Collocation methods for second order systems

Lluís Ros, May 2022

Institut de Robòtica i Informàtica Industrial (CSIC-UPC)

https://www.iri.upc.edu/people/ros/

This MATLAB livescript contains the symbolic derivations of the trapezoidal and Hermite-Simpson methods explained in the paper "**Collocation methods for second order systems**" by Siro Moreno-Martín, Lluís Ros, and Enric Celaya, presented in Robotics: Science and Systems XVIII (New York, 2022).

You can download the paper and the livescript from:

http://www.roboticsproceedings.org/rss18/p038.html

https://www.iri.upc.edu/people/ros/Separates/rss2022_derivations.mlx

See also the RSS'22 presentation and poster:

https://youtu.be/yxnbhNxOLLk?t=3039

https://bit.ly/3Pmqc5k

Warning: unfortunately, the published paper from the previous link has a typo that must be attributed to ourselves. The first h in Eq. (24b) should be a 1. The same happens in the copy of this equation in Table I.

In the derivations that follow, we include both the classical and modified versions of the methods (for first and second order systems respectively). We closely follow the paper notation and explanations except for the subindices k and $k+1$, which are replaced by 0 and 1 in this livescript. This is because Matlab cannot work symbolically with a subindex like $k+1$ in the 2022a release. Equation numbers cannot be managed either, but we have found a workaround to label the main equations as they appear in the paper.

We recall that, as in the paper, the terminology "first order methods" refers to collocation methods that guarantee

$$\dot{x} = f(x, u, t)$$

at the collocation points, whereas "second order methods" are those that guarantee

$$\ddot{q} = g(q, \dot{q}, u, t)$$

at such points.

Table of Contents

```
% Clear all vars, close all figs, and clear command window
clearvars
close all
clc

% Start stopwatch
tic

% Symbolic variables
syms h                  % Time step
syms x_0 x_1 x_c        % Initial, final, and midpoint states
syms x_dot_0            % Initial state derivative
syms x_dot_c            % Midpoint state derivative
syms x_dot_1            % Final state derivative
syms tau                % Modified time parameter tau
syms f_0 f_1 f_c        % 1st order dynamics at x0, x1, and xc
syms x(tau)             % Polynomial for x(tau)
syms q(tau) q_dot(tau)  % Polynomials for q(tau) and q_dot(tau)
syms q_0 q_1 q_c        % Configurations at t_0, t_1, t_c
syms v_0 v_1 v_c        % Velocities at t_0, t_1, t_c
syms q_ddot_0 q_ddot_1  % Accelerations at t_0 and t_1
syms g_0 g_1 g_c        % 2nd order dynamics at t_0, t_1, t_c
syms q_ddot_c           % Midpoint acceleration ddq/ddt
```

First order methods

The trapezoidal method

In trapezoidal collocation, the state trajectories are approximated by quadratic polynomials. If, for each interval $[t_0, t_1]$, we define $\tau = t - t_0$, we can write the polynomial approximation for a component x of the state in this interval, and its temporal derivative, as

$$x(\tau) = a + b\tau + c\tau^2,$$
$$\dot{x}(\tau) = b + 2c\tau,$$

where a, b, and c are real coefficients. To facilitate the application of collocation constraints, however, we will rewrite $x(\tau)$ using the three parameters

$$x_0 = x(0),$$
$$\dot{x}_0 = \dot{x}(0),$$
$$\dot{x}_1 = \dot{x}(h),$$

where $h = t_1 - t_0$. Evaluating the right-hand sides of the latter equations using the previous expressions of $x(\tau)$ and $\dot{x}(\tau)$ we obtain

$$\begin{bmatrix} x_0 \\ \dot{x}_0 \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 2h \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix},$$

so solving for a, b, c we have

```
N = [x_0; x_dot_0; x_dot_1];
A = [1 0  0; 0 1 0; 0 1 2*h];
soln = A\N;
a = soln(1); b = soln(2); c = soln(3);
display(a); display(b); display(c);
```

