Modified Collocation Methods for Functional Differential Equations

Stepan Samulevich

Abstract. We modify classical collocation methods so that delayed and advanced arguments are evaluated by the same polynomial that represents the solution on a step. The derivative is interpolated at Lobatto nodes in Newton form, while shifted stage values are obtained by integrating this polynomial beyond the current step when necessary. The resulting implicit equations are solved by a Gauss–Seidel iteration initialized by extrapolation from the preceding step. The construction applies to first- and second-order functional differential equations and treats delay and advance terms uniformly.

Keywords: collocation methods, functional differential equations, delay differential equations, advanced arguments, Lobatto nodes.

Introduction

Functional differential equations contain values of the unknown function at shifted times. A representative first-order problem is

$$\dot{x}(t) = f\big(t, x(t), x(t \pm \tau)\big), \qquad x(t_0) = x_0, \qquad \tau > 0. \tag{1}$$

The minus sign describes a delay and the plus sign an advance. Standard delay solvers usually recover shifted values from a stored history, whereas an advanced value is not yet available. Nested integration can handle both cases, but requires repeatedly solving auxiliary initial-value problems [1, 2, 3].

The proposed modification is simple: shifted values are evaluated from the collocation polynomial itself. Delay and advance terms therefore enter the numerical scheme in exactly the same form. This is useful for problems that must be integrated both forward and backward in time.

1. Modified collocation scheme

Let

$$t_i = t_0 + hc_i, \qquad 0 = c_1 < c_2 < \cdots < c_s = 1,$$

where c_i are Lobatto nodes and h is the step size. Define

$$p(c) := \dot{u}(t_0 + hc), \tag{2}$$

where u approximates the solution on the step. The collocation conditions are

$$p(c_i) = f_i, \qquad f_i = f\big(t_i, u(t_i), u(t_i \pm \tau)\big), \qquad i = 1, \dots, s. \tag{3}$$

Introduce

$$\mathcal{I}_k p(c) := \frac{1}{(k-1)!} \int_0^c (c - \xi)^{k-1} p(\xi)\, d\xi, \qquad k \geq 1. \tag{4}$$

For a first-order equation,

$$u(t_i) = x_0 + h\mathcal{I}_1 p(c_i), \qquad u(t_i \pm \tau) = x_0 + h\mathcal{I}_1 p(c_i \pm \delta), \qquad \delta = \frac{\tau}{h}. \tag{5}$$

The second formula is the essential modification. If $c_i \pm \delta$ lies outside $[0, 1]$, the same collocation polynomial is extrapolated rather than switching to a separate history or advance procedure.

For a second-order equation

$$\ddot{x}(t) = f\big(t, x(t), \dot{x}(t), x(t \pm \tau), \dot{x}(t \pm \tau)\big), \tag{6}$$

set $p(c) = \ddot{u}(t_0 + hc)$. With $v = \dot{u}$,

$$\begin{aligned} u(t_0 + hc) &= x_0 + hx_0' c + h^2 \mathcal{I}_2 p(c), \\ v(t_0 + hc) &= x_0' + h\mathcal{I}_1 p(c), \\ u(t_0 + h(c \pm \delta)) &= x_0 + hx_0'(c \pm \delta) + h^2 \mathcal{I}_2 p(c \pm \delta), \\ v(t_0 + h(c \pm \delta)) &= x_0' + h\mathcal{I}_1 p(c \pm \delta). \end{aligned} \tag{7}$$

Thus, the first- and second-order constructions differ only in the number of integrations applied to p.

1.1. Newton–Lobatto implementation

Write

$$p(c) = \sum_{j=1}^{s} a_j \prod_{m=1}^{j-1} (c - c_m), \qquad a_j = [f_1, \dots, f_j]. \tag{8}$$

The integrated basis functions

$$\gamma_{j,k}(c) := \mathcal{I}_k \left[\prod_{m=1}^{j-1} (c - c_m) \right] \tag{9}$$

satisfy

$$\gamma_{j,k}(c) = (c - c_{j-1})\gamma_{j-1,k}(c) - k\gamma_{j-1,k+1}(c), \qquad \gamma_{1,k}(c) = \frac{c^k}{k!}. \tag{10}$$

Hence, all stage values are linear combinations of a_j with precomputed weights evaluated at c_i or $c_i \pm \delta$.

Because f_i depends on the polynomial coefficients, the stage equations are implicit. They are solved by Gauss–Seidel iteration. The first step uses the initial condition, or a linear predictor for a second-order problem. Later steps use $p_{\mathrm{prev}}(1+c_i)$ as the predictor. Since Lobatto nodes include both endpoints, the final derivative of one step is reused as the first derivative of the next step (FSAL).

The algorithm is:

1. predict the stage derivatives from the preceding polynomial;
2. integrate the Newton polynomial at c_i and $c_i \pm \tau/h$ to form all stage arguments;
3. update the stages and divided differences by Gauss–Seidel iteration;
4. evaluate the integrated polynomial at $c = 1$ to advance the solution.

2. Brief lunar-orbit example

As a short illustration, the method was applied to

$$\ddot{\mathbf{r}}(t) = -\frac{(\mu+\mu_\oplus)\mathbf{r}(t)}{r^3} - \frac{3k_2\mu}{r^3}\left(1+\frac{\mu}{\mu_\oplus}\right)\left(\frac{a_e}{r}\right)^5 \mathbf{r}(t-\tau), \qquad r = \|\mathbf{r}(t)\|, \tag{11}$$

a simplified lunar tidal model [4]. The orbit was propagated for 40 years and then integrated backward. With $s = 11$, $h = 2.5$ days, and two Gauss–Seidel iterations per step, the collocation run used 116861 right-hand-side evaluations. The reference eleventh-order Adams–Bashforth–Moulton delayed calculation [3], with $h = 1/8$ day, used 243953 evaluations.

The example indicates that the modified scheme can use substantially larger steps. A systematic stability and convergence analysis with respect to τ/h remains necessary.

Conclusion

The main modification is the uniform evaluation of current, delayed, and advanced stage values from one integrated collocation polynomial. Lobatto nodes, Newton interpolation, precomputed integral weights, FSAL reuse, and a Gauss–Seidel stage iteration make the resulting implicit method practical. The same construction covers first- and second-order equations without a separate numerical mechanism for advanced arguments.

Acknowledgements

The author thanks Dmitry A. Pavlov for helpful discussions. This work was supported by the Ministry of Science and Higher Education of the Russian Federation

(agreement No. 075-15-2025-344 of April 29, 2025, at the St. Petersburg Leonhard Euler International Mathematical Institute, PDMI RAS).

References

[1] A. D. Polyanin, V. G. Sorokin, and A. I. Zhurov, *Delay Differential Equations: Properties, Methods, Solutions, and Models*, IPMech RAS, Moscow, 2022 (in Russian).

[2] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd ed., Springer, Berlin–Heidelberg, 1993.

[3] D. Aksim and D. Pavlov, On the extension of Adams–Bashforth–Moulton methods for numerical integration of delay differential equations and application to the Moon's orbit, *Mathematics in Computer Science*, vol. 14, pp. 103–109, 2020. `https://doi.org/10.1007/s11786-019-00447-y`.

[4] P. K. Seidelmann, ed., *Explanatory Supplement to the Astronomical Almanac*, University Science Books, Mill Valley, CA, 1992.

Stepan Samulevich
Saint Petersburg Electrotechnical University LETI
St. Petersburg, Russia
e-mail: `stepansamulevic@gmail.com`