Research Paper
Computational Methods for Differential Equations
http://cmde.tabrizu.ac.ir
Vol. *, No. *, *, pp. 1-20
DOI:10.22034/cmde.2026.73996.3804

Machine learning-guided selection of the decomposition length in the Laplace–Adomian decomposition method for Liouville–Caputo fractional logistic and Riccati equations

Sathish Marakonda[1,*] **and Y. Rajashekhar Reddy**[2]

[1]Research Scholar, Department of Mathematics, JNTU, Hyderabad, Telangana, India.

[2]Department of Mathematics, JNTU, Hyderabad, Telangana, India.

Abstract

We study the Laplace–Adomian decomposition method (LADM) for Liouville–Caputo fractional equations with a quadratic nonlinearity, focusing on the fractional logistic and Riccati equations, and the selection of its decomposition length M by machine learning. We show that the LADM approximation of order M is a partial sum of a power series in t^α, prove an *a priori* convergence theorem with an explicit error bound, and give a sharp convergence criterion with a computable *a posteriori* certificate. Certified truncation errors, cross-checked against a fractional Adams method, serve as training labels. Tuned regressors predict the logarithm of the truncation error from (α, M) with $R^2 = 0.9996$ for the logistic equation and 0.9987 for two Riccati equations, far sharper than the *a priori* bound (median: seven orders of magnitude, logistic case) and more accurate than a reference-free tail estimate; the minimal M meeting a tolerance is predicted with $R^2 \geq 0.99$. Including the training cost, the surrogate breaks even against full grid search after about 50 queries with a certified reference and about 500 with a fine-step Adams reference. Each equation requires its own dataset and model; adding (u_0, r) as features gives $R^2 = 0.99$ on unseen parameter configurations.

Keywords. Convergence analysis, Truncation error, Surrogate model, Gradient boosting, Power series.

2010 Mathematics Subject Classification. 34A08, 65L05, 65L20, 68T05.

Uncorrected

1. Introduction

The Adomian Decomposition Method (ADM) [2] and its Laplace-transform variant (LADM) [17] construct semi-analytical series solutions of nonlinear differential equations without linearization or discretization; the classical convergence theory of the method is due to Cherruault and Abbaoui [1, 8]. For fractional-order equations the method is paired with a fractional derivative whose Laplace transform is tractable. The Liouville–Caputo derivative [9, 23], originally considered by Liouville [24], is the standard choice, because its Laplace transform involves only the initial value $u(0)$.

Two nonlinear model problems are studied in this paper: the Liouville–Caputo fractional logistic equation

$$ {}^{LC}D_t^\alpha u(t) = r\,u(t)\big(1-u(t)\big), \qquad u(0)=u_0, \qquad 0<\alpha\leq 1, \tag{1.1}$$

and the fractional Riccati equation (Section 7). Both have been solved by a range of numerical and semi-analytical methods: Legendre-collocation schemes [14], a discretization approach [15], fractional-order Bessel and Legendre bases [16], a hybrid Fibonacci wavelet method [3], a perturbation–iteration approach [20] and a power-series solution [5] for the logistic equation; and the decomposition method [19] and, recently, multilayer-perceptron (MLP) solvers trained on the equation residual [21] for the Riccati equation. These works compute or approximate the solution. The present paper addresses a different, complementary question: *given* a semi-analytical series method, how should its numerical parameters be selected, and can a machine-learning surrogate make that selection cheaper than searching? In [21]

Received: 06 June 2026; Accepted: 15 November 2026.
∗ Corresponding author. Email: sathishmarakonda@gmail.com.

machine learning *is* the solver; here it is a surrogate for parameter selection inside an analytically understood solver, and the comparison in Section 7.3 quantifies the difference between the two uses.

The accuracy of a truncated LADM solution is governed by the truncation (decomposition) length M and, in the fractional setting, by the order α. Selecting M for a prescribed tolerance usually requires a convergence study or a search over candidate values for every α of interest. Following our earlier work on machine-learning-assisted parameter selection for decomposition methods [18], two complementary strategies are used: *search-then-learn*, in which a regressor predicts the minimal M that meets a tolerance for a given α, and *predict-and-skip*, in which a regressor predicts the achieved accuracy for a given pair (M, α) without evaluating the series or a reference solution.

Remark 1.1. We also considered the Atangana–Baleanu–Caputo (ABC) derivative [6]. Its Laplace transform carries an additional term proportional to $\mathcal{L}\{u\}$, and direct term-by-term decomposition of the ABC integral equation was found to violate the initial condition at every finite truncation order unless $\alpha = 1$. This is consistent with the known solvability subtleties of the ABC operator [25, 26], which motivated the modified ABC derivative [4]; even for that operator, closed-form treatment is available only for linear systems. We therefore use the Liouville–Caputo derivative, for which the LADM approximation satisfies the initial condition exactly at every order (Proposition 3.1), and leave the ABC case open.

The contributions of the paper are as follows.

(1) For the class of fractional equations with a quadratic nonlinearity, which contains the logistic and Riccati equations, the LADM approximation of order M is shown to be the degree-M partial sum of a power series in t^α (Proposition 3.1). On this basis we prove an *a priori* convergence theorem with a condition and an error bound that are explicit in the problem data (Theorem 3.5), and a sharp convergence criterion with a computable *a posteriori* certificate (Theorem 3.8).
(2) The training labels are certified truncation errors, and their accuracy is documented against an independent fractional Adams–Bashforth–Moulton solver whose observed convergence order is reported (Section 4).
(3) Five regressors (Random Forest, Gradient Boosting, support vector regression, k-nearest neighbours and polynomial least squares) are tuned by nested cross-validation and assessed by random, leave-α-out and repeated splits, by learning curves, under random and structured label noise, against the *a priori* error bound, and in a cost comparison that includes the training cost and reports break-even query counts (Section 6).
(4) The construction is repeated, with a new dataset and newly trained models of the same size, for two fractional Riccati equations (Section 7), and the feature set is extended to (α, M, u_0, r) (Section 8).

Section 2 recalls the fractional operators. Section 3 constructs the LADM series and proves its convergence. Section 4 describes the reference solution, and Section 5 the learning framework. Section 6 reports the results for the logistic equation, Section 7 those for the Riccati equations, and Section 8 the extended feature set. Section 9 discusses design choices and limitations, and Section 10 concludes.

Uncorrected Proof

2. Preliminaries

This section fixes the fractional operators used throughout the paper and records three elementary facts—the Laplace transform of the Liouville–Caputo derivative, the action of the Riemann–Liouville integral on power functions, and a monotonicity property of the resulting Gamma-function ratios—on which the convergence analysis of Section 3 rests.

Definition 2.1. For $\alpha > 0$ and $g \in L^1(0, T)$, the Riemann–Liouville fractional integral of order α is

$$I_t^\alpha g(t) = \frac{1}{\Gamma(\alpha)} \int_0^t (t - \tau)^{\alpha-1} g(\tau)\, d\tau. \tag{2.1}$$

Definition 2.2. Let $0 < \alpha < 1$. For an absolutely continuous function $u \in AC[0, T]$ (so that $u' \in L^1(0, T)$), the Liouville–Caputo fractional derivative of order α is the Riemann–Liouville integral of u',

$${}^{LC}D_t^\alpha u(t) = I_t^{1-\alpha}[u'](t) = \frac{1}{\Gamma(1 - \alpha)} \int_0^t (t - \tau)^{-\alpha} u'(\tau)\, d\tau. \tag{2.2}$$

More generally, for $u \in C[0,T]$ it is defined as ${}^{LC}D_t^\alpha u = \frac{d}{dt} I_t^{1-\alpha}[u - u(0)]$ whenever the right-hand side exists, which coincides with (2.2) for $u \in AC[0,T]$ [9, Ch. 3]; and ${}^{LC}D_t^1 u = u'$. The definition requires a derivative of u, whereas I_t^α only requires integrability. Definition 2.2, however, does *not* require $u \in C^1[0,T]$; this matters here, since the LADM terms $t^{n\alpha}$, and hence the solutions studied below, have an unbounded derivative at $t = 0$ when $0 < \alpha < 1$, although they are absolutely continuous. The operator was already considered by Liouville [24, p. 136], hence the name Liouville–Caputo derivative.

Definition 2.3. A function $u \in C[0,T]$ is a solution of the initial value problem ${}^{LC}D_t^\alpha u = f(u)$, $u(0) = u_0$, with f continuous, if it satisfies the Volterra integral equation $u = u_0 + I_t^\alpha[f(u)]$ on $[0,T]$. By [9, Ch. 6] this is equivalent to $u(0) = u_0$ and ${}^{LC}D_t^\alpha u = f(u)$ on $(0,T]$ in the sense of Definition 2.2.

