BVPs for ODEs
— Finite Difference Methods —

MECH 309 - *Numerical Methods in Mech. Eng.*

Prof. James Richard Forbes ©

McGill University, Department of Mechanical Engineering

Finite-Difference Method

- Consider a second-order ODE with BCs of the form

$$\ddot{y}(t) = f(y(t), \dot{y}(t), t), \quad a < t < b, \quad y(a) = y_a, \quad y(b) = y_b. \tag{1}$$

- Recall that second-order accurate (i.e., $\mathcal{O}((\Delta t)^2)$) difference approximations

$$\dot{y}(t) \approx \frac{y_{k+1} - y_{k-1}}{2\Delta t}, \tag{2}$$

$$\ddot{y}(t) \approx \frac{y_{k+1} - 2y_k + y_{k-1}}{(\Delta t)^2}, \tag{3}$$

  where $\Delta t = t_k - t_{k-1}$ and/or $\Delta t = t_{k+1} - t_k$.

- Using (2) and (3) to approximate (1) gives

$$\frac{y_{k+1} - 2y_k + y_{k-1}}{(\Delta t)^2} = f\left(y_k, \frac{y_{k+1} - y_{k-1}}{2\Delta t}, t_k\right), \quad k = 1, \ldots, K-1 \tag{4}$$

- Form a system of equations, where the BCs $y(a) = y(t_0) = y_0$ and $y(b) = y(t_K) = y_K$ are included, and solve for $y_1, y_2, \ldots, y_{K-2}, y_{K-1}$.

- If the ODE (1) is linear, the system of equations is linear, that is, a big $\mathbf{Ax} = \mathbf{b}$ problem that can be solved using LU, for example.
- If the ODE (1) is nonlinear, the system of equations is nonlinear, that is, a big $\mathbf{f}(\mathbf{x}) = \mathbf{b}$ problem that can be solved using nonlinear least squares.

- In practice, acceptable accuracy of a finite-difference solution requires many *mesh points* in order to reduce the impact of truncation error.
- Of course, can't make the mesh too small, else round-off error accumulates.

Example

Q. Consider

$$\ddot{q}(t) = \gamma q(t), \quad 0 < t < 1, \quad q(a) = 1, \quad q(b) = 3, \tag{5}$$

where $\gamma = 4$, $a = 0$, and $b = 1$.

- Find the solution using finite differences.

A. Discretizing (5) using (2) and (3) gives

$$\frac{q_{k+1} - 2q_k + q_{k-1}}{(\Delta t)^2} = \gamma q_k, \tag{6}$$

$$q_{k-1} + (-\gamma(\Delta t)^2 - 2)q_k + q_{k+1} = 0. \tag{7}$$

- Setting $\Delta t = 0.25$ (s) results in

$$q_0 + (-\gamma(\Delta t)^2 - 2)q_1 + q_2 = 0, \tag{8}$$
$$q_1 + (-\gamma(\Delta t)^2 - 2)q_2 + q_3 = 0, \tag{9}$$
$$q_2 + (-\gamma(\Delta t)^2 - 2)q_3 + q_4 = 0, \tag{10}$$

  for $k = 1, 2, 3$, where q_0 and q_4 are the prescribed (known!) BCs.

- The linear system of equations can be written in familiar form

$$\underbrace{\begin{bmatrix} (-\gamma(\Delta t)^2 - 2) & 1 & 0 \\ 1 & (-\gamma(\Delta t)^2 - 2) & 1 \\ 0 & 1 & (-\gamma(\Delta t)^2 - 2) \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} -q_0 \\ 0 \\ -q_4 \end{bmatrix}}_{\mathbf{b}}. \tag{11}$$

- Solve for q_1, q_2, q_3.

- For smaller Δt the resultant $\mathbf{Ax} = \mathbf{b}$ problem will be

$$\underbrace{\begin{bmatrix} \vartheta & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & \vartheta & 1 & \ddots & 0 & 0 & 0 \\ 0 & 1 & \vartheta & \ddots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \vartheta & 1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & \vartheta & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & \vartheta \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \vdots \\ q_{K-4} \\ q_{K-3} \\ q_{K-2} \\ q_{K-1} \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} -{\color{red}q_0} \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ -{\color{red}q_K} \end{bmatrix}}_{\mathbf{b}}. \qquad (12)$$

where $\vartheta = (-\gamma(\Delta t)^2 - 2)$.

- Solve for $q_1, q_2, q_3, \ldots, q_{K-3}, q_{K-2}, q_{K-1}, q_{K-1}$.