a = x_0

b = $\dot{x}_0$

c = $-\dfrac{\dot{x}_0 - \dot{x}_1}{2\,h}$

Substituting these expressions in $x(\tau) = a + b\tau + c\tau^2$ we have

```
x(tau) = symfun(a + b * tau + c * tau^2,tau)
```

x(tau) = $x_0 + \tau\,\dot{x}_0 - \dfrac{\tau^2\,(\dot{x}_0 - \dot{x}_1)}{2\,h}$

This expression is known as the **interpolation polynomial**, as it allows us to estimate the intermediate states for $t \in [t_0, t_1]$, once the NLP problem has been solved.

Following (Hairer 2002), we determine the three parameters of this polynomial by enforcing the initial value constraint $x(0) = x_0$ and two collocation constraints of the form

$$\dot{x}(t_i) = f(x(t_i), u(t_i), t_i).$$

From the expression of $x(\tau)$ we see that $x(0) = x_0$ by construction. As for the collocation constraints, the trapezoidal method imposes them at the knot points t_0 and t_1 so it must be

$$\begin{aligned}\dot{x}_0 &= f_0,\\ \dot{x}_1 &= f_1,\end{aligned}$$

where f_k is a shorthand of $f(x_k, u_k, t_k)$. The value x_1, then, is obtained by evaluating $x(\tau)$ for $\tau = h$. This results in the constraint

```
Eq_14 = (x_1 == simplify(x(h)));
disp(Eq_14); disp('(Eq. 14)');
```

$$x_1 = x_0 + \frac{h\,\dot{x}_0}{2} + \frac{h\,\dot{x}_1}{2}$$

```
(Eq. 14)
```

which ensures the continuity of the trajectory across the consecutive intervals $[t_0, t_1]$ and $[t_1, t_2]$.

The last three equations already form a transcription of our ODE in the interval $[t_0, t_1]$ since, if x_0, u_0, and u_1 were known, they would suffice to determine their unknowns $\dot{x}_0$, $\dot{x}_1$, and x_1. However, to avoid treating $\dot{x}_0$ and $\dot{x}_1$ as decision variables, we can apply the substitutions $\dot{x}_0 = f_0$ and $\dot{x}_1 = f_1$ to the last equation, to obtain

```
Eq_15 = subs(Eq_14,[x_dot_0 x_dot_1],[f_0 f_1]);
disp(Eq_15); disp('(Eq. 15)');
```

$$x_1 = x_0 + \frac{f_0\,h}{2} + \frac{f_1\,h}{2}$$

```
(Eq. 15)
```

Eq. (15) is the common transcription rule from trapezoidal collocation (Kelly 2017, Betts 2010).

The Hermite-Simpson method

In Hermite-Simpson collocation, the state trajectories $x(\tau)$ in each interval are approximated by **cubic polynomials:**

$$x(\tau) = a + b\tau + c\tau^2 + d\tau^3$$

$$\dot{x}(\tau) = b + 2c\tau + 3d\tau^2$$

By analogy with the trapezoidal method, we first express the coefficients $a, b, c,$ and d in terms of the parameters

$$\begin{aligned} x_0 &= x(0) \\ \dot{x}_0 &= \dot{x}(0) \\ \dot{x}_c &= \dot{x}(h/2) \\ \dot{x}_1 &= \dot{x}(h) \end{aligned}$$

where the extra parameter $\dot{x}_c$ is added since four parameters are needed to determine a third degree polynomial. Evaluating the right hand sides of these identities using the above expressions for $x(\tau)$ and $\dot{x}(\tau)$, we obtain the system of equations

$$\begin{bmatrix} x_0 \\ \dot{x}_0 \\ \dot{x}_c \\ \dot{x}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & h & \frac{3h^2}{4} \\ 0 & 1 & 2h & 3h^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}.$$