Lemma 2.4 ([23])**.** *For $u \in AC[0,T]$ extended to $[0,\infty)$ with exponential growth, $\mathcal{L}\left\{{}^{LC}D_t^\alpha u\right\}(s) = s^\alpha U(s) - s^{\alpha-1}u(0)$, and $\mathcal{L}\left\{I_t^\alpha g\right\}(s) = s^{-\alpha}\mathcal{L}\left\{g\right\}(s)$ for locally integrable g of exponential growth, where $U = \mathcal{L}\left\{u\right\}$.*

Lemma 2.5 ([23])**.** *For $\alpha > 0$ and $\mu > -1$, $I_t^\alpha[t^\mu] = \dfrac{\Gamma(\mu+1)}{\Gamma(\mu+\alpha+1)}\, t^{\mu+\alpha}$.*

Lemma 2.6. *For $0 < \alpha \le 1$ let $\beta_n := \Gamma(n\alpha+1)/\Gamma((n+1)\alpha+1)$, $n \ge 0$. Then (β_n) is non-increasing and $\beta_0 = 1/\Gamma(\alpha+1)$.*

Proof. Let $h(x) = \ln\Gamma(x+1) - \ln\Gamma(x+\alpha+1)$ for $x \ge 0$. Then $h'(x) = \psi(x+1) - \psi(x+\alpha+1) \le 0$, because the digamma function ψ is increasing on $(0,\infty)$. Hence $\beta_n = e^{h(n\alpha)}$ is non-increasing in n. □

3. LADM construction and convergence analysis

We first construct the LADM series for the whole class of fractional equations with a quadratic right-hand side, which contains every equation studied in this paper, and then prove two convergence results: an *a priori* theorem with a condition and an error bound that are explicit in the problem data (Theorem 3.5), and a *sharp* characterization together with a computable *a posteriori* certificate (Theorem 3.8). The two results answer different questions—whether convergence can be guaranteed from the data alone, and exactly where the series converges—and their relation explains the behaviour observed in Section 6.

3.1. The LADM series.

Consider

$$ {}^{LC}D_t^\alpha u(t) = f(u(t)) := a_0 + a_1 u(t) + a_2 u(t)^2, \qquad u(0) = u_0, \qquad t \in [0,T],\ 0 < \alpha \le 1, \tag{3.1}$$

with $a_2 \neq 0$. The logistic equation (1.1) corresponds to $(a_0,a_1,a_2) = (0,r,-r)$ and the Riccati equations of Section 7 to $(1,0,-1)$ and $(1,2,-1)$. Solutions are understood in the sense of Definition 2.3, i.e. as continuous solutions of $u = u_0 + I_t^\alpha[f(u)]$. The Laplace transform is used only to *derive* the recursion below; every convergence statement is proved directly for this integral equation, so no growth or regularity assumption beyond continuity is needed. Applying the Laplace transform and Lemma 2.4 gives $U(s) = u_0/s + s^{-\alpha}\mathcal{L}\left\{f(u)\right\}(s)$. Writing $u = \sum_{n\ge0} u_n$ and $f(u) = \sum_{n\ge0} A_n$, where A_n are the Adomian polynomials

$$ A_n = \frac{1}{n!}\frac{d^n}{d\lambda^n} f\Big(\sum_{k=0}^{n} \lambda^k u_k\Big)\Big|_{\lambda=0} = a_0\delta_{n0} + a_1 u_n + a_2 \sum_{k=0}^{n} u_k u_{n-k}, \tag{3.2}$$

and matching terms after the inverse transform yields the LADM recursion

$$ u_0(t) = u_0, \qquad u_{n+1}(t) = I_t^\alpha[A_n](t), \qquad n \ge 0. \tag{3.3}$$

Proposition 3.1. *Every LADM term is a single power of t^α: $u_n(t) = c_n t^{n\alpha}$, where $c_0 = u_0$ and*

$$ c_{n+1} = \beta_n\Big(a_0\delta_{n0} + a_1 c_n + a_2 \sum_{k=0}^{n} c_k c_{n-k}\Big), \qquad n \ge 0, \tag{3.4}$$

with β_n as in Lemma 2.6. Consequently the truncated LADM solution $u^{(M)} := \sum_{n=0}^{M} u_n$ is the partial sum $u^{(M)}(t) = \sum_{n=0}^{M} c_n s^n$, $s = t^\alpha$, of the power series $\Phi(s) = \sum_{n\ge0} c_n s^n$, and $u^{(M)}(0) = u_0$ for every $M \ge 0$.

Uncorrected Proof

Proof. By induction: if $u_k = c_k t^{k\alpha}$ for $k \le n$, then $A_n = (a_0\delta_{n0} + a_1 c_n + a_2 \sum_k c_k c_{n-k}) t^{n\alpha}$, and Lemma 2.5 with $\mu = n\alpha$ gives $u_{n+1} = c_{n+1} t^{(n+1)\alpha}$ with (3.4). Since $u_n(0) = 0$ for $n \ge 1$, $u^{(M)}(0) = u_0$. □

Lemma 3.2 (Shift invariance)**.** *Let $c_* = -a_1/(2a_2)$, $w_0 = u_0 - c_*$, $A_0 = f(c_*)$ and $A_2 = a_2$, so that $f(c_* + w) = A_0 + A_2 w^2$. Let (e_n) be the LADM coefficients of ${}^{LC}D_t^\alpha w = A_0 + A_2 w^2$, $w(0) = w_0$. Then $e_0 = c_0 - c_*$ and $e_n = c_n$ for $n \ge 1$; that is, the LADM approximations of u and of $w = u - c_*$ differ exactly by the constant c_* at every truncation order.*

Proof. By (3.4), $e_1 = \beta_0(A_0 + A_2 w_0^2) = \beta_0 f(u_0) = c_1$. Assume $e_k = c_k$ for $1 \le k \le n$ ($n \ge 1$). In $\sum_{k=0}^n e_k e_{n-k}$ only the terms $k = 0$ and $k = n$ differ from those of $\sum_k c_k c_{n-k}$, each by $-c_* c_n$; hence $A_2 \sum_k e_k e_{n-k} = a_2 \sum_k c_k c_{n-k} - 2a_2 c_* c_n = a_2 \sum_k c_k c_{n-k} + a_1 c_n$, and $e_{n+1} = c_{n+1}$. □

Proposition 3.3 (Scaling)**.** *Replacing (a_0, a_1, a_2) by $\lambda(a_0, a_1, a_2)$, $\lambda > 0$, multiplies c_n by λ^n. Consequently, the LADM approximation of (3.1) on $[0, T]$ coincides, after the change of variable $t = T\tau$, with that of the problem with right-hand side $T^\alpha f$ on $[0, 1]$. In particular, for the logistic equation (1.1) the truncation error $\|u - u^{(M)}\|_{\infty,[0,T]}$ depends on (r, T) only through rT^α.*

Proof. The first claim follows from (3.4) by induction (c_{n+1} is a sum of products of λ with terms of total degree n). Hence $\sum_n c_n(\lambda f)\tau^{n\alpha} = \sum_n c_n(f)(\lambda^{1/\alpha}\tau)^{n\alpha}$; with $\lambda = T^\alpha$ and $t = T\tau$ the two partial sums coincide. □

Remark 3.4. For the logistic equation with $u_0 = \frac{1}{2}$ one has $w_0 = 0$ in Lemma 3.2, and (3.4) for (e_n) shows inductively that $e_{2k} = 0$ for $k \ge 1$. Thus $c_{2k} = 0$ and $u^{(2k)} = u^{(2k-1)}$: two consecutive truncation lengths always give the same approximation. To avoid this degenerate, duplicated information in the training data, the main computations of this paper use $u_0 = 0.4$ (Section 6).

3.2. **A priori convergence and an explicit error bound.**

Theorem 3.5. *Let c_*, w_0, A_0, A_2 be as in Lemma 3.2 and let $x_* > 0$ be the positive root of*

$$4\,|A_2|\,|A_0|\,x^2 + 4\,|A_2|\,|w_0|\,x - 1 = 0. \tag{3.5}$$

($x_* = 1/(4|A_2||w_0|)$ if $A_0 = 0$; the case $A_0 = w_0 = 0$ is excluded, since then $u \equiv u_0$ is an equilibrium and $u^{(M)} = u_0$ is exact for every M). Define

$$\gamma_0 := \frac{T^\alpha}{\Gamma(\alpha + 1)\, x_*}. \tag{3.6}$$

If $\gamma_0 < 1$, then:

(i) *the LADM series $\sum_n u_n(t)$ converges absolutely and uniformly on $[0, T]$;*
(ii) *its sum u is continuous on $[0, T]$ and is the unique continuous solution of $u = u_0 + I_t^\alpha[f(u)]$, i.e. the unique solution of (3.1) in the sense of Definition 2.3; moreover $u \in AC[0, T]$;*
(iii) *for every $M \ge 0$,*

$$\left\|u - u^{(M)}\right\|_{\infty,[0,T]} \le \frac{1}{2\,|A_2|\,x_*}\,\frac{\gamma_0^{M+1}}{1 - \gamma_0}. \tag{3.7}$$

Proof. Step 1 (reduction). By Lemma 3.2, $u - u^{(M)} = \sum_{n>M} e_n t^{n\alpha}$, so it suffices to study (e_n), which satisfies $e_0 = w_0$ and $e_{n+1} = \beta_n(A_0\delta_{n0} + A_2 \sum_k e_k e_{n-k})$.

