An implicit method of steps for differential equations with state-dependent delay

Kevin E. M. Church

McGill University, Department of Mathematics and Statistics
August 17, 2021

Abstract

In this short paper, we present an approach to the numerical integration of differential equations with state-dependent delay using an implicit method of steps.

1 Introduction

The method of steps for delay differential equations is beautiful in its simplicity: given the initial-value problem

$$\dot{x}(t) = f(x(t), x(t-\tau_1), \ldots, x(t-\tau_n)), \qquad x(\theta) = \phi(\theta),\ \theta \in [-\tau, 0]$$

for real constant delays $\{\tau_i,\ i = 1, \ldots, n\}$, and $\tau = \max\{\tau_1, \ldots, \tau_n\}$, one observes for $t \in [0, \tau]$ that $x(t-\tau_i) = \phi(t-\tau_i)$ for $i = 1, \ldots, n$. As a consequence, solving the above initial-value problem on the interval $[0, \tau]$ is equivalent to solving

$$\dot{x}(t) = f(x(t), \phi(t-\tau_1), \ldots, \phi(t-\tau_n)), \qquad x(0) = \phi(0).$$

Once the solution on $[0, \tau]$ has been computed, the procedure can be repeated to compute the solution on $[\tau, 2\tau]$, $[2\tau, 3\tau]$ and so on. This is the *method of steps*, and its applications in the sciences and numerical mathematics [4, 7, 8, 9, 10, 12] are diverse.

When the delays τ_i are instead functions of time, the method of steps remains applicable so long as each delay $\tau_i : \mathbb{R} \to \mathbb{R}$ has bounded range in the half-line $[0, \infty)$. In this case, with $\tau = \sup_{t\in\mathbb{R}} \max_i \tau_i(t)$, the method of steps proceeds with no modification.

Suppose now that each of the delays τ_i is a function $\tau_i : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$. Once again, if the range of the delays are constrained, then the method of steps can proceed. However, in many applications — see e.g. [2, 3, 5, 6, 11] — these delays can be unbounded and, in principle, could even be negative. The method of steps is no longer applicable. In some cases, the delays are not given by explicit functions and are instead defined implicitly.

In this short paper, we present a superficially simple modification to the method of steps that allows it to be applied to the case of differential equations with state-dependent delay. We then provide a further adjustment that is applicable when the delays are implicitly defined. Finally, we test an implementation of the method on a sample problem.

We should emphasize that while the proposed method may be of interest to those readers that require simulations of differential equations with state-dependent delay, our motivation is ultimately to formalize the method in the language of rigorous numerics and use it to obtain computer-assisted proofs of connecting orbits. It is for this reason that we present an associated zero-finding problem in Section 3, to which Newton's method is applicable and a rigorous numerics approach could be applied.

2 Implicit method of steps

Consider the initial-value problem

$$\dot{x}(t) = f(x(t), x(t-\tau_1), \ldots, x(t-\tau_n)) \tag{1}$$
$$x(\theta) = \phi(\theta), \qquad \theta \in \mathcal{J} \tag{2}$$

where $\mathcal{J}$ is a real, closed interval with $\sup \mathcal{J} < \infty$, and the delays are $\tau_i = \tau_i(t, x(t))$ for some functions $\tau_i : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$. We will assume from this point on that $f : (\mathbb{R}^d)^{n+1} \to \mathbb{R}^d$, all delays $\tau_i : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$, and the initial function ϕ are continuously differentiable, with $d \geq 1$.

Definition 1. *Let $g_1 : [a, b] \to \mathbb{R}^d$ and $g_2 : [b, c] \to \mathbb{R}^d$ satisfy $g_1(a) = g_2(b)$. The* annealing *of g_1 and g_2 is the unique function $g_1 \cup g_2 : [a, c] \to \mathbb{R}^d$ such that $g_1 \cup g_2|_{[a,b]} = g_1$ and $g_1 \cup g_2|_{[b,c]} = g_2$.*

The *implicit method of steps* is the following algorithm. It iteratively constructs a function $x : \mathcal{I} \to \mathbb{R}^d$ on an interval $\mathcal{I} \supset \mathcal{J}$ and a discrete set of *crossing times* $\mathcal{T} \subset \mathcal{I}$.