Solving for $a, \ldots, d$ we get:

```
N = [x_0; x_dot_0; x_dot_c; x_dot_1];
A = [1 0 0 0; 0 1 0 0; 0 1 h 3*h^2/4; 0 1 2*h 3*h^2];
soln = A\N;

a = soln(1); b = soln(2); c = soln(3); d = soln(4);
display(a); display(b); display(c); display(d);
```

a = x_0

b = $\dot{x}_0$

c = $-\dfrac{3\,\dot{x}_0 + \dot{x}_1 - 4\,\dot{x}_c}{2\,h}$

d = $\dfrac{2\,(\dot{x}_0 + \dot{x}_1 - 2\,\dot{x}_c)}{3\,h^2}$

Thus, in terms of $x_0, , \dot{x}_0, \dot{x}_c, \dot{x}_1$, the interpolation polynomial $x(\tau)$ can be written as:

```
x(tau) = symfun(a + b*tau + c*tau^2 + d*tau^3,tau)
```

x(tau) = $x_0 + \tau\,\dot{x}_0 + \frac{2\,\tau^3\,(\dot{x}_0 + \dot{x}_1 - 2\,\dot{x}_c)}{3\,h^2} - \frac{\tau^2\,(3\,\dot{x}_0 + \dot{x}_1 - 4\,\dot{x}_c)}{2\,h}$

In order to determine the four parameters of this polynomial, four conditions have to be imposed, and the Hermite-Simpson method makes this by fixing $x(0) = x_0$ (which holds by construction) and imposing the dynamics at the two bounding knot points and the midpoint between them:

$$\begin{aligned}\dot{x}_0 &= f_0,\\ \dot{x}_1 &= f_1,\\ \dot{x}_c &= f_c.\end{aligned}$$

In the latter equation, $f_c = f(x_c, u_c, t_c)$, where $x_c = x(h/2)$, $u_c = u(h/2)$, and $t_c = t_0 + h/2$. Moreover, the values x_c that are needed in f_c can be expressed in terms of the above four parameters by evaluating $x(\tau)$ for $\tau = h/2$, which yields

```
Eq_21 = (x_c == simplify(x(h/2)));
disp(Eq_21); disp('(Eq. 21)');
```

$x_c = x_0 + \frac{5\,h\,\dot{x}_0}{24} - \frac{h\,\dot{x}_1}{24} + \frac{h\,\dot{x}_c}{3}$

```
(Eq. 21)
```

Finally, the continuity constraint between consecutive intervals $[t_0, t_1]$ and $[t_1, t_2]$ is obtained by evaluating the interpolation polynomial for $\tau = h$:

```
Eq_22 = (x_1 == simplify(x(h)));
disp(Eq_22); disp('(Eq. 22)');
```

$x_1 = x_0 + \frac{h\,\dot{x}_0}{6} + \frac{h\,\dot{x}_1}{6} + \frac{2\,h\,\dot{x}_c}{3}$

```
(Eq. 22)
```

The last five equations already form a transcription of our ODE in $[t_0, t_1]$, but a transcription involving less variables can be obtained by applying the substitutions $\dot{x}_0 = f_0, \dot{x}_1 = f_1$, and $\dot{x}_c = f_c$ to the last two equations. This gives

```
Eq_23a = subs(Eq_22,[x_dot_0,x_dot_1,x_dot_c],[f_0,f_1,f_c]);
Eq_23b = subs(Eq_21,[x_dot_0,x_dot_1,x_dot_c],[f_0,f_1,f_c]);

disp(Eq_23a); disp('(Eq. 23a)'); disp(' '); ...
disp(Eq_23b); disp('(Eq. 23b)');
```

$$x_1 = x_0 + \frac{f_0\,h}{6} + \frac{f_1\,h}{6} + \frac{2\,f_c\,h}{3}$$

```
(Eq. 23a)
```

$$x_c = x_0 + \frac{5\,f_0\,h}{24} - \frac{f_1\,h}{24} + \frac{f_c\,h}{3}$$

```
(Eq. 23b)
```