Step 2 (majorant). Let $\kappa = \beta_0 = 1/\Gamma(\alpha + 1)$ and define $m_0 = |w_0|$, $m_{n+1} = |A_0|\,\delta_{n0} + |A_2| \sum_{k=0}^n m_k m_{n-k}$. Since $\beta_n \le \kappa$ (Lemma 2.6), induction gives $|e_n| \le \kappa^n m_n$. The generating function $\mathcal{M}(x) = \sum_n m_n x^n$ satisfies, as a formal power series, $\mathcal{M} = |w_0| + x(|A_0| + |A_2|\,\mathcal{M}^2)$. The function

$$\widetilde{\mathcal{M}}(x) = \frac{1 - \sqrt{\Delta(x)}}{2\,|A_2|\,x}, \qquad \Delta(x) = 1 - 4\,|A_2|\,x\big(\,|w_0| + |A_0|\,x\big), \tag{3.8}$$

is analytic in $|x| < x_*$: if $A_0 \ne 0$, the roots of Δ are $x_* > 0$ and a root $x_- \le -x_*$ (their product is $-1/(4|A_2||A_0|) < 0$ and their sum $-|w_0|/|A_0| \le 0$); if $A_0 = 0$, x_* is the only root. Hence $\Delta \ne 0$ in the disc $|x| < x_*$, and the branch of $\sqrt{\Delta}$

Uncorrected Proof

with $\sqrt{\Delta(0)} = 1$ is analytic there. It satisfies the same quadratic identity and $\widetilde{\mathcal{M}}(0) = |w_0|$, and since this identity determines the Taylor coefficients recursively, $\widetilde{\mathcal{M}}(x) = \sum_n m_n x^n$ for $|x| < x_*$. As $m_n \geq 0$, monotone convergence gives $\sum_n m_n x_*^n = \lim_{x\uparrow x_*} \widetilde{\mathcal{M}}(x) = 1/(2|A_2|x_*)$, hence

$$m_n x_*^n \leq \frac{1}{2\,|A_2|\,x_*} \qquad (n \geq 0). \tag{3.9}$$

Step 3 (bound). For $t \in [0,T]$, $|e_n t^{n\alpha}| \leq \kappa^n m_n T^{n\alpha} = m_n x_*^n \gamma_0^n \leq \gamma_0^n/(2|A_2|x_*)$ by (3.9). The Weierstrass M-test gives (i), and summing the geometric tail over $n > M$ gives (3.7).

Step 4 (the sum solves the problem). Let $W(t) = \sum_n e_n t^{n\alpha}$, so $u = c_* + W$. Because the series converges absolutely and uniformly, its Cauchy square converges uniformly to W^2, and I_t^α is a bounded operator on $C[0,T]$ ($\|I_t^\alpha g\|_\infty \leq T^\alpha \|g\|_\infty/\Gamma(\alpha+1)$). Integrating term by term and using Lemma 2.5, $I_t^\alpha[A_0 + A_2 W^2] = \sum_{n\geq 0} e_{n+1} t^{(n+1)\alpha} = W - w_0$, that is, $u = u_0 + I_t^\alpha[f(u)]$. Since u is continuous and f is locally Lipschitz, this Volterra equation has at most one continuous solution on $[0,T]$ [9, Ch. 6], which proves (ii) except for the regularity claim. For the latter, each term $c_n t^{n\alpha}$ is absolutely continuous with $\int_0^T \left|\frac{d}{dt} c_n t^{n\alpha}\right| dt = |c_n| T^{n\alpha}$ for $n \geq 1$, and $\sum_n |c_n| T^{n\alpha} < \infty$ by Step 3. Hence the differentiated series converges in $L^1(0,T)$, and $u(t) = u_0 + \int_0^t \sum_{n\geq 1} n\alpha c_n \tau^{n\alpha-1} d\tau$, so $u \in AC[0,T]$ (in general $u \notin C^1[0,T]$, because $u'(t) \sim \alpha c_1 t^{\alpha-1}$ as $t \to 0^+$ when $c_1 \neq 0$). No *a priori* bound on the solution is assumed: boundedness of u is a consequence of Step 3. □

Corollary 3.6. *For the logistic equation (1.1) ($A_0 = r/4$, $A_2 = -r$, $w_0 = u_0 - \frac{1}{2}$), with $\theta(u_0) := \sqrt{1 + (2u_0 - 1)^2} - |2u_0 - 1|$,*

$$\gamma_0 = \frac{r\,T^\alpha}{\Gamma(\alpha+1)\,\theta(u_0)}, \qquad \left\|u - u^{(M)}\right\|_\infty \leq \frac{1}{2\theta(u_0)} \frac{\gamma_0^{M+1}}{1-\gamma_0}. \tag{3.10}$$

For the Riccati equation ${}^{LC}D_t^\alpha u = 1 - u^2$ ($A_0 = 1$, $A_2 = -1$, $w_0 = u_0$), $\gamma_0 = 2T^\alpha/\big(\Gamma(\alpha+1)(\sqrt{1+u_0^2} - |u_0|)\big)$ and the constant in (3.7) is $1/(\sqrt{1+u_0^2} - |u_0|)$. For the Riccati equation ${}^{LC}D_t^\alpha u = 1 + 2u - u^2$ ($c_* = 1$, $A_0 = 2$, $A_2 = -1$, $w_0 = u_0 - 1$), with $\eta(u_0) := \sqrt{(u_0-1)^2 + 2} - |u_0 - 1|$, $\gamma_0 = 4T^\alpha/\big(\Gamma(\alpha+1)\,\eta(u_0)\big)$ and the constant in (3.7) is $2/\eta(u_0)$.

Remark 3.7. The convergence parameter $LT^\alpha/\Gamma(\alpha+1)$ often used for decomposition methods, with L a Lipschitz constant of f on an assumed range of u, is a contraction constant for the *Picard* map $u \mapsto u_0 + I_t^\alpha[f(u)]$; the partial sums of the LADM series are not Picard iterates, so that constant does not control $u - u^{(M)}$, and it presupposes a range of u that has to be established separately. Theorem 3.5 replaces it by γ_0, which is computed from the data alone, refers to the LADM partial sums themselves, and needs no *a priori* bound on u.

3.3. **Sharp convergence region and an *a posteriori* certificate.** Condition $\gamma_0 < 1$ is sufficient but not necessary: the majorant in Step 2 replaces every β_n by β_0, whereas β_n decays like $n^{-\alpha}$. The exact region follows from Proposition 3.1.

Theorem 3.8. *Let $R_\alpha := 1/\limsup_n |c_n|^{1/n} \in (0,\infty]$ be the radius of convergence of $\Phi(s) = \sum_n c_n s^n$ and $q_\alpha := T^\alpha/R_\alpha$. ($R_\alpha \geq \Gamma(\alpha+1)\,x_* > 0$, because $|c_n| = |e_n| \leq \kappa^n m_n$ for $n \geq 1$ and $m_n x_*^n$ is bounded by Step 2 of the proof of Theorem 3.5.)*

(a) *If $q_\alpha < 1$, the LADM series converges absolutely and uniformly on $[0,T]$, its sum is the unique continuous solution of (3.1), and for every $q \in (q_\alpha, 1)$ there is $C_q < \infty$ with $\left\|u - u^{(M)}\right\|_\infty \leq C_q\, q^{M+1}/(1-q)$. If $q_\alpha > 1$, the terms $u_n(T)$ are unbounded and the LADM series diverges at $t = T$.*

(b) *(Certificate.) Let $N \geq 1$, let $d_n = |c_n|$ for $n \leq N$, and let $\mathcal{D}_N$ be the solution of*

$$\beta_N\,|a_2|\;s\,\mathcal{D}^2 + (\beta_N\,|a_1|\,s - 1)\mathcal{D} + G_N(s) = 0,$$
$$G_N(s) = P_N(s) - \beta_N s\big(\,|a_1|\,P_{N-1}(s) + |a_2|\,Q_{N-1}(s)\big), \tag{3.11}$$

with $\mathcal{D}_N(0) = d_0$, where $P_k(s) = \sum_{n\leq k} d_n s^n$ and Q_{N-1} is P_{N-1}^2 truncated at degree $N-1$. Let σ_N be the first positive point at which $\Delta_N(s) = (1 - \beta_N|a_1|s)^2 - 4\beta_N|a_2|sG_N(s)$ vanishes or $1 - \beta_N|a_1|s$ vanishes. Then

Uncorrected Proof

$R_\alpha \geq \sigma_N$, *and for* $T^\alpha < \sigma_N$ *and all* $M \geq N$,

$$\left\| u - u^{(M)} \right\|_\infty \leq \mathcal{D}_N(T^\alpha) - P_M(T^\alpha), \qquad P_M(s) := P_N(s) + \sum_{N<n\leq M} d_n s^n. \tag{3.12}$$

Proof. (a) By the Cauchy–Hadamard theorem the power series Φ converges absolutely and uniformly on $[0, T^\alpha]$ when $T^\alpha < R_\alpha$ and its terms are unbounded at $s = T^\alpha$ when $T^\alpha > R_\alpha$. For $q \in (q_\alpha, 1)$, $|c_n|\,(T^\alpha/q)^n$ is bounded, by C_q say, since $T^\alpha/q < R_\alpha$, which gives the error bound. That the sum solves (3.1) in the sense of Definition 2.3 and is absolutely continuous follows exactly as in Step 4 of the proof of Theorem 3.5, since $\sum_n |c_n| T^{n\alpha} < \infty$ for $T^\alpha < R_\alpha$.