0. Initialize solution: $x \leftarrow\!\shortmid \phi$, $\mathcal{I} \leftarrow\!\shortmid \mathcal{J}$, $\mathcal{T} \leftarrow\!\shortmid \emptyset$. Define $t_0 = \sup \mathcal{J}$ and verify $\bigcup_i \{t_0 - \tau_i(t_0, \phi(t_0))\} \subset \mathcal{J}$.
1. Select an index $m \in \{1, \ldots, n\}$. Solve the following boundary-value problem for $\psi : [t_0, t_1] \to \mathbb{R}^d$:

$$\dot{\psi}(t) = f(\psi(t), \phi(t - \tau_1(t, \psi(t))), \ldots, \phi(t - \tau_n(t, \psi(t)))), \quad t \in [t_0, t_1] \tag{3}$$
$$\psi(t_0) = \phi(t_0) \tag{4}$$
$$t_1 - \tau_m(t_1, \psi(t_1)) = t_0. \tag{5}$$

2. Update the solution: $x \leftarrow\!\shortmid x \cup \psi$, $\mathcal{I} \leftarrow\!\shortmid \mathcal{J} \cup [t_0, t_1]$, $\mathcal{T} \leftarrow\!\shortmid \mathcal{T} \cup \{t_0, t_1\}$.
3. Update the initial data: $\phi \leftarrow\!\shortmid x$, $\mathcal{J} \leftarrow\!\shortmid \mathcal{I}$. Define $t_0 = \sup \mathcal{J}$.
4. Return to step 1.

Proposition 1. *Every time step 4 of the implicit method of steps is reached, the function $x : \mathcal{I} \to \mathbb{R}^d$ is a solution of the initial-value problem* (1)–(2).

Proof. Suppose step 4 has been reached for the first time. Consider the state of the algorithm at the end of step 2. Due to step 1, we have $t - \tau_i(t, \psi(t)) \in \mathcal{J}$ for $t \in [t_0, t_1]$ and $i = 1, \ldots, n$. By definition of x — see step 2 — it follows that $x(t - \tau_i(t, x(t))) = \phi(t - \tau_i(t, \psi(t)))$ for $t \in [t_0, t_1]$. At the end of step 2, we have $x|_{[t_0,t_1]} = \psi$, from which it follows that x satisfies the differential equation (1). It also satisfies the initial condition, since $x|_{\mathcal{J}} = \phi$. Therefore, $x : \mathcal{I} \to \mathbb{R}^d$ is a solution of the initial-value problem. Given step 3, the result follows by a straightforward inductive argument. □

Remark 1. *The set $\mathcal{T}$ is not essential to the algorithm; it is merely a construct that collects the* crossing times *of the time lag (with distinguished index of step 1) with the right boundary of the initial condition ϕ at each step. In step 1, the selection of the index $m \in \{1, \ldots, n\}$ would ideally be done based on* a priori *knowledge of which of the delays i will satisfy the equality $t - \tau_i(t, \psi(t)) = t_0$* first *in the interval (t_0, ∞). Of course, this information will be in general unavailable. An alternative is to replace the terminal boundary condition* (5) *with*

$$0 = \prod_{i=1}^{n} (t_1 - \tau_i(t_1, \psi(t_1)) - t_0). \tag{6}$$

If this equation is satisfied, then $t - \tau_i(t_1, \psi(t_1)) = t_0$ for at least one index i.

3 Transformation to a zero-finding problem

The only noteworthy part of the implementation of the implicit method of steps is in solving the BVP (3)–(5). Existing black-box software can be used to accomplish this if desired. As an alternative, we will present a transformation to a zero-finding problem to which one can apply Newton's method after

applying discretization. To do this, we will need a piece of formal machinery. Let $\mathcal{E}_k : C^k([\alpha,\beta],\mathbb{R}^d) \to C^k(\mathbb{R},\mathbb{R}^d)$ be the linear map defined by

$$\mathcal{E}_k y(x) = \begin{cases} y(x), & x \in [\alpha,\beta] \\ \sum_{r=0}^k \frac{(x-\alpha)^r}{r!} y^{(r)}(\alpha), & x < \alpha \\ \sum_{r=0}^k \frac{(x-\beta)^r}{r!} y^{(r)}(\beta), & x > \beta \end{cases}$$

That is, $\mathcal{E}_k y$ is simply a degree k Taylor extension of y. This map commutes with differentiation in the following sense: $\frac{d}{dx}\mathcal{E}_k = \mathcal{E}_{k-1}\frac{d}{dx}$. In what follows we will drop the subscript on $\mathcal{E}$.