If preferred, we can also remove the dependence on f_c in Eq. 23b. This is achieved by isolating f_c from Eq. (23a) and substituting the result in Eq. (23b), which yields the alternative transcription

```
Eq_24b = subs(Eq_23b,f_c,solve(Eq_23a,f_c));
disp(Eq_23a); disp('(Eq. 24a)'); ...
disp(' '); disp(Eq_24b); disp('(Eq. 24b)');
```

$$x_1 = x_0 + \frac{f_0\,h}{6} + \frac{f_1\,h}{6} + \frac{2\,f_c\,h}{3}$$

```
(Eq. 24a)
```

$$x_c = \frac{x_0}{2} + \frac{x_1}{2} + \frac{f_0\,h}{8} - \frac{f_1\,h}{8}$$

```
(Eq. 24b)
```

Both transcriptions in Eqs. (23) and (24) are called **separated forms** of Hermite-Simpson collocation, in the sense they both keep x_c as a decision variable of the problem. They are equivalent, but the one in (24) allows us to eliminate x_c by substituting Eq. 24b in the expression of f_c in 24a, which results in a single equation that is known as the **compressed form** of Hermite-Simpson collocation (Kelly 2017, Betts 2010). While the use of a separated form tends to be better when working with a small number of intervals, the compressed form is preferable when such a number is large (Kelly 2017).

Second order methods

The trapezoidal method for 2nd order systems

The essential feature characterizing trapezoidal collocation is that the dynamics is imposed just at the knot points or, otherwise said, that each interval bound is a collocation point. When the dynamics is governed by a second order ODE

$$\ddot{q} = g(q, \dot{q}, u, t),$$

using the same strategy as the trapezoidal method will consist in imposing $\ddot{q} = g(q, \dot{q}, u, t)$, at each interval bound. This means that, for each interval, two constraints have to be imposed on the second derivative of the polynomial approximating each component q of the configuration. But, since the second derivative of a quadratic polynomial is constant, only one constraint could be imposed on it. This implies that the interpolating polynomial $q(\tau)$ must be of degree three at least. So we will have, for a given interval,

$$\begin{aligned}
&q(\tau) = a + b\tau + c\tau^2 + d\tau^3,\\
&\dot{q}(\tau) = b + 2c\tau + 3d\tau^2,\\
&\ddot{q}(\tau) = 2c + 6d\tau
\end{aligned}$$

To determine the coefficients a, b, c, d, we need to impose four conditions. While in the trapezoidal method three conditions were used (the value x_0 at the initial bound and the derivatives $\dot{x}_0$ and $\dot{x}_1$ at the two bounds), here we will impose, in addition to the initial value q_0 and the second derivative at the interval bounds $\ddot{q}_0$ and $\ddot{q}_1$, the value v_0 of the first derivative at the initial bound. Note that, for a cubic polynomial, no more than two independent conditions can be fulfilled by its second derivative, so imposing the dynamics at the midpoint of the interval as in the Hermite-Simpson method is not possible here. Thus we will use the parameters

$$\begin{aligned}
q_0 &= q(0)\\
v_0 &= \dot{q}(0)\\
\ddot{q}_0 &= \ddot{q}(0)\\
\ddot{q}_1 &= \ddot{q}(h).
\end{aligned}$$

Evaluating these identities using the previous expressions for $q(\tau)$ and its derivatives, we obtain the system

$$\begin{bmatrix} q_0 \\ v_0 \\ \ddot{q}_0 \\ \ddot{q}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 6h \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

so solving for a, b, c, d, e, we have

```
N = [q_0; v_0; q_ddot_0; q_ddot_1];
A = [1 0 0 0; 0 1 0 0; 0 0 2 0; 0 0 2 6*h];
soln = A\N;

a = soln(1); b = soln(2); c = soln(3); d = soln(4);
display(a); display(b); display(c); display(d);
```

a = q_0

b = v_0

c = $\frac{\ddot{q}_0}{2}$

d = $-\frac{\ddot{q}_0 - \ddot{q}_1}{6\,h}$

Thus, we can write the interpolation polynomial $q(\tau)$ and its derivative as:

```
q(tau) = symfun(a + b*tau + c*tau^2 + d*tau^3,tau)
```