(b) Define $d_{n+1} = \beta_N(|a_1|\,d_n + |a_2| \sum_{k=0}^{n} d_k d_{n-k})$ for $n \geq N$. Since $\beta_n \leq \beta_N$ for $n \geq N$ (Lemma 2.6) and a_0 enters (3.4) only at $n = 0$, induction gives $|c_n| \leq d_n$ for all n. Multiplying the recursion by s^{n+1} and summing over $n \geq N$ shows that $\mathcal{D}(s) = \sum_n d_n s^n$ satisfies the quadratic equation in the statement as a formal power series, and that this equation determines its coefficients uniquely; the root $\mathcal{D}_N = \big((1 - \beta_N|a_1|s) - \sqrt{\Delta_N}\big)/(2\beta_N|a_2|s)$ has the same Taylor coefficients and is analytic on a neighbourhood of $[0, \sigma_N)$. If the radius ρ of $\mathcal{D}$ were smaller than σ_N, Pringsheim's theorem (non-negative coefficients) would make $s = \rho$ a singular point of $\mathcal{D}$, whereas $\mathcal{D}_N$ continues $\mathcal{D}$ analytically across ρ—a contradiction. Hence $R_\alpha \geq \rho \geq \sigma_N$, $\mathcal{D}(T^\alpha) = \mathcal{D}_N(T^\alpha)$ for $T^\alpha < \sigma_N$, and $\left|u - u^{(M)}\right| \leq \sum_{n>M} d_n T^{n\alpha} = \mathcal{D}_N(T^\alpha) - P_M(T^\alpha)$. □

Remark 3.9. Theorem 3.8 identifies $q_\alpha = T^\alpha/R_\alpha$ as the exact convergence parameter: $q_\alpha < 1$ is necessary and sufficient (up to the boundary case). It is not available in closed form, but $q_{\text{cert}} := T^\alpha/\sigma_N \geq q_\alpha$ is a rigorous upper bound computable from $c_0, \ldots, c_N$ (its numerical evaluation is validated in interval arithmetic; Section 4), and in all our computations σ_N increased with N (for example $\sigma_N = 2.97, 3.06, 3.10, 3.12$ for $N = 50, 100, 200, 400$ in the case below); convergence of σ_N to R_α is not proved here. For $\alpha = 1$, $r = 1$, $u_0 = \frac{1}{2}$, where $R_1 = \pi$ is known from the poles of $1/(1 + e^{-t})$, $N = 200$ gives $\sigma_N = 3.10$, i.e. q_{cert} exceeds q_1 by about 1.3%; for small α, where β_N decays slowly, the approach is slower and q_{cert} is more conservative. In the tables we therefore report both γ_0 (explicit, sufficient) and q_{cert} (certified); a case with $\gamma_0 > 1$ and $q_{\text{cert}} < 1$ is convergent and is *not* in contradiction with the theory.

4. Reference solution and accuracy of the training labels

The machine-learning targets are truncation errors, so their accuracy is only as good as the reference solution. This section states how the reference is obtained and how its accuracy was verified independently.

4.1. Certified series reference.

For each equation and each α the coefficients $c_0, \ldots, c_N$ of Proposition 3.1 are computed, with N doubled from 64 until the certificate (3.12) of Theorem 3.8 bounds the neglected remainder $\sum_{n>N} |c_n|\, T^{n\alpha}$ by $\max(10^{-30}, 10^{-6} E(40, \alpha))$. For the logistic dataset $N \in \{64, 128\}$ sufficed; for the Riccati datasets $N \leq 512$.

Validated certificate. The certificate was verified as a computer-assisted proof. The coefficients c_n and the factors β_n are enclosed in interval arithmetic (60 digits), and d_n is taken as the upper end of the enclosure of $|c_n|$, rounded up. The positivity of Δ_N on $[0, T^\alpha]$ is *proved*, not sampled: $[0, T^\alpha]$ is divided into cells, and on each cell $[a, b]$ the polynomials P_N, P_{N-1} and Q_{N-1}, which have non-negative coefficients and are therefore increasing for $s \geq 0$, are bounded by their values at the cell ends. This gives

$$\Delta_N(s) \;\geq (1 - \beta_N|a_1|b)^2 - 4\beta_N|a_2|\,b\, \max\big\{0,\; P_N(b) - \beta_N a\big(|a_1| P_{N-1}(a) + |a_2| Q_{N-1}(a)\big)\big\}, \quad (s \in [a, b]), \tag{4.1}$$

evaluated with rigorous rounding-error bounds, and cells on which the right-hand side is not positive are bisected. A zero of Δ_N between sample points therefore cannot be missed. The remainder bound $\mathcal{D}_N(T^\alpha) - P_N(T^\alpha)$ is evaluated in interval arithmetic. For all 144 values of α in the three datasets the certificate holds, and the neglected remainder is at most $8.5 \times 10^{-7} E(40, \alpha)$. The convergence factors q_{cert} in the tables and figures are validated in the same way: with $N = 256$, positivity of Δ_{256} is proved on $[0, 0.99\,\sigma]$, where σ is the value of σ_{256} located numerically, and $q_{\text{cert}} := T^\alpha/(0.99\,\sigma)$. For the logistic dataset $q_{\text{cert}} \leq 0.94$ (the largest value occurs at $\alpha = 0.05$, where the certificate is most conservative).

Labels. The training label is the truncation error $E(M, \alpha) = \sup_{t\in[0,T]} |u(t) - u^{(M)}(t)|$. It is computed from the tail of the series, which avoids the cancellation of subtracting two nearly equal numbers, as the maximum over the grid

Uncorrected Proof

$\mathcal{G} = \{jT/2000\}_{j=0}^{2000}$:

$$E_{\mathcal{G}}(M,\alpha) = \max_{t\in\mathcal{G}} \Big| \sum_{M<n\le N} c_n\, t^{n\alpha} \Big|. \tag{4.2}$$

Three error sources separate $E_{\mathcal{G}}$ from E, and each was bounded.

(i) Rounding: every stored label was recomputed in 40-digit arithmetic, with a maximum relative difference of 5×10^{-12}.
(ii) The neglected remainder is below $10^{-6}E(40,\alpha)$, as certified above.
(iii) The grid versus the whole interval: the supremum of $|\sum_{n>M} c_n s^n|$ over $s\in[0,T^\alpha]$ was enclosed for every label by a branch-and-bound search with second-order Taylor bounds, to a relative bracket width of 10^{-8}. It exceeds $E_{\mathcal{G}}$ by at most 2.9×10^{-5} (relative), with largest gap 7×10^{-6} for the logistic dataset.

Combining (ii) and (iii) (the remainder is at most 1.3×10^{-6} of every individual label), $(1-2\times10^{-6})\,E_{\mathcal{G}} \le E \le (1+3\times10^{-5})\,E_{\mathcal{G}}$ for every label. On the $\log_{10}$ scale used for learning, the difference is below 1.3×10^{-5}, far below the accuracy of any regressor, and the tables report $E_{\mathcal{G}}$. These labels are truncation errors of the exact LADM partial sums; they do not include the rounding error of evaluating $u^{(M)}$ in floating-point arithmetic.

4.2. **Independent check against a fractional Adams method.** The certified reference was compared with the fractional Adams–Bashforth–Moulton (ABM) predictor–corrector scheme [10] on uniform grids with $N = 250,\dots,8000$ steps (Table 1). The differences decrease monotonically; the observed order is $p\approx2\alpha$ for $0.3\le\alpha\le0.7$, slightly below 2α at $\alpha=0.1$ (where it is still increasing with N), $p\approx1.9$ at $\alpha=0.9$ and $p=2$ at $\alpha=1$. The two independent computations therefore agree to within the ABM discretization error. The reduced order for small α is expected: ${}^{LC}D_t^\alpha u = f(u(t))$ contains a t^α term and is therefore not smooth at $t=0$, and the convergence order of the method depends on that smoothness [11]. The table also quantifies the error of an ABM reference with a typical step such as $h=0.01$: at $\alpha=0.1$ the ABM error is about 3×10^{-4} even with 250 steps, and it decays very slowly. Labels below that level, if computed against such a reference, are dominated by the reference error. Section 6.4 measures the effect of such labels on the trained models.