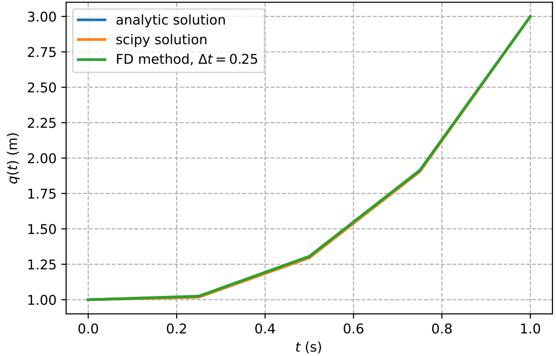

Figure 1: Analytic solution versus `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

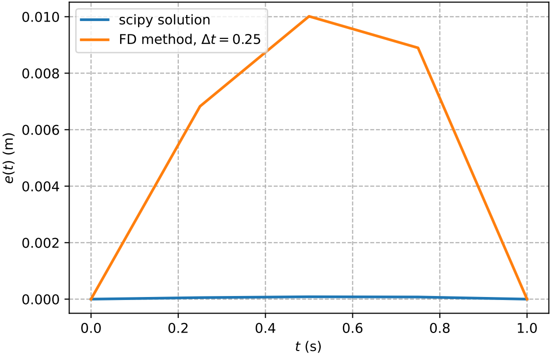

Figure 2: Error between analytic solution and `scipy`'s `solve_bvp` solution, as well as Forbes' custom FD solution.

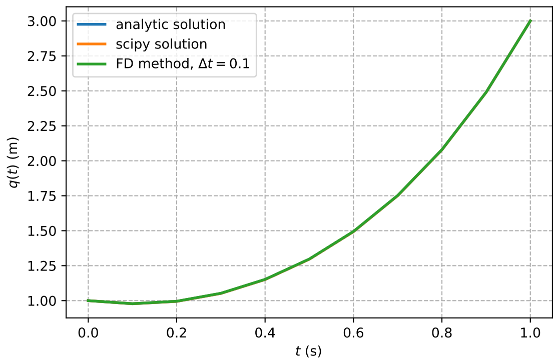

Figure 3: Analytic solution versus `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

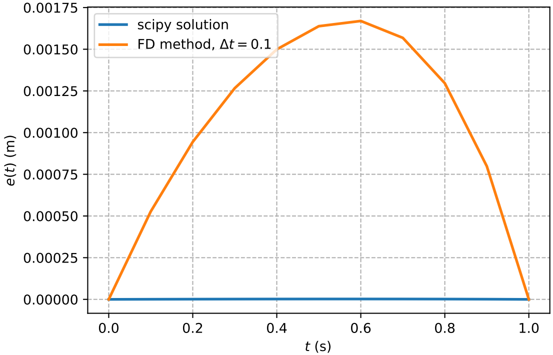

Figure 4: Error between analytic solution and `scipy`'s `solve_bvp` solution, as well as Forbes' custom FD solution.

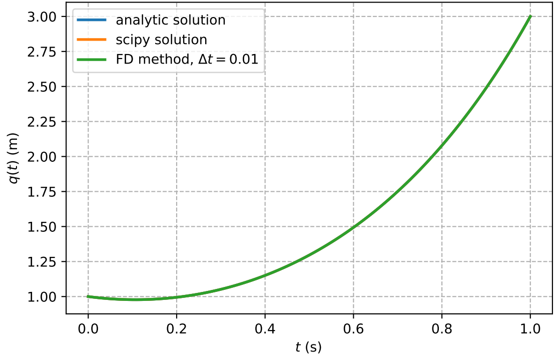

Figure 5: Analytic solution versus `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

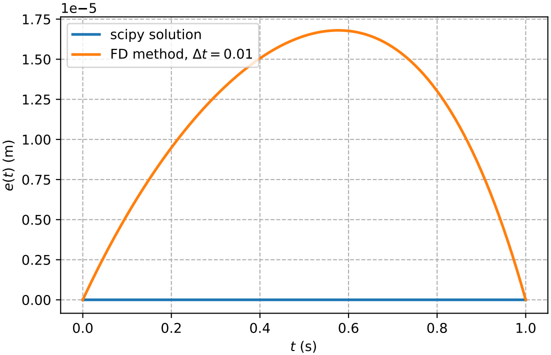

Figure 6: Error between analytic solution and `scipy`'s `solve_bvp` solution, as well as Forbes' custom FD solution.

