-	1.00e+00	1.00e+00h	1
5	0.0000000e+00	2.30e-05	0.00e+00	-5.7	1.68e-02	-	1.00e+00	1.00e+00h	1
6	0.0000000e+00	5.32e-06	0.00e+00	-5.7	8.03e-03	-	1.00e+00	1.00e+00h	1
7	0.0000000e+00	1.34e-06	0.00e+00	-8.6	3.98e-03	-	1.00e+00	1.00e+00h	1
8	0.0000000e+00	3.35e-07	0.00e+00	-8.6	1.98e-03	-	1.00e+00	1.00e+00h	1
9	0.0000000e+00	8.38e-08	0.00e+00	-8.6	9.87e-04	-	1.00e+00	1.00e+00h	1
iter	objective	inf_pr	inf_du	lg(mu)	d	lg(rg)	alpha_du	alpha_pr	ls
10	0.0000000e+00	2.10e-08	0.00e+00	-8.6	4.93e-04	-	1.00e+00	1.00e+00h	1
11	0.0000000e+00	5.24e-09	0.00e+00	-12.9	2.46e-04	-	1.00e+00	1.00e+00h	1
12	0.0000000e+00	1.31e-09	0.00e+00	-12.9	1.23e-04	-	1.00e+00	1.00e+00h	1
13	0.0000000e+00	3.28e-10	0.00e+00	-12.9	6.15e-05	-	1.00e+00	1.00e+00h	1
14	0.0000000e+00	8.19e-11	0.00e+00	-12.9	3.08e-05	-	1.00e+00	1.00e+00h	1
15	0.0000000e+00	2.05e-11	0.00e+00	-12.9	1.54e-05	-	1.00e+00	1.00e+00h	1
16	0.0000000e+00	5.12e-12	0.00e+00	-12.9	7.69e-06	-	1.00e+00	1.00e+00h	1
17	0.0000000e+00	1.28e-12	0.00e+00	-12.9	3.84e-06	-	1.00e+00	1.00e+00h	1
18	0.0000000e+00	3.20e-13	0.00e+00	-12.9	1.92e-06	-	1.00e+00	1.00e+00h	1
19	0.0000000e+00	8.00e-14	0.00e+00	-15.0	9.61e-07	-	1.00e+00	1.00e+00h	1
iter	objective	inf_pr	inf_du	lg(mu)	d	lg(rg)	alpha_du	alpha_pr	ls
20	0.0000000e+00	2.00e-14	0.00e+00	-15.0	4.80e-07	-	1.00e+00	1.00e+00h	1
21	0.0000000e+00	5.04e-15	0.00e+00	-15.0	2.40e-07	-	1.00e+00	1.00e+00h	1

Number of Iterations...: 21

	(scaled)	(unscaled)
Objective...:	0.0000000000000000e+00	0.0000000000000000e+00
Dual infeasibility...:	0.0000000000000000e+00	0.0000000000000000e+00
Constraint violation...:	5.0404780683255265e-15	5.0404780683255265e-15
Variable bound violation:	0.0000000000000000e+00	0.0000000000000000e+00
Complementarity...:	0.0000000000000000e+00	0.0000000000000000e+00
Overall NLP error...:	5.0404780683255265e-15	5.0404780683255265e-15

```

Number of objective function evaluations      = 22
Number of objective gradient evaluations     = 22
Number of equality constraint evaluations     = 22
Number of inequality constraint evaluations   = 0
Number of equality constraint Jacobian evaluations = 22
Number of inequality constraint Jacobian evaluations = 0
Number of Lagrangian Hessian evaluations    = 21
Total seconds in IPOPT                      = 0.012

```

EXIT: Optimal Solution Found.

nlpsol	:	t_proc	(avg)	t_wall	(avg)	n_eval
nlp_f		159.00us	(7.23us)	41.52us	(1.89us)	22
nlp_g		454.00us	(20.64us)	109.70us	(4.99us)	22
nlp_grad_f		199.00us	(8.65us)	49.53us	(2.15us)	23
nlp_hess_l		451.00us	(21.48us)	116.09us	(5.53us)	21
nlp_jac_g		467.00us	(20.30us)	118.65us	(5.16us)	23
total		50.04ms	(50.04ms)	12.82ms	(12.82ms)	1

Solution = [1.1547, -1.5708]