TABLE 1. Independent check of the certified reference: maximum difference on $[0,1]$ between the ABM solution with N uniform steps ($h=1/N$) and the certified LADM reference (logistic equation, $r=0.5$, $u_0=0.4$), and the observed ABM order p (mean of the last three step halvings).

α	$N=250$	$N=500$	$N=1000$	$N=2000$	$N=4000$	$N=8000$	p
0.1	2.8×10^{-4}	2.5×10^{-4}	2.3×10^{-4}	2.1×10^{-4}	1.9×10^{-4}	1.7×10^{-4}	0.16
0.3	7.3×10^{-5}	4.8×10^{-5}	3.2×10^{-5}	2.1×10^{-5}	1.4×10^{-5}	9.1×10^{-6}	0.60
0.5	7.5×10^{-6}	3.7×10^{-6}	1.8×10^{-6}	9.2×10^{-7}	4.6×10^{-7}	2.3×10^{-7}	1.00
0.7	5.0×10^{-7}	1.9×10^{-7}	7.1×10^{-8}	2.7×10^{-8}	1.0×10^{-8}	3.8×10^{-9}	1.40
0.9	3.9×10^{-8}	1.0×10^{-8}	2.7×10^{-9}	7.3×10^{-10}	1.9×10^{-10}	5.1×10^{-11}	1.91
1.0	2.0×10^{-8}	4.9×10^{-9}	1.2×10^{-9}	3.1×10^{-10}	7.7×10^{-11}	1.9×10^{-11}	2.00

4.3. **Validation at $\alpha=1$.** At $\alpha=1$ all three equations have closed-form solutions. For the logistic equation ($r=0.5$, $u_0=0.4$) the certified series agrees with $u_0/(u_0+(1-u_0)e^{-rt})$ to 1.1×10^{-16} on $[0,1]$, and the $M=10$ approximation has error 1.8×10^{-10}; the ABM solution with $h=10^{-3}$ differs from the exact solution by 1.2×10^{-9}. For Riccati equation I the series agrees with $\tanh(t+\operatorname{arctanh}u_0)$ to 2.2×10^{-16} (the $M=12$ approximation: 2.1×10^{-3}), and for Riccati equation II with $1+\sqrt{2}\tanh\big(\sqrt{2}t+\frac{1}{2}\ln\frac{\sqrt{2}-1}{\sqrt{2}+1}\big)$ to 5.6×10^{-16} on $[0,0.5]$.

5. Machine learning framework

This section defines the two learning tasks, the datasets, the models and the validation protocol.

Uncorrected Proof

5.1. **Data.** For the logistic equation (1.1) the grid consists of $\alpha \in \{0.05, 0.07, \ldots, 0.99\}$ (48 values) and $M \in \{1, \ldots, 40\}$, i.e. 1920 certified labels $E(M, \alpha)$, with $r = 0.5$, $u_0 = 0.4$ and $T = 1$. These parameters were chosen so that Theorem 3.5 applies on the whole grid ($0.61 \le \gamma_0 \le 0.69$) and so that the degeneracy of Remark 3.4 is avoided. All tables of the logistic equation use the same interval $[0, 1]$. The grids of the Riccati equations have the same size and are described in Section 7.

5.2. **Search-then-learn (STL).** Because $E(M, \alpha)$ need not decrease monotonically in M (this is pronounced for the Riccati equations, whose series converge slowly near the boundary of the convergence region), the target is the *robust* minimal length

$$M^*(\alpha, \varepsilon) = \min \{M : \ E(M', \alpha) \le \varepsilon \text{ for all } M \le M' \le 40\}, \tag{5.1}$$

i.e. the shortest truncation from which every longer one also meets the tolerance. A regressor $\widehat{M}(\alpha, \log_{10} \varepsilon)$ is trained on 30 tolerances, log-uniformly spaced in $[10^{-14}, 10^{-2}]$ for the logistic equation; pairs with $E(40, \alpha) > \varepsilon$ are excluded. In use, the prediction is rounded up, $M = \lceil \widehat{M} \rceil$.

5.3. **Predict-and-skip (PAS).** A second regressor is trained on $(\alpha, M) \mapsto \log_{10} E(M, \alpha)$. It estimates the accuracy of a candidate pair without evaluating the series or a reference solution; the logarithm is used because E spans 31 orders of magnitude on the logistic grid.

5.4. **Models, hyperparameters and validation.** Five regressors are compared. Three are representatives of distinct regression paradigms: Random Forest [7] (bagged trees), Gradient Boosting [12] (boosted trees) and support vector regression with an RBF kernel (SVR, kernel methods). Two are classical baselines: k-nearest neighbours (kNN) and polynomial least squares. All models are implemented in scikit-learn [22]. Hyperparameters are selected by *nested* cross-validation: an outer shuffled 5-fold split (random seed 0) for assessment and, inside every outer training set, a 3-fold grid search minimizing the mean squared error over the grids of Table 2. We report the mean $\pm$ standard deviation over the outer folds of the MAE, RMSE and R^2, and for STL also the MAPE. Three further protocols are used: a leave-α-out split (`GroupKFold`, 6 folds), in which all samples of the held-out α values are unseen; 5×5 repeated cross-validation; and learning curves computed with shuffled folds.

TABLE 2. Hyperparameter grids searched by the inner cross-validation, and the configuration selected most often across the five outer folds for the logistic equation (PAS / STL). The software is scikit-learn 1.9.1 [22]; unlisted hyperparameters take their default values; random_state= 0.

Model	grid	selected (PAS)	selected (STL)
Random Forest	n_estimators $\in \{300\}$, max_depth $\in$ {None, 12}, min_samples_leaf $\in \{1, 2, 4\}$	max_depth=12, min_samples_leaf=1, n_estimators=300	max_depth=12, min_samples_leaf=1, n_estimators=300
Gradient Boosting	n_estimators $\in \{300, 1000\}$, max_depth $\in \{2, 3, 4\}$, learning_rate $\in \{0.05, 0.1\}$	learning_rate=0.1, max_depth=4, n_estimators=1000	learning_rate=0.1, max_depth=3, n_estimators=300
SVR	(standardized inputs, RBF) $C \in \{10, 100, 1000\}$, $\gamma \in \{0.3, 1, 3\}$, $\epsilon \in \{0.01, 0.05\}$	C=1000, epsilon=0.05, gamma=1	C=100, epsilon=0.05, gamma=0.3
kNN	(standardized inputs) $k \in \{3, 5, 8\}$, weights $\in$ {uniform, distance}	n_neighbors=8, weights=distance	n_neighbors=8, weights=distance
Polynomial LS	degree $\in \{2, 3, 4\}$ (least squares on standardized monomials)	degree=4	degree=4

6. NUMERICAL RESULTS FOR THE LOGISTIC EQUATION

This section first examines the convergence behaviour predicted by Theorems 3.5 and 3.8, and then evaluates the two learning tasks: accuracy, robustness, comparison with the analytical bound and with reference-free alternatives, and cost. Unless stated otherwise, $r = 0.5$, $u_0 = 0.4$ and the interval is $[0, 1]$.

Uncorrected Proof

6.1. **Convergence and the role of the two convergence parameters.** Table 3 lists the certified truncation error for $\alpha = 0.7$ together with the *a priori* bound (3.7). The error decays geometrically overall, though not monotonically: $E(2, 0.7)$ slightly exceeds $E(1, 0.7)$, and such non-monotone steps are frequent for the Riccati equations (Sections 5.2 and 7). The initial condition is reproduced exactly at every order, and the bound holds but is pessimistic by about 2 to 10 orders of magnitude, because γ_0 exceeds the certified factor q_{cert} (Remark 3.9). Over the whole 1920-point grid the bound was never violated, and $\log_{10}(\mathrm{bound}/E)$ ranged from 1.7 to 24.5 (median 7.6). The bound also held for 60 further random configurations with $\gamma_0 < 1$, 20 for each of the three equation types of Sections 6 and 7 with random α, u_0 and r or T (2400 certified errors $E(M, \alpha)$, $M = 1, \ldots, 40$); there the bound exceeded the error by at least a factor of 17. Figures 1 and 2 show the decay over the grid: convergence is faster for larger α, consistent with the dependence of $q_\alpha = T^\alpha / R_\alpha$ on α.

TABLE 3. Logistic equation, $\alpha = 0.7$, $r = 0.5$, $u_0 = 0.4$, interval $[0, 1]$: certified truncation error $E(M, \alpha) = \sup_{[0,1]} |u - u^{(M)}|$ (grid maximum $E_{\mathcal{G}}$, within 3×10^{-5} relative of the supremum, Section 4) and the *a priori* bound (3.7) ($\gamma_0 = 0.671$, $q_{\mathrm{cert}} = 0.262$). $u^{(M)}(0) = u_0$ exactly for every M.