q(tau) = $q_0 + \tau\, v_0 + \frac{\ddot{q}_0\, \tau^2}{2} - \frac{\tau^3\,(\ddot{q}_0 - \ddot{q}_1)}{6\,h}$

```
q_dot(tau) = symfun(b + 2*c*tau + 3*d*tau^2,tau)
```

q_dot(tau) = $v_0 + \ddot{q}_0\, \tau - \frac{\tau^2\,(\ddot{q}_0 - \ddot{q}_1)}{2\,h}$

The evaluation of this polynomial and its derivative $\dot{q}(\tau)$ for $\tau = h$ yields

```
Eq_27a = (q_1 == simplify(q(h)));
Eq_27b = (v_1 == simplify(q_dot(h)));
disp(Eq_27a); disp('(Eq. 27a)'); disp(' '); ...
disp(Eq_27b); disp('(Eq. 27b)');
```

$$q_1 = q_0 + h\,v_0 + \frac{h^2\,\ddot{q}_0}{3} + \frac{h^2\,\ddot{q}_1}{6}$$

```
(Eq. 27a)
```

$$v_1 = v_0 + \frac{h\,\ddot{q}_0}{2} + \frac{h\,\ddot{q}_1}{2}$$

```
(Eq. 27b)
```

Imposing the collocation constraints

$$\ddot{q}_0 = g_0,$$
$$\ddot{q}_1 = g_1,$$

where $g_k = g(q_k, v_k, u_k, t_k)$, we finally obtain the trapezoidal method for second order systems:

```
Eq_29a = subs(Eq_27a,[q_ddot_0, q_ddot_1],[g_0, g_1]);
Eq_29b = subs(Eq_27b,[q_ddot_0, q_ddot_1],[g_0, g_1]);

disp(Eq_29a); disp('(Eq. 29a)'); disp(' '); ...
disp(Eq_29b); disp('(Eq. 29b)');
```

$$q_1 = q_0 + h\,v_0 + \frac{g_0\,h^2}{3} + \frac{g_1\,h^2}{6}$$

```
(Eq. 29a)
```

$$v_1 = v_0 + \frac{g_0\,h}{2} + \frac{g_1\,h}{2}$$

```
(Eq. 29b)
```

Note that, in this case, the trapezoidal rule only applies for the velocity, but not for the configuration itself, which is given by Eq. (29a).

It is worth observing that, as opposed to the trapezoidal method for first order systems, the continuity between neighboring polynomials at the knot points is of second order in this case, since the collocation constraints impose the coincidence of the second derivative of $q(t)$. Second order continuity for the configuration trajectory implies smooth velocity profiles and continuous accelerations, which are desirable properties in many robotics applications.

The Hermite-Simpson method for 2nd order systems

Our purpose now is to impose the second order dynamics on the two bounds and the midpoint of each interval, in similarity with the conventional Hermite-Simpson method. Clearly, if we want to impose three conditions to the second derivative of a polynomial $q(\tau)$, such a derivative must be quadratic at least, what implies that the polynomial must have degree four at least. Thus, we propose to approximate the configuration trajectory, and its derivatives, by

$$\begin{aligned}
q(\tau) &= a + b\tau + c\tau^2 + d\tau^3 + e\tau^4,\\
\dot{q}(\tau) &= b + 2c\tau + 3d\tau^2 + 4e\tau^3,\\
\dot{q}(\tau) &= 2c + 6d\tau + 12e\tau^2.
\end{aligned}$$

Since five parameters are needed to determine the five coefficients of $q(\tau)$, we will use, in addition to the three accelerations $\ddot{q}_0, \ddot{q}_c, \ddot{q}_1$, the values of the configuration coordinate q_0 and its derivative v_0 at the initial point:

$$\begin{aligned}
q_0 &= q(0)\\
v_0 &= \dot{q}(0)\\
\ddot{q}_0 &= \ddot{q}(0)\\
\ddot{q}_c &= \ddot{q}(h/2)\\
\ddot{q}_1 &= \ddot{q}(h).
\end{aligned}$$