Assume without loss of generality that $t_0 = 0$. Let us denote $\delta = t_1$ to remove another subscript. Further, let us densify our notation by defining $h_i(t,x) = t - \tau_i(t,x)$ for $i = 1,\dots,n$. If we perform a re-scaling of time to our boundary-value problem by defining $\tilde{\psi}(t) = \psi(t\delta)$, substituting this into (3)–(5) and dropping the tildes, we get

$$\dot{\psi}(t) = \delta f(\psi(t), \phi(h_1(t\delta,\psi(t))),\dots,\phi(h_n(t\delta,\psi(t)))), \quad t \in [0,1] \tag{7}$$

$$\psi(0) = \phi(0) \tag{8}$$

$$0 = h_m(\delta,\psi(1)). \tag{9}$$

Recalling that $\phi \in C^1(\mathcal{J},\mathbb{R}^d)$, consider the nonlinear map $F : C([0,1],\mathbb{R}^d)\times\mathbb{R} \to C([0,1],\mathbb{R}^d)\times\mathbb{R}$,

$$F(\psi,\delta) = \begin{pmatrix} -\psi + \phi(0) + \delta\int_0^{(\cdot)} f(\psi(s), \mathcal{E}\phi(h_1(s\delta,\psi(s))),\dots,\mathcal{E}\phi(h_n(s\delta,\psi(s))))ds \\ h_m(\delta,\psi(1)) \end{pmatrix} \tag{10}$$

Let $C([0,1],\mathbb{R}^d)\times\mathbb{R}$ be equipped with the norm $||(u,w)|| = \max\{||u||_\infty, |w|\}$, where $||\cdot||_\infty$ is the supremum norm on $C([0,1],\mathbb{R}^d)$.

Theorem 2. *$F : C([0,1],\mathbb{R}^d)\times\mathbb{R} \to C([0,1],\mathbb{R}^d)\times\mathbb{R}$ is continuously differentiable. If $F(\psi,\delta) = 0$ and, further, $h_i(t\delta,\psi(t)) \in \mathcal{J}$ for all $t \in [0,1]$ and $i = 1,\dots,n$, then $t \mapsto \psi(t/\delta)$ is a solution of the BVP (3)–(5), with $t_0 = 0$ and $t_1 = \delta$. In particular, ψ is continuously differentiable. If, further, $\phi \in C^2(\mathcal{J},\mathbb{R}^d)$ and each of f and τ_i are twice continuously differentiable, then so are F and ψ.*

Proof. This follows mostly from the commutativity of the Taylor extension with the derivative, the continuous differentiability of f and all of the delay functions τ_i, and the C^1 initial condition ϕ. It is necessary to verify $h_i(t,\delta,\psi(t)) \in \mathcal{J}$ for $t \in [0,1]$ in order to ensure that ψ will actually satisfy the differential equation (7), which does not have the Taylor extension operator $\mathcal{E}$ and instead has the prior inclusion implicit. □

Remark 2. *If the map $\mathcal{E}$ is not included in (10), then F need not be well-defined.*

Remark 3. *The Fréchet derivative of the second component, $F_2(\psi,\delta) = h_m(\delta,\psi(1))$, is easy to compute. Namely, it is*

$$DF_2(\psi,\delta)v = D_1 h_m(\delta,\psi(1))v_2 + D_2 h_m(\delta,\psi(1))v_1(1)$$

for $v = (v_1,v_2) \in C([0,1],\mathbb{R}^d)\times\mathbb{R}$ and D_1, D_2 the Fréchet derivatives in the first and second variables. The Fréchet derivatives of the first component, F_1, are a bit more complicated. In the case of a single lag function h, they are, for $u \in C([0,1],\mathbb{R}^d)$,