M	$E(M, 0.7)$	bound (3.7)
1	4.71×10^{-3}	8.36×10^{-1}
2	4.95×10^{-3}	5.61×10^{-1}
4	2.32×10^{-4}	2.53×10^{-1}
6	1.01×10^{-5}	1.14×10^{-1}
8	3.47×10^{-7}	5.13×10^{-2}
12	9.94×10^{-10}	1.04×10^{-2}
16	1.19×10^{-11}	2.11×10^{-3}
20	6.41×10^{-14}	4.29×10^{-4}

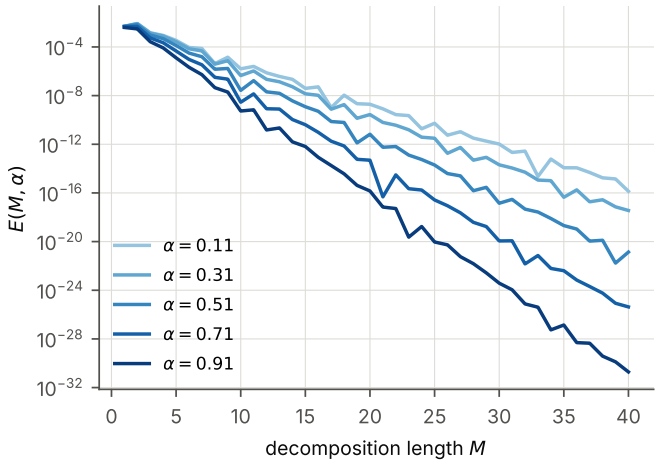

FIGURE 1. Certified truncation error $E(M, \alpha)$ of the logistic equation on $[0, 1]$ ($r = 0.5$, $u_0 = 0.4$) for representative α.

Figure 3 shows the truncated solutions ($M = 60$, certified convergent) for several α. Smaller α produces a faster initial rise followed by a slower approach to the carrying capacity, which is the familiar memory effect of the Liouville–Caputo derivative.

Tables 4 and 5 vary u_0 and r on the *same* interval $[0, 1]$, so all three tables can be compared directly; the row $u_0 = 0.4$ of Table 4 and the row $r = 0.5$ of Table 5 reproduce Table 3. Table 5 and Figure 4(a) also explain why the

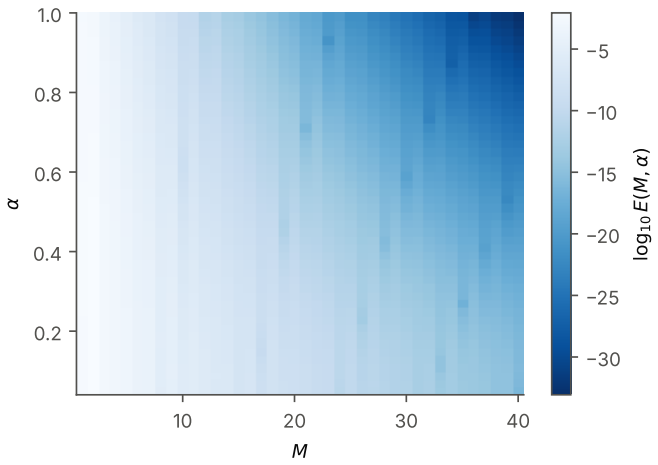

FIGURE 2. $\log_{10} E(M, \alpha)$ over the full grid (48 values of α, $M = 1, \ldots, 40$; $[0, 1]$, $r = 0.5$, $u_0 = 0.4$).

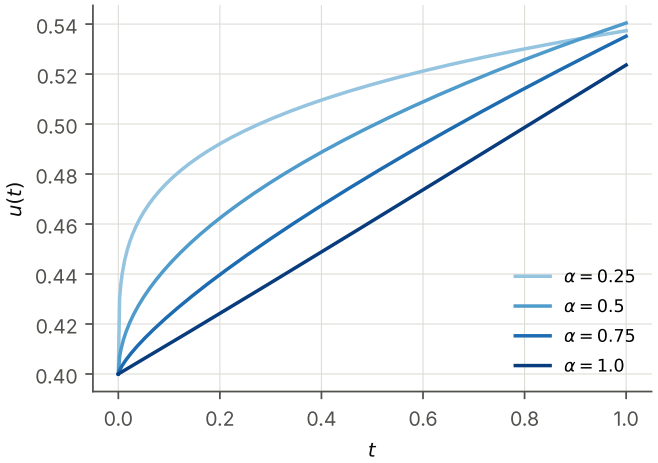

FIGURE 3. LADM solutions of the logistic equation on $[0, 1]$ ($r = 0.5$, $u_0 = 0.4$, $M = 60$) for several α.

error can decrease although the explicit convergence parameter exceeds one. For $r = 1$ we have $\gamma_0 > 1$, so Theorem 3.5 does not apply, but $q_{\text{cert}} < 1$: by Theorem 3.8 the series converges, and the errors decrease as observed. Condition $\gamma_0 < 1$ is sufficient, not necessary, and the table is consistent with the theory. For $r \geq 2$, convergence is not certified ($q_{\text{cert}} > 1$), and the numerical root-test estimate $\hat{q}$ also exceeds one, which indicates divergence. For $r = 2$ ($\hat{q} \approx 1.02$, just above one) the error does not decrease with M: between $M = 10$ and $M = 40$ it oscillates irregularly, with isolated dips (1.3×10^{-2} at $M = 14$, 2×10^{-3} at $M = 25$) between peaks that grow slowly from 0.20 to 0.27 (Figure 4(a)); the small entry $E(14, 0.7)$ in Table 5 is one of these dips and not a sign of convergence. For $r = 4, 8$ the errors grow rapidly. Since q_{cert} is only an upper bound for q_α and $\hat{q}$ is a finite-N estimate, this is numerical evidence of divergence, not a proof. Table 4 also contains two cases ($u_0 = 0.1, 0.9$) with $\gamma_0 > 1$ and $q_{\text{cert}} < 1$. Figure 4(b) shows γ_0 and q_{cert} as functions of α for all three equations studied.

TABLE 4. Logistic equation, $\alpha = 0.7$, $r = 0.5$, interval $[0, 1]$: truncation error for varying u_0 (same interval and parameters as Table 3). γ_0: *a priori* parameter (3.6); q_{cert}: certified parameter of Theorem 3.8.

u_0	γ_0	q_{cert}	$E(2, 0.7)$	$E(6, 0.7)$	$E(10, 0.7)$	$E(14, 0.7)$
0.1	1.14	0.23	2.63×10^{-3}	3.62×10^{-6}	3.26×10^{-9}	4.45×10^{-12}
0.25	0.89	0.25	1.41×10^{-3}	1.30×10^{-5}	1.42×10^{-8}	1.44×10^{-10}
0.4	0.67	0.26	4.95×10^{-3}	1.01×10^{-5}	2.91×10^{-9}	1.31×10^{-10}
0.5	0.55	0.26	5.07×10^{-3}	1.67×10^{-5}	6.43×10^{-8}	2.59×10^{-10}
0.6	0.67	0.26	3.46×10^{-3}	2.52×10^{-6}	2.54×10^{-8}	2.02×10^{-10}
0.75	0.89	0.25	4.43×10^{-4}	1.14×10^{-5}	4.54×10^{-9}	1.28×10^{-10}
0.9	1.14	0.23	2.39×10^{-3}	3.92×10^{-6}	5.78×10^{-9}	4.83×10^{-12}

TABLE 5. Logistic equation, $\alpha = 0.7$, $u_0 = 0.4$, interval $[0, 1]$: truncation error for varying r. $\gamma_0 < 1$ is sufficient for convergence; $q_{\text{cert}} < 1$ certifies convergence; $\hat{q}$ is the root-test estimate of q_α. †Not certified to converge ($q_{\text{cert}}, \hat{q} > 1$); errors for these cases are measured against an ABM reference with $h = 10^{-4}$.

r	γ_0	q_{cert}	$\hat{q}$	$E(2, 0.7)$	$E(6, 0.7)$	$E(10, 0.7)$	$E(14, 0.7)$
0.25	0.34	0.13	0.13	5.91×10^{-4}	6.56×10^{-8}	2.53×10^{-12}	4.83×10^{-15}
0.5	0.67	0.26	0.25	4.95×10^{-3}	1.01×10^{-5}	2.91×10^{-9}	1.31×10^{-10}
1	1.34	0.52	0.51	3.98×10^{-2}	1.58×10^{-3}	3.49×10^{-5}	2.36×10^{-6}
2	2.68	1.05	1.02	2.58×10^{-1}	2.02×10^{-1}	1.24×10^{-1}	$1.30 \times 10^{-2\dagger}$
4	5.37	2.10	2.04	1.14×10^{0}	1.72×10^{1}	2.23×10^{2}	$1.75 \times 10^{3\dagger}$
8	10.74	4.19	4.07	4.02×10^{0}	1.12×10^{3}	2.70×10^{5}	$4.49 \times 10^{7\dagger}$