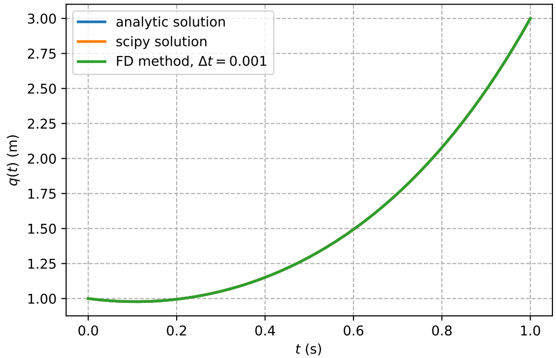

Figure 7: Analytic solution versus `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

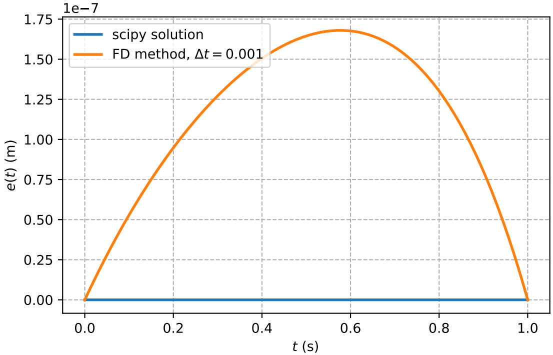

Figure 8: Error between analytic solution and `scipy`'s `solve_bvp` solution, as well as Forbes' custom FD solution.

Example

Q. Consider

$$\ddot{q}(t) = q(t) - (q(t))^2, \quad 0 < t < 1, \quad q(a) = 1, \quad q(b) = 4, \tag{13}$$

$a = 0$ and $b = 1$.

- Find the solution using finite differences.

A. Discretizing (13) using (2) and (3) gives

$$\frac{q_{k+1} - 2q_k + q_{k-1}}{(\Delta t)^2} = q_k - q_k^2, \tag{14}$$

$$q_{k-1} + (-(\Delta t)^2 - 2)q_k + (\Delta t)^2 q_k^2 + q_{k+1} = 0. \tag{15}$$

- Rewrite (15) as

$$\underbrace{\begin{bmatrix} (-(\Delta t)^2 - 2)q_1 + (\Delta t)^2 q_1^2 + q_2 \\ q_1 + (-(\Delta t)^2 - 2)q_2 + (\Delta t)^2 q_2^2 + q_3 \\ \vdots \\ q_{K-4} + (-(\Delta t)^2 - 2)q_{K-3} + (\Delta t)^2 q_{K-3}^2 + q_{K-2} \\ q_{K-2} + (-(\Delta t)^2 - 2)q_{K-1} + (\Delta t)^2 q_{K-1}^2 \end{bmatrix}}_{\mathbf{f}(\mathbf{x})} = \underbrace{\begin{bmatrix} -q_0 \\ 0 \\ \vdots \\ 0 \\ -q_K \end{bmatrix}}_{\mathbf{b}}. \tag{16}$$

- Solve for $\mathbf{x} = \begin{bmatrix} q_1 & q_2 & \cdots & q_{K-2} & q_{K-1} \end{bmatrix}^\mathsf{T}$ using nonlinear least squares.

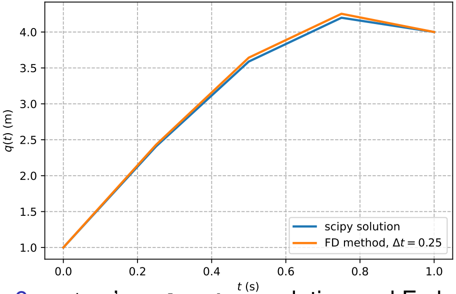

Figure 9: `scipy`’s `solve_bvp` solution and Forbes’ custom FD solution.

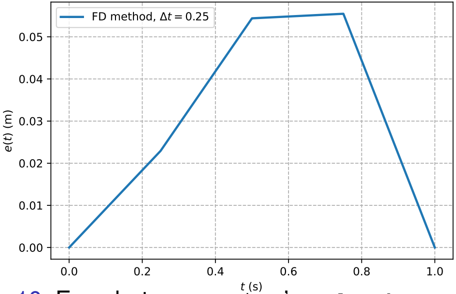

Figure 10: Error between `scipy`’s `solve_bvp` solution Forbes’ custom FD solution.