$$\begin{aligned} D_1F_1(\psi,\delta)u = -u &+ \delta\int_0^{(\cdot)} D_1 f(\psi(s),\mathcal{E}\phi(h(s\delta,\psi(s))))u(s)ds \\ &+ \delta\int_0^{(\cdot)} D_2 f(\psi(s),\mathcal{E}\phi(h(s\delta,\psi(s))))\mathcal{E}\phi'(h(s\delta,\psi(s)))D_2 h(s\delta,\psi(s))u(s)ds \\ D_2F_1(\psi,\delta) = &\int_0^{(\cdot)} f(\psi(s),\mathcal{E}\phi(h(s\delta,\psi(s))))ds \\ &+ \delta\int_0^{(\cdot)} D_2 f(\psi(s),\mathcal{E}\phi(h(s\delta,\psi(s))))\mathcal{E}\phi'(h(s\delta,\psi(s)))D_1 h(s\delta,\psi(s))sds. \end{aligned}$$

Remark 4. *If the terminal boundary condition* (5) *is replaced with* (6)*, then the second component of F is replaced with $F_2(\psi, \delta) = \prod_{i=1}^{n} h_i(\delta, \psi(1))$.*

In practice, the domain of F is discretized by a projection procedure, and we iterate Newton's method on a finite-dimensional vector space. See Section 5.

4 A modification for implicitly defined delays

Suppose now that instead of being explicitly defined functions, the delays τ_i are defined in terms of an implicit relationship between the state variables. We assume that at all times t (after integration has started), the delays $(\tau_1, \ldots, \tau_n)$ satisfy

$$0 = g(t, x(t), x(t-\tau_1), \ldots, x(t-\tau_n), \tau_1, \ldots, \tau_n) \tag{11}$$

for some continuously differentiable function $g : \mathbb{R} \times (\mathbb{R}^d)^n \times \mathbb{R}^n \to \mathbb{R}^n$. Define $X = C([0,1], \mathbb{R}^d) \times \mathbb{R} \times C([0,1], \mathbb{R})^n$ and consider the map $G : X \to X$ defined by

$$F(\psi, \delta, \tau) = \begin{pmatrix} -\psi + \phi(0) + \delta \int_0^{(\cdot)} f(\psi(s), \mathcal{E}\phi(\delta(s-\tau_1(s))), \ldots, \mathcal{E}\phi(\delta(s-\tau_n(s))) ds \\ \delta(1-\tau_m(1)) \\ t \mapsto g(t\delta, \psi(t), \mathcal{E}\phi(\delta(t-\tau_1(t))), \ldots, \mathcal{E}\phi(\delta(t-\tau_n(t))), \delta\tau_1(t), \ldots, \delta\tau_n(t)) \end{pmatrix}. \tag{12}$$

Essentially by construction, we have the following result whose proof we once again omit.

Theorem 3. *$G : X \to X$ is continuously differentiable. If $G(\psi, \delta, \tau) = 0$ and, further, $\delta(t - \tau_i(t)) \in \mathcal{J}$ for all $t \in [0,1]$ and $i = 1, \ldots, n$, then $(z(t), \rho(t)) = (\psi(t/\delta), \delta\tau(t/\delta))$ is a solution of the differential-algebraic boundary-value problem*

$$\dot{z}(t) = f(z(t), \phi(t-\rho_1(t)), \ldots, \phi(t-\rho_n(t))), \qquad t \in [0, \delta] \tag{13}$$

$$0 = g(z(t), \phi(t-\rho_1(t)), \ldots, \phi(t-\rho_n(t)), \rho_1(t), \ldots, \rho_n(t)), \qquad t \in [0, \delta] \tag{14}$$

$$z(0) = \phi(0) \tag{15}$$

$$0 = \delta - \rho_m(\delta). \tag{16}$$

In particular, ψ and τ are continuously differentiable. Moreover, $x = \phi \cup z$ is a solution of the initial-value problem (1)–(2) *subject to the constraint that the delays satisfy* (11)*. If furthermore ϕ, f and g are twice continuously differentiable, then so is G.*

5 An implementation and example

We discretize F by linear interpolation. This is not the only choice, but it is easy to implement in a high-level programming language. A similar discretization could be done for G.