Typeset Proof

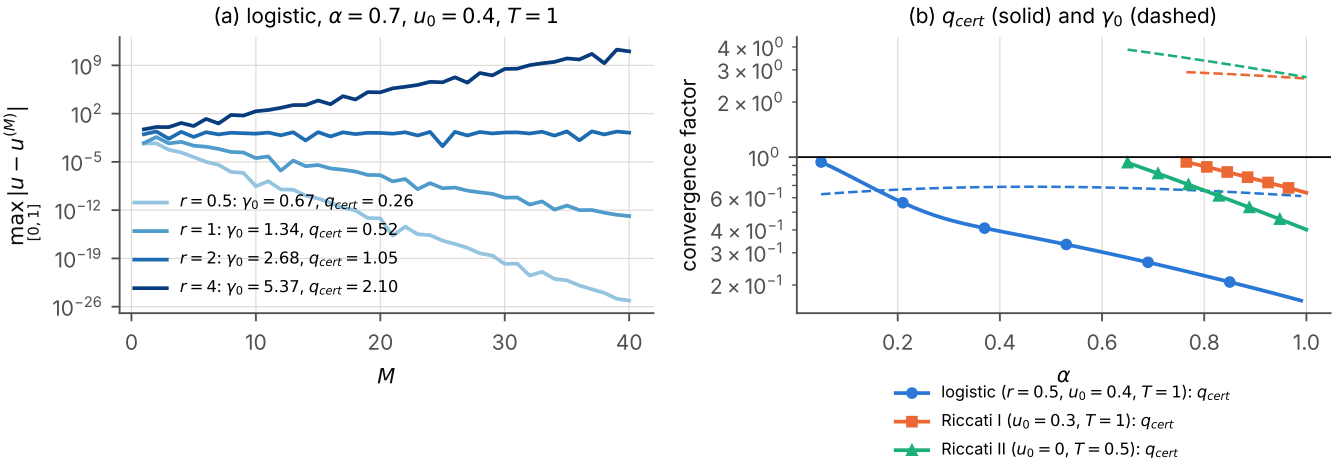

FIGURE 4. (a) Error of the LADM approximation of the logistic equation ($\alpha = 0.7$, $u_0 = 0.4$, $[0, 1]$) for several r; certified reference for $r \leq 1$, ABM reference with $h = 10^{-4}$ for the cases $r \geq 2$, which are not certified to converge. (b) Certified convergence factor q_{cert} (solid) and *a priori* parameter γ_0 (dashed) versus α for the three datasets; values below the horizontal line certify convergence.

Remark 6.1. The learning datasets lie entirely inside the certified convergence region ($q_{\mathrm{cert}} < 1$). Extending the regressors to parameter ranges that cross $q_\alpha = 1$ would require a divergence-aware target, for example the predicted value of q_α itself, and is left for future work.

6.2. **Accuracy of the regressors.** Table 6 reports the nested cross-validated performance. For predict-and-skip, Gradient Boosting is the most accurate model ($R^2 = 0.9996$, MAE 0.079 decades), followed by Random Forest ($R^2 = 0.9992$). The smooth models (SVR, kNN, degree-4 polynomial) reach $R^2 \approx 0.997$ with a three to four times larger MAE. For search-then-learn, all five models perform similarly ($R^2 = 0.9959$–0.9967, MAE 0.32–0.43 in units of M). The robust target $M^*(\alpha, \varepsilon)$ is a smooth staircase in $(\alpha, \log_{10}\varepsilon)$, and even a degree-4 polynomial fits it well. Figure 5 shows the out-of-fold predictions.

TABLE 6. Logistic equation: nested 5-fold cross-validated performance (mean ± std over outer folds; hyperparameters tuned in an inner 3-fold grid search, Table 2).

	Predict-and-skip ($n = 1920$, $\log_{10} E$ units)			Search-then-learn ($n = 1440$, units of M)			
Model	MAE	RMSE	R^2	MAE	RMSE	MAPE (%)	R^2
Random Forest	0.109 ± 0.005	0.186 ± 0.017	0.9992 ± 0.0001	0.32 ± 0.02	0.49 ± 0.03	2.5	0.9965 ± 0.0004
Gradient Boosting	0.079 ± 0.008	0.138 ± 0.022	0.9996 ± 0.0001	0.41 ± 0.02	0.52 ± 0.01	4.0	0.9961 ± 0.0001
SVR	0.249 ± 0.013	0.385 ± 0.025	0.9967 ± 0.0004	0.40 ± 0.01	0.49 ± 0.02	4.7	0.9966 ± 0.0002
kNN	0.301 ± 0.011	0.400 ± 0.022	0.9965 ± 0.0004	0.43 ± 0.02	0.53 ± 0.02	6.7	0.9959 ± 0.0002
Polynomial LS	0.267 ± 0.015	0.364 ± 0.030	0.9971 ± 0.0004	0.39 ± 0.01	0.48 ± 0.02	5.3	0.9967 ± 0.0002

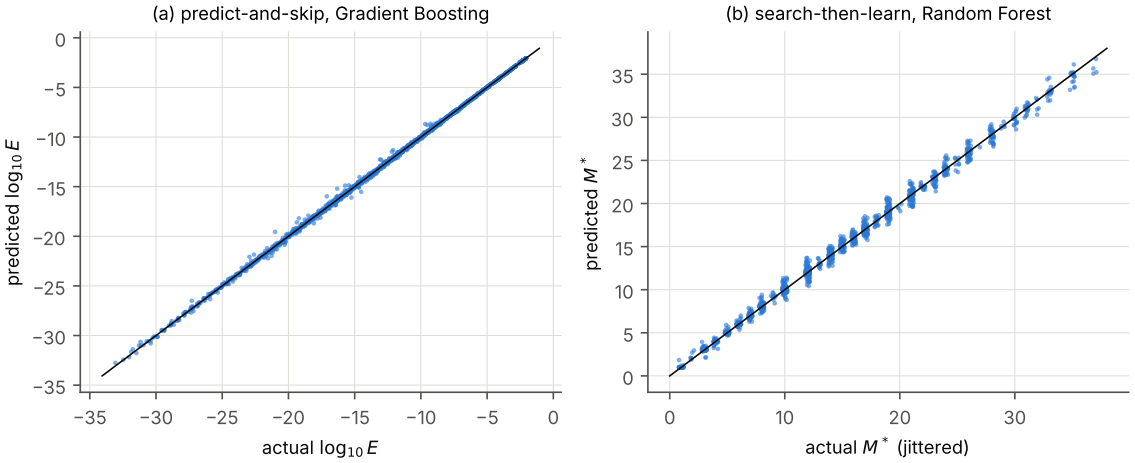

FIGURE 5. Out-of-fold predictions on the logistic dataset: (a) predict-and-skip (Gradient Boosting), (b) search-then-learn (Random Forest; the actual M^* is jittered horizontally for visibility).

6.3. **Leave-α-out, repeated cross-validation and learning curves.** Random fold assignment places samples of the same α in training and test sets. Table 7 therefore also reports a leave-α-out split, in which entire values of α are held out: R^2 remains above 0.996 for every model on both logistic tasks, so the regressors interpolate in α rather than memorize it. Five-times repeated 5-fold cross-validation gives $R^2 = 0.99961 \pm 0.00009$ and MAE 0.077 ± 0.006 for Gradient Boosting on predict-and-skip, and $R^2 = 0.9965 \pm 0.0006$ for Random Forest on search-then-learn, consistent with Table 6.

Figure 6 shows learning curves computed with shuffled folds and the tuned hyperparameters. On the logistic data, the validation R^2 of Gradient Boosting is already 0.979 with 76 training samples, exceeds 0.998 from about 470 samples

Proof

TABLE 7. Leave-α-out validation (6-fold `GroupKFold`: all samples of a held-out α are unseen in training), R^2 with the tuned hyperparameters of Table 2.

	Random Forest	Gradient Boosting	SVR	kNN	Polynomial LS
logistic PAS	0.9987	0.9986	0.9969	0.9981	0.9971
logistic STL	0.9971	0.9961	0.9967	0.9968	0.9967
Riccati I PAS	0.9970	0.9974	0.9538	0.9750	0.9599
Riccati I STL	0.9914	0.9907	0.9957	0.9925	0.9956
Riccati II PAS	0.9967	0.9968	0.9835	0.9965	0.9866
Riccati II STL	0.9907	0.9879	0.9954	0.9934	0.9901

and reaches 0.999 at about 740; the training R^2 stays close to 1, and the gap closes as the sample size grows. Shuffling matters here: with unshuffled folds, each validation fold would be a contiguous block of α values, and the curve would measure extrapolation in α rather than learning; generalization to unseen α is assessed separately by the leave-α-out split (Table 7).