Evaluating the right hand sides we obtain the following system of equations

$$\begin{bmatrix} q_0 \\ v_0 \\ \ddot{q}_0 \\ \ddot{q}_c \\ \ddot{q}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 3h & 3h^2 \\ 0 & 0 & 2 & 6h & 12h^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix}$$

Solving for $a, \ldots, e$ we get

```
N = [q_0; v_0; q_ddot_0; q_ddot_c; q_ddot_1];
A = [1 0 0 0 0;
     0 1 0 0 0;
     0 0 2 0 0;
     0 0 2 3*h 3*h^2; 0 0 2 6*h 12*h^2];

soln = A\N;
a = soln(1); b = soln(2); c = soln(3); d = soln(4); e = soln(5);
```

```
display(a); display(b); display(c); display(d); display(e);
```

a = q_0

b = v_0

c = $\frac{\ddot{q}_0}{2}$

d = $-\frac{3\,\ddot{q}_0+\ddot{q}_1-4\,\ddot{q}_c}{6\,h}$

e = $\frac{\ddot{q}_0+\ddot{q}_1-2\,\ddot{q}_c}{6\,h^2}$

Thus, we can write the interpolation polynomial $q(\tau)$ and its derivative as:

```
q(tau) = symfun(a + b*tau + c*tau^2 + d*tau^3 + e*tau^4,tau)
```

q(tau) = $q_0+\tau\,v_0+\frac{\ddot{q}_0\,\tau^2}{2}+\frac{\tau^4\,(\ddot{q}_0+\ddot{q}_1-2\,\ddot{q}_c)}{6\,h^2}-\frac{\tau^3\,(3\,\ddot{q}_0+\ddot{q}_1-4\,\ddot{q}_c)}{6\,h}$

```
q_dot(tau) = symfun(b + 2*c*tau + 3*d*tau^2 + 4*e*tau^3,tau)
```

q_dot(tau) = $v_0+\ddot{q}_0\,\tau+\frac{2\,\tau^3\,(\ddot{q}_0+\ddot{q}_1-2\,\ddot{q}_c)}{3\,h^2}-\frac{\tau^2\,(3\,\ddot{q}_0+\ddot{q}_1-4\,\ddot{q}_c)}{2\,h}$

Evaluating $q(\tau)$ and its derivative for $\tau = h$ we get

```
Eq_34a = (q_1 == simplify(q(h)));
Eq_34b = (v_1 == simplify(q_dot(h)));
disp(Eq_34a); disp('(Eq. 34a)'); disp(' '); ...
disp(Eq_34b); disp('(Eq. 34b)');
```

$$q_1 = q_0 + h\,v_0 + \frac{h^2\,\ddot{q}_0}{6} + \frac{h^2\,\ddot{q}_c}{3}$$

(Eq. 34a)

$$v_1 = v_0 + \frac{h\,\ddot{q}_0}{6} + \frac{h\,\ddot{q}_1}{6} + \frac{2\,h\,\ddot{q}_c}{3}$$

(Eq. 34b)

and imposing the collocation constraints

$$\begin{aligned}\ddot{q}_0 &= g_0,\\ \ddot{q}_c &= g_c,\\ \ddot{q}_1 &= g_1,\end{aligned}$$

yields

```
Eq_36a=subs(Eq_34a,[q_ddot_0,q_ddot_1,q_ddot_c],[g_0,g_1,g_c]);
Eq_36b=subs(Eq_34b,[q_ddot_0,q_ddot_1,q_ddot_c],[g_0,g_1,g_c]);

disp(Eq_36a); disp('(Eq. 36a)'); disp(' '); ...
disp(Eq_36b); disp('(Eq. 36b)');
```

$$q_1 = q_0 + h\,v_0 + \frac{g_0\,h^2}{6} + \frac{g_c\,h^2}{3}$$

```
(Eq. 36a)
```