Let $\mathcal{P} : C([0,1], \mathbb{R}^d) \to C([0,1], \mathbb{R}^d)$ denote the linear interpolation operator on $M \geq 1$ sub-intervals of $[0,1]$ of equal width. Denote $S_M = \mathcal{P}(C([0,1], \mathbb{R}^d))$, the space of all linear interpolants. Then $\mathcal{P}$ factors by way of an isomorphism $\mathcal{S} : (\mathbb{R}^d)^{M+1} \to S_M$, in the sense that $\mathcal{P}y = \mathcal{S}(y(x_0), \ldots, y(x_M))$, where $x_i = \frac{i}{M}$. Explicitly, the inverse of $\mathcal{S}$ is the gridpoint evaluation: $\mathcal{S}^{-1}u = (u(x_0), \ldots, u(x_M))$. We then define $\overline{F} : (\mathbb{R}^d)^{M+1} \times \mathbb{R} \to (\mathbb{R}^d)^{M+1} \times \mathbb{R}$ componentwise, $\overline{F} = (\overline{F}_0, \ldots, \overline{F}_M, \overline{F}_*)$,

$$\overline{F}_i(u, \delta) = -u_i + \phi(0) + \delta \int_0^{x_i} f(\mathcal{S}u(s), \mathcal{E}\phi(h_1(\delta s, \mathcal{S}u(s))), \ldots, \mathcal{E}\phi(h_n(\delta s, \mathcal{S}u(s)))) ds, \quad i = 0, \ldots, M$$

