4. CasADi nlpsol

Concrete nlpsol example

$$\underset{x,y,z}{minimize} \quad x^2+100z^2$$
$$s.t. \quad z+(1-x)^2-y=0$$

```
x = MX.sym('x');
y = MX.sym('y');
z = MX.sym('z');
v = [x;y;z];
f = x^2 + 100*z^2;
g = z + (1-x)^2 - y;
nlp = struct('x', v, 'f', f, 'g', g);

solver = nlpsol('solver', 'ipopt', nlp);

res = solver('x0' , [2.5 3.0 0.75],... % solution guess
             'lbg',    0,...           % lower bound on g
             'ubg',    0);             % upper bound on g

x_opt = full(res.x)
```

Concrete nlpsol example

$$\underset{x}{\text{minimize}} \quad f(x,p)$$

$$s.t. \quad x_{LB} \le x \le x_{UB}$$

$$g_{LB} \le g(x,p) \le g_{UB}$$

```
res = solver('x0' , [2.5 3.0 0.75],...
             'lbg',    0,...
             'ubg',    0);
```

Full name	Short	Description
NLPSOL_X0	x0	Decision variables, initial guess (nx x 1)
NLPSOL_P	p	Value of fixed parameters (np x 1)
NLPSOL_LBX	lbx	Decision variables lower bound (nx x 1), default -inf.
NLPSOL_UBX	ubx	Decision variables upper bound (nx x 1), default +inf.
NLPSOL_LBG	lbg	Constraints lower bound (ng x 1), default -inf.
NLPSOL_UBG	ubg	Constraints upper bound (ng x 1), default +inf.
NLPSOL_LAM_X0	lam_x0	Lagrange multipliers for bounds on X, initial guess (nx x 1)
NLPSOL_LAM_G0	lam_g0	Lagrange multipliers for bounds on G, initial guess (ng x 1)

```
doc nlpsol
```

ya coda

Concrete nlpsol example

$$\underset{x}{\text{minimize}} \quad f(x,p)$$

$$s.t. \quad x_{LB} \le x \le x_{UB}$$
$$g_{LB} \le g(x,p) \le g_{UB}$$

```
x_opt = full(res.x)
```

```
>Output scheme: casadi::NlpsolOutput (NLPSOL_NUM_OUT = 6)
```

Full name	Short	Description
NLPSOL_X	x	Decision variables at the optimal solution (nx x 1)
NLPSOL_F	f	Cost function value at the optimal solution (1 x 1)
NLPSOL_G	g	Constraints function at the optimal solution (ng x 1)
NLPSOL_LAM_X	lam_x	Lagrange multipliers for bounds on X at the solution (nx x 1)
NLPSOL_LAM_G	lam_g	Lagrange multipliers for bounds on G at the solution (ng x 1)
NLPSOL_LAM_P	lam_p	Lagrange multipliers for bounds on P at the solution (np x 1)

```
doc nlpsol
```

Concrete nlpsol example

```
solver = nlpsol('solver', 'ipopt', nlp);
```

```
List of plugins
===============


- AmplInterface
- blocksqp
- bonmin
- ipopt
- knitro
- snopt
- worhp
- scpgen
- sqpmethod
```

```
doc nlpsol
```

Concrete nlpsol example

```
options = struct;
options.expand = true;
options.ipopt.print_level = 0;

solver = nlpsol('solver', 'ipopt', nlp, options);
```

doc nlpsol

Search web for ipopt options

Ipopt output

How good are my optimality conditions met?

How big was my step?

```
iter    objective    inf_pr   inf_du lg(mu)  ||d||  lg(rg) alpha_du alpha_pr  ls
   0  6.2500000e+01 0.00e+00 9.00e+01  -1.0 0.00e+00    -  0.00e+00 0.00e+00   0
   1  1.8457621e+01 1.48e-02 4.10e+01  -1.0 4.10e-01   2.0 1.00e+00 1.00e+00f  1
   2  7.8031530e+00 3.84e-03 8.76e+00  -1.0 2.63e-01   1.5 1.00e+00 1.00e+00f  1
```

Log10 of barrier parameter

How much are constraints violated?

How much regularisation was needed
to force a positive definite hessian?

How many steps did line-search do?

Generic syntax

Structure with SX/MX expressions

- solver(label,plugin_name,problem_struct,options_struct)

Dynamically linked in

e.g. casadi_nlpsol_**ipopt**.dll

Only for debugging

Structure with integers/doubles/strings

rootfinder
integrator
qpsol
nlpsol

Generic syntax

- solver(label,plugin_name,problem_struct,options_struct)
- Creates a CasADi Function
  - Can be evaluated numerically, symbolically
  - Can be differentiated (not yet conic/qpsol)

```
x = MX.sym('x');
```

$$\underset{x}{minimize} \quad x^2$$

```
solver = nlpsol('solver', 'ipopt', struct('x',x,'f',x^2));
```

$$\frac{dx}{dt} = x^2$$

```
integr = integrator('integr', 'cvodes', struct('x',x,'ode',x^2));
```

$$\sin(x) - 0.5 = 0$$

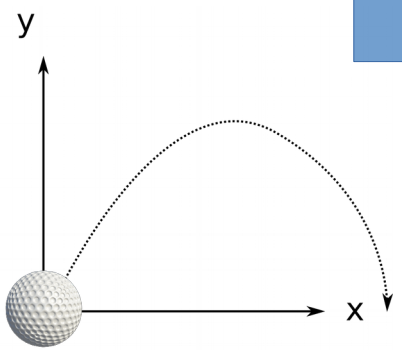
```
rf     = rootfinder('rf', 'newton', struct('x',x,'g',sin(x)-0.5));
```

Day 1 synthesis exercise: golf

rootfinder

integrator

nlpsol



4. CasADi nlpsol