$$v_1 = v_0 + \frac{g_0\,h}{6} + \frac{g_1\,h}{6} + \frac{2\,g_c\,h}{3}$$

```
(Eq. 36b)
```

where we recognize that Eq. (36b) is the Simpson quadrature for the velocity. The terms g_c in these equations involve the midpoint coordinate $q_c = q(h/2)$, and the velocity $v_c = \dot{q}(h/2)$, but these can be obtained by evaluating the interpolation polynomial and its derivative for $\tau = h/2$, and imposing the collocation constraints, which yields

```
q_hhalf = simplify(q(h/2)); v_hhalf = simplify(q_dot(h/2));
Eq_37a = ...
    (q_c == subs(q_hhalf,[q_ddot_0 q_ddot_1 q_ddot_c],...
    [g_0 g_1 g_c]));
Eq_37b = ...
    (v_c == subs(v_hhalf,[q_ddot_0 q_ddot_1 q_ddot_c],...
    [g_0 g_1 g_c]));

disp(Eq_37a); disp('(Eq. 37a)'); disp(' '); ...
disp(Eq_37b); disp('(Eq. 37b)');
```

$$q_c = q_0 + \frac{h\,v_0}{2} + \frac{7\,g_0\,h^2}{96} - \frac{g_1\,h^2}{96} + \frac{g_c\,h^2}{16}$$

```
(Eq. 37a)
```

$$v_c = v_0 + \frac{5\,g_0\,h}{24} - \frac{g_1\,h}{24} + \frac{g_c\,h}{3}$$

```
(Eq. 37b)
```

Eqs. (36a) to (37b) together constitute the **separated form** of the second order Hermite-Simpson method. Note however that, since q_c and v_c are to be used in the evaluation of g_c, we may prefer not to express them in terms of g_c itself. For this we simply isolate g_c from Eq. (36b) and substitute the result in Eqs. (37a) and (37b) to yield:

```
g_csolved = solve(Eq_36b,g_c);
Eq_38a = (q_c == simplify(subs(rhs(Eq_37a),g_c,g_csolved)));
Eq_38b = (v_c == simplify(subs(rhs(Eq_37b),g_c,g_csolved)));

disp(Eq_38a); disp('(Eq. 38a)'); disp(' '); ...
disp(Eq_38b); disp('(Eq. 38b)');
```

$$q_c = q_0 + \frac{13\,h\,v_0}{32} + \frac{3\,h\,v_1}{32} + \frac{11\,g_0\,h^2}{192} - \frac{5\,g_1\,h^2}{192}$$

```
(Eq. 38a)
```

$$v_c = \frac{v_0}{2} + \frac{v_1}{2} + \frac{g_0\,h}{8} - \frac{g_1\,h}{8}$$

```
(Eq. 38b)
```

Written in this way, q_c and v_c can be replaced in the expression of g_c in Eqs. (36a) and (36b) to transcribe the problem in **compressed form**, i.e., eliminating the need to treat q_c and v_c as decision variables of the NLP problem.

In this collocation scheme, the continuity across knot points is also of second order due to the coincidence of the second derivative imposed by the collocation constraints, what gives rise to smooth, continuous acceleration trajectories just like in the second order trapezoidal method.

Print execution time

```
toc;
```

```
Elapsed time is 3.199847 seconds.
```

References

Hairer, Ernst, Christian Lubich, and Gerhard Wanner. *Geometric Numerical integration: structure-preserving algorithms for ordinary differential equations.* Springer, 2006.

Hargraves, Charles R., and Stephen W. Paris. "Direct trajectory optimization using nonlinear programming and collocation." *Journal of guidance, control, and dynamics* 10.4 (1987): 338-342.

Kelly, Matthew. "An introduction to trajectory optimization: How to do your own direct collocation." *SIAM Review* 59.4 (2017): 849-904.

Betts, J. T. (2010). Practical Methods for Optimal Control and Estimation Using Nonlinear Programming. SIAM.