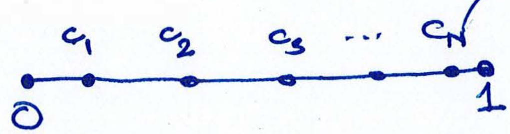
```
% An implementation of direct collocation
% Joris Gillis, 2018
import casadi.*

% Degree of interpolating polynomial
d = 3;

% Get collocation points
tau = collocation_points(d, 'legendre');

% Collocation linear maps
[C,D,B] = collocation_coeff(tau);

% Time horizon
T = 10;

% Declare model variables
x1 = SX.sym('x1');
x2 = SX.sym('x2');
x = [x1; x2];
u = SX.sym('u');
```

mapped from $(-1, 1)$

Gauss points over $(0,1)$

①



② Interpolation through 0 and the N quadrature points; i.e.; passing through the N+1 points $[x\_0,\ x\_c_1, \ldots, x\_c_N] =: Z$ i.e. polynomial of degree $d = N$

③ $\Pi(\tau)$ is the interpolating polynomial. $\tau \in (0,1)$

Defn of C

④ $\Pi'([c_1, c_2, \ldots c_N]) = Z * C.$

for $t \in (0,h)$, $\hat{\Pi}(t) = \Pi(t/h)$.

$\hat{\Pi}'(t) = \Pi'(t/h) \cdot \frac{1}{h} = \frac{1}{h} Z * C.$

⑤ $\Pi(1) = Z * B$ – defn of B

⑥ B = integration wts over $(0,1)$ – mapped by jacobian $1/2$ from $(-1,1)$.

```
% Model equations
xdot = [(1-x2^2)*x1 - x2 + u; x1];

% Objective term
L = x1^2 + x2^2 + u^2;

% Continuous time dynamics
f = Function('f', {x, u}, {xdot, L});

% Control discretization
N = 20; % number of control intervals
h = T/N;

% Start with an empty NLP

opti = Opti();
J = 0;

% "Lift" initial conditions
Xk = opti.variable(2);
opti.subject_to(Xk==[0; 1]);
opti.set_initial(Xk, [0; 1]);

% Collect all states/controls
Xs = {Xk};
Us = {};

% Formulate the NLP
for k=0:N-1
    % New NLP variable for the control
    Uk = opti.variable();
    Us{end+1} = Uk;
    opti.subject_to(-1<=Uk<=1);
    opti.set_initial(Uk, 0);

    % Decision variables for helper states at each collocation point
    Xc = opti.variable(2, d);
    opti.subject_to(-0.25 <= Xc(1,:));
    opti.set_initial(Xc, repmat([0;0],1,d));

    % Evaluate ODE right-hand-side at all helper states
    [ode, quad] = f(Xc, Uk);

    % Add contribution to quadrature function
    J = J + quad*B*h;

    % Get interpolating points of collocation polynomial
    Z = [Xk Xc];
```

$\dot{x}_1 = (1-x_2^2)x_1 - x_2 + u.$

$\dot{x}_2 = x_1.$

minimize $\int_0^T (x_1^2+x_2^2+u^2)\,dt$

s.t. above dynamics

$-1 \le u \le 1$

$x_2 \ge -0.25$

$x(0) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

initial guess.

do methodsview(opti)
or methodsview(casadi.Opti)
to see the methods such as set_initial.

doc casadi.Opti/set_initial
to get help.

```
    % Get slope of interpolating polynomial (normalized)
    Pidot = Z*C;
    % Match with ODE right-hand-side
    opti.subject_to(Pidot == h*ode);

    % State at end of collocation interval
    Xk_end = Z*D;

    % New decision variable for state at end of interval
    Xk = opti.variable(2);
    Xs{end+1} = Xk;
    opti.subject_to(-0.25 <= Xk(1));
    opti.set_initial(Xk, [0;0]);

    % Continuity constraints
    opti.subject_to(Xk_end==Xk)
end

Xs = [Xs{:}];
Us = [Us{:}];

opti.minimize(J);

opti.solver('ipopt');

sol = opti.solve();

x_opt = sol.value(Xs);
u_opt = sol.value(Us);

% Plot the solution
tgrid = linspace(0, T, N+1);
clf;
hold on
plot(tgrid, x_opt(1,:), '--')
plot(tgrid, x_opt(2,:), '-')
stairs(tgrid, [u_opt nan], '-.')
xlabel('t')
legend('x1','x2','u')
```