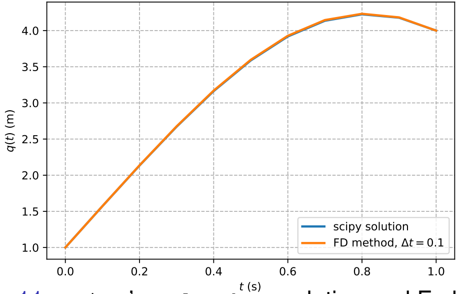

Figure 11: `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

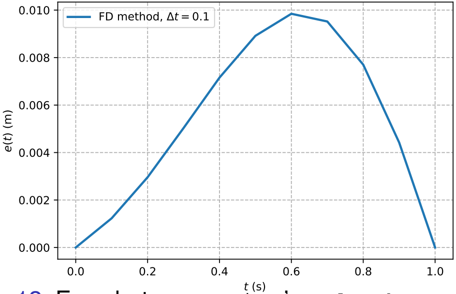

Figure 12: Error between `scipy`'s `solve_bvp` solution Forbes' custom FD solution.

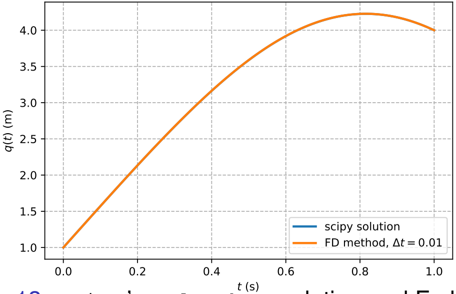

Figure 13: `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

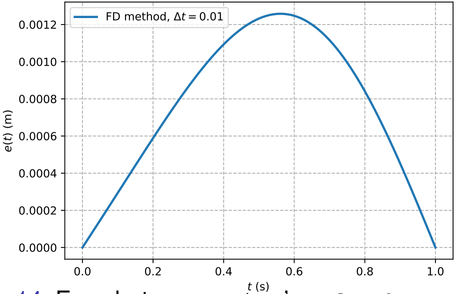

Figure 14: Error between `scipy`'s `solve_bvp` solution Forbes' custom FD solution.

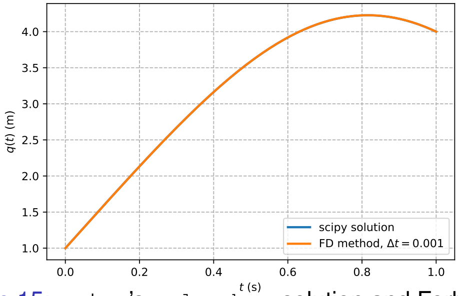

Figure 15: `scipy`'s `solve_bvp` solution and Forbes' custom FD solution.

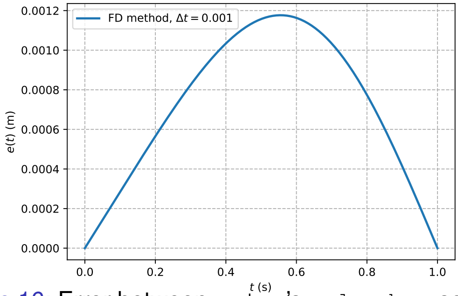

Figure 16: Error between `scipy`'s `solve_bvp` solution Forbes' custom FD solution.

References

Slides based on [1, Ch. 10], [2], [3], [4], [5, Ch. 7], [6, Ch. 9].

[1] M. T. Heath, *Scientific Computing — An Introductory Survey*, Second. New York, NY: McGraw-Hill, 2002.

[2] M. T. Heath, *CS 450: Numerical Analysis/Scientific Computing Lecture Notes*. Urbana-Chamapign, IL: University of Illinois at Urbana-Chamapign, 2019. [Online]. Available: `https://heath.cs.illinois.edu/scicomp/notes/`.

[3] A. Kloeckner, *CS 450: Numerical Analysis/Scientific Computing Lecture Notes*. Urbana-Chamapign, IL: University of Illinois at Urbana-Chamapign, 2019.

[4] E. Solomonik, *CS 450: Numerical Analysis/Scientific Computing Lecture Notes*. Urbana-Chamapign, IL: University of Illinois at Urbana-Chamapign, 2017.

[5] T. Sauer, *Numerical Analysis*, Third. Boston, MA: Pearson, 2018.

[6] D. G. Zill and M. R. Cullen, *Differential Equations with Boundary-Value Problems*, 6th. Toronto, ON: Thomson Brooks/Cole, 2005.