Compare with the analytic solution:

```
[8]: x = [sqrt(4.0/3*sigma_/alpha_), -0.5*pi]
      print("Reference solution = ", x)
```

Reference solution = [1.1547005383792515, -1.5707963267948966]

We show that the residual is indeed (close to) zero

```
[9]: residual = f(x_, params_)
      print("residual = ", residual)
```

residual = [2.498e-15, 5.04048e-15]

```
[10]: for i in range(1):
       assert(abs(x_[i]-x[i])<1e-6)
```

Solver statistics

```
[11]: print(s.stats())
```

```

{'n_call_jac_g_x': 0, 'nlpsol': {'iter_count': 21, 'iterations': {'alpha_du':
[0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0], 'alpha_pr': [0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0],
'd_norm': [0.0, 0.23960592290193847, 0.2002776648351239, 0.11462838134468271,
0.029608423804759445, 0.016763794276380726, 0.008032525850076699,
0.003977508980797005, 0.0019782511377219327, 0.000986563314767321,
0.0004926468651346938, 0.0002461654667430133, 0.00012304333360478557,
6.151182779221677e-05, 3.075345789240383e-05, 1.5376112167388994e-05,
7.687910912446485e-06, 3.8438997812749356e-06, 1.92194207223915e-06,

```

```

9.61008195593693e-07, 4.804365673649445e-07, 2.404509433067261e-07], 'inf_du':
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'inf_pr': [0.08306046117362796,
0.012290452768353485, 0.0020204300142100742, 0.0012794397549934066,
4.421852776412272e-05, 2.300330418598907e-05, 5.321988750748349e-06,
1.3390883536304509e-06, 3.3506843297604477e-07, 8.381216439051482e-08,
2.09588221506264e-08, 5.24043895038358e-09, 1.310202083785794e-09,
3.2756211715529815e-10, 8.189201858746866e-11, 2.047313311593076e-11,
5.118331222212175e-12, 1.2796006133619685e-12, 3.1984623222892154e-13,
8.003960858313451e-14, 1.9956499544108217e-14, 5.0404780683255265e-15], 'mu':
[0.1, 0.002828427124746191, 0.00015042412372345582, 0.00015042412372345582,
1.8449144625279508e-06, 1.8449144625279508e-06, 1.8449144625279508e-06,
2.5059035596800618e-09, 2.5059035596800618e-09, 2.5059035596800618e-09,
2.5059035596800618e-09, 1.2544302826334687e-13, 1.2544302826334687e-13,
1.2544302826334687e-13, 1.2544302826334687e-13, 1.2544302826334687e-13,
1.2544302826334687e-13, 1.2544302826334687e-13, 1.2544302826334687e-13,
9.090909090909092e-16, 9.090909090909092e-16, 9.090909090909092e-16], 'obj':
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'regularization_size': [0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0]}, 'n_call_callback_fun': 0, 'n_call_nlp_f': 22, 'n_call_nlp_g': 22,
'n_call_nlp_grad': 0, 'n_call_nlp_grad_f': 23, 'n_call_nlp_hess_l': 21,
'n_call_nlp_jac_g': 23, 'n_call_total': 1, 'return_status': 'Solve_Succeeded',
'success': True, 't_proc_callback_fun': 0.0, 't_proc_nlp_f':
0.00015900000000000002, 't_proc_nlp_g': 0.000454, 't_proc_nlp_grad': 0.0,
't_proc_nlp_grad_f': 0.00019900000000000004, 't_proc_nlp_hess_l':
0.00045100000000000007, 't_proc_nlp_jac_g': 0.000467, 't_proc_total': 0.050037,
't_wall_callback_fun': 0.0, 't_wall_nlp_f': 4.151999999999999e-05,
't_wall_nlp_g': 0.000109704, 't_wall_nlp_grad': 0.0, 't_wall_nlp_grad_f':
4.9534e-05, 't_wall_nlp_hess_l': 0.00011608899999999998, 't_wall_nlp_jac_g':
0.00011865399999999999, 't_wall_total': 0.012821663, 'unified_return_status':
'SOLVER_RET_SUCCESS'}, 'success': True, 't_proc_jac_g_x': 0.0, 't_wall_jac_g_x':
0.0, 'unified_return_status': 'SOLVER_RET_SUCCESS'}

```