$$\overline{F}_*(u, \delta) = h_m(\delta, u_M).$$

Our example will involve the equation

$$\dot{x}(t) = a\big(x(t) - x(t - r(x(t)))\big) - |x(t)|x(t), \tag{17}$$

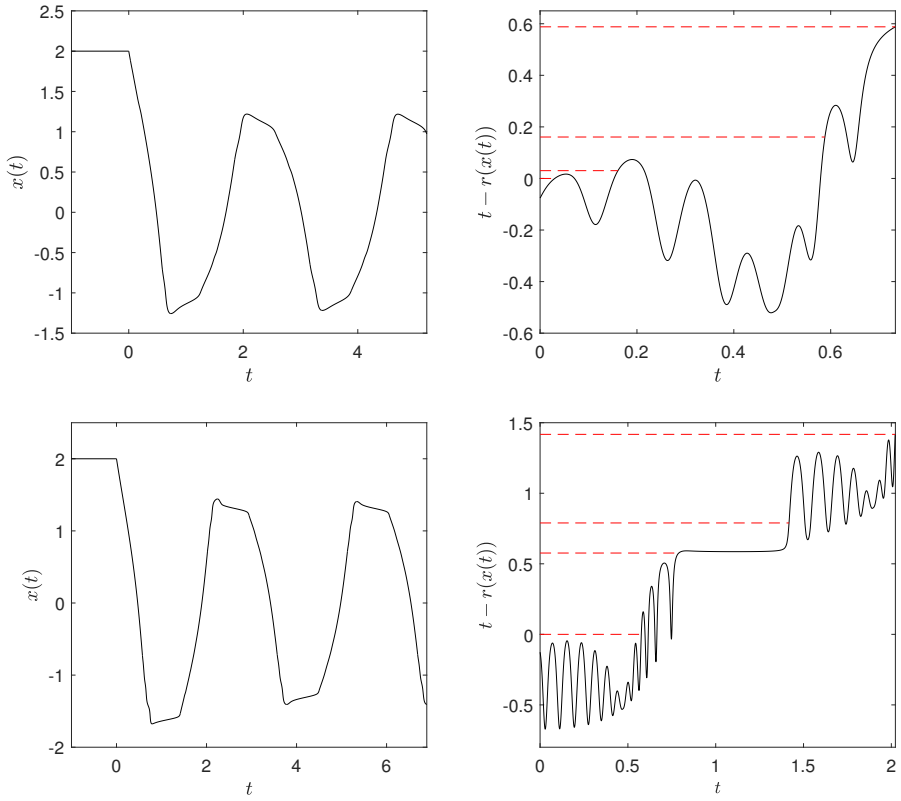

Figure 1: Top left: for $(\eta, \mu) = (6, 1/2)$, solution of (17) satisfying the initial condition $x_0(\theta) = 2$ for $\theta \leq 0$, plotted for fourteen iterations of the implicit method of steps. Top right: plot of the associated lag function $t \mapsto t - r(x(t)) = h(t, x(t))$, with red dashed lines $[t_0, t_{k+1}] \times \{t_k\}$ for $t_k \in \mathcal{T}$ and $k = 0, \ldots, 3$. For visual clarity, only the first four such crossings are plotted. The inclusion condition $h(t\delta, \psi(t)) \in \mathcal{J}$ of Theorem 2 is satisfied because these lines do not fall below the curve $t \mapsto h(t, x(t))$. Bottom row: $(\eta, \mu) = (10, 1/10)$, otherwise same as top row. Note that for these parameter values, the technical condition $|r'(s)| \leq 1/(4a^2)$ for $|s| \leq 2a$ is violated severely, resulting in the striking non-monotonicity in the time lag functions.

for which a detailed analysis of connecting orbits has been completed [11]. The equation is a state-dependent modification of an earlier model of short-term fluctuations of exchange rates in a floating regime. In that work, $a > 0$ is a constant and the function r is assumed to be continuously differentiable, even, positive, bounded above with its maximum occuring at $r(0) = 1$, and satisfy $|r'(s)| \leq 1/(4a^2)$ for $|s| \leq 2a$. As a test problem, we will consider here the case with $a = 3$ and

$$r(x) = \exp(-|x| \sin^2(\eta x) - \mu x^2)$$

for a few choices of η and $\mu > 0$. We used Newton's method applied to $\overline{F}$. The initial data for Newton iteration was obtained by solving the boundary-value problem using MATLAB's `ddesd` solver, with `events` to handle the boundary condition (5). No other options were changed from default. MATLAB R2020a was used for all computations. We used $M = 40$ subdivisions for all iterations, and Newton's method was run until the numerical defect was at most 2×10^{-15}. We have made the codes available at [1]; running `script.m` will reproduce these figures.

References

[1] Kevin E.M. Church. Codes to accompany "An implicit method of steps for differential equations with state-dependent delay", https://github.com/kemchurch/state-dependent-delay.

[2] Jayme De Luca, Nicola Guglielmi, Tony Humphries, and Antonio Politi. Electromagnetic two-body problem : recurrent dynamics in the presence of state-dependent delay. *Journal of Physics A: Mathematical and Theoretical*, 43:205103, 2010.

[3] Philipp Getto and Marcus Waurick. A differential equation with state-dependent delay from cell population biology. *Journal of Differential Equations*, 260(7):6176–6200, 2016.

[4] David E. Gilsinn and Florian A. Potra. Integral operators and delay differential equations. *Journal of Integral Equations and Applications*, 18(3):297–336, sep 2006.

[5] Ferenc Hartung, Tibor Krisztin, Hans-Otto Walther, and Jianhong Wu. Functional Differential Equations with State-Dependent Delays: Theory and Applications. In *Handbook of differential equations: ordinary differential equations*, volume 3, pages 435–545. 2006.

[6] Tamás Insperger, Gábor Stépán, and Janos Turi. State-dependent delay in regenerative turning processes. *Nonlinear Dynamics*, 47(1-3):275–283, dec 2006.

[7] T. Kalmar-Nagy. A novel method for efficient numerical stability analysis of delay-differential equations. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 2823–2826. IEEE.

[8] Eva Kaslik and Seenith Sivasundaram. Analytical and numerical methods for the stability analysis of linear fractional delay differential equations. *Journal of Computational and Applied Mathematics*, 236(16):4027–4041, 2012.

[9] Jean-Philippe Lessard and J. D. Mireles James. A Rigorous Implicit C^1 Chebyshev Integrator for Delay Equations. *Journal of Dynamics and Differential Equations*, aug 2020.

[10] Umer Saeed and Muhammad Asad. Modified Chebyshev wavelet methods for fractional delay-type equations. *Applied Mathematics and Computation*, 264:431–442, 2015.

[11] Eugen Stumpf. On a Differential Equation with State-Dependent Delay: A Center-Unstable Manifold Connecting an Equilibrium and a Periodic Orbit. *Journal of Dynamics and Differential Equations*, 24(2):197–248, 2012.

[12] Marco Tosato, Xue Zhang, and Jianhong Wu. Multi-cycle Periodic Solutions of a Differential Equation with Delay that Switches Periodically. *Differential Equations and Dynamical Systems*, (0123456789), 2020.