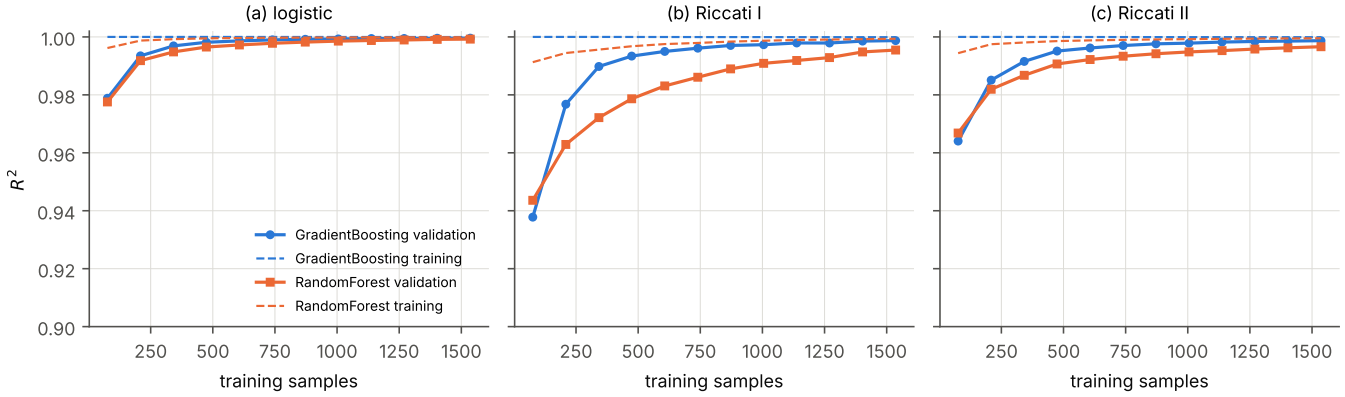

FIGURE 6. Learning curves (predict-and-skip, shuffled 5-fold CV, tuned hyperparameters) for the three equations: validation (solid) and training (dashed) R^2.

6.4. **Robustness to label errors.** Table 8 (top) adds i.i.d. Gaussian noise to the training targets. Performance degrades gracefully ($R^2 = 0.9992$ at $\sigma = 0.2$ decades). Reference errors are not i.i.d., however: they are deterministic and correlated in (α, M). The bottom part of Table 8 therefore trains on labels computed against ABM references with step h and scores the predictions against the certified labels. With $h = 0.01$, 84% of the labels are wrong by more than a factor of two, because no label can fall below the ABM error floor, and the trained model inherits the floor ($R^2 < 0$ on the full range). Restricted to errors above 10^{-4}, the same model is accurate ($R^2 = 0.998$, $n = 212$), but already for errors above 10^{-6} it degrades to $R^2 = 0.65$. The structured test is therefore far more severe than i.i.d. noise, and it shows that a characterized reference is a prerequisite for the method rather than a refinement.

6.5. **Comparison with the analytical error bound.** Since (3.7) is explicit, it could be used instead of a learned model. Figure 7 compares, for three values of α, the certified error, the bound and the out-of-fold Random Forest prediction. Over the grid, the bound overestimates the error by a median factor of $10^{7.6} \approx 4 \times 10^7$ (the median of $\log_{10}(\text{bound}/E)$ quoted in Section 6.1), and choosing M from it would require a median of 3.9 times as many terms (32 more) as the true M^*. The Random Forest prediction deviates from the true error by a median of 0.06 decades (95th percentile 0.39). The bound guarantees convergence; the learned model is what makes the error *estimate* usable.

TABLE 8. Predict-and-skip robustness (Gradient Boosting, logistic equation, 5-fold CV). Top: i.i.d. Gaussian noise of standard deviation σ added to the $\log_{10} E$ training targets. Bottom: *structured* label error—training targets computed against ABM references with step h ; in both cases predictions are scored against the certified labels. The trained model reproduces the erroneous labels almost exactly, so its scores agree with those of the labels themselves to three or four digits.

noise σ	0.00	0.01	0.05	0.10	0.20
MAE	0.079	0.077	0.084	0.101	0.136
R^2	0.9996	0.9996	0.9996	0.9995	0.9992

ABM step h of the label reference	0.01	0.005	0.0025
labels off by more than 2× (%)	84.0	82.7	81.0
MAE of the labels themselves	7.534	7.269	7.014
R^2 of the labels themselves	−1.0343	−0.9198	−0.8104
MAE of the trained model	7.536	7.272	7.017
R^2 of the trained model	−1.0343	−0.9198	−0.8104

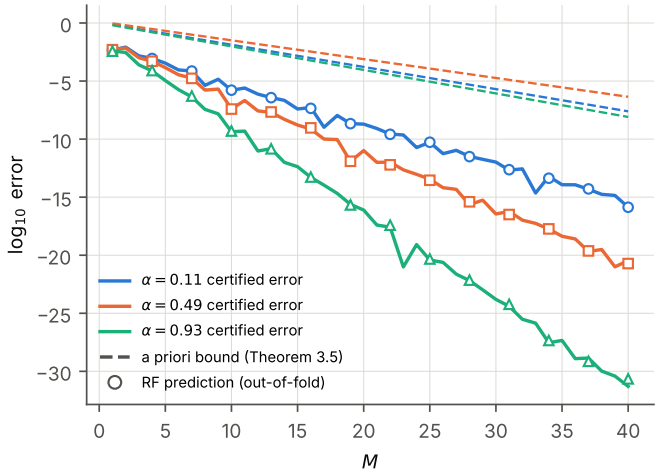

FIGURE 7. Certified error (solid), *a priori* bound (3.7) (dashed) and out-of-fold Random Forest prediction (markers) for three values of α (logistic equation, $[0, 1]$).

6.6. **Interaction between α and M.** Feature importance alone does not show that α and M are coupled: equal importance only means that both matter. We therefore test for an interaction directly. A Gradient Boosting model restricted to depth-one trees is additive, $\widehat{y} = g_1(\alpha) + g_2(M)$, and reaches only $R^2 = 0.933$ on predict-and-skip (0.934 on search-then-learn), against 0.9996 (0.9961) for the tuned model with interactions. The Friedman–Popescu statistic [13] computed from the partial dependence of the tuned model is $H^2 = 0.059$ (0.058), i.e. about 6% of the variance of the joint partial dependence is due to the interaction. For the Riccati equation I the effect is stronger (additive $R^2 = 0.877$, $H^2 = 0.104$). Permutation importances on held-out folds, which are not biased towards high-cardinality features, are 0.43 ± 0.03 for α and 1.68 ± 0.03 for M (decrease in R^2 when the feature is permuted). The impurity-based importances, which are biased towards high-cardinality features, are 0.21 for α and 0.79 for M. The coupling of α and M is genuine but secondary to the dominant effect of M.

6.7. **Quantization in search-then-learn.** The target M^* is integer-valued and a step function of $\log_{10} \varepsilon$, while the regressors are smooth. The errors concentrate near the steps: for the Random Forest, the MAE is 0.46 for the half of the queries closest to a step boundary and 0.17 for the other half (correlation between the absolute error and the

distance to the nearest step: -0.43). This is why the prediction is rounded up. With $M = \lceil \widehat{M} \rceil$ the tolerance is met in 96.5% of the logistic queries, and with one additional term in 100%.

6.8. **Reference-free alternatives.** A natural cheap competitor estimates the truncation error from the next terms of the series alone. Table 9 compares the learned models with the two-term proxy $|u_{M+1}(T)| + |u_{M+2}(T)|$. As a *selector of* M the proxy is very reliable here: it met the tolerance in 100% of the logistic queries with a mean excess of 0.25 terms, compared with 96.5–97.4% for the rounded regressor predictions. As an *estimator of the error* the regressors are clearly more accurate, with an MAE of 0.079 versus 0.197 decades on the logistic data and 0.035 versus 0.469 on the Riccati equation I, where the proxy is conservative but loose. We therefore recommend the regressors for accuracy prediction (predict-and-skip) and, for this scalar problem, the proxy or a regressor with a one-term safety margin for the selection of M.

TABLE 9. Machine-learning surrogates versus the reference-free two-term tail proxy $|u_{M+1}(T)| + |u_{M+2}(T)|$. Left: share of STL queries for which the selected length meets the tolerance ($M = \lceil \widehat{M} \rceil$ for RF/GB, out-of-fold predictions) and mean excess $M - M^*$. Right: accuracy of the predicted $\log_{10} E(M, \alpha)$ over the whole grid (GB out-of-fold vs proxy).

	STL success (%)			mean $M - M^*$		PAS MAE		PAS R^2	
Equation	RF	GB	proxy	GB	proxy	GB	proxy	GB	proxy
logistic	96.5	97.4	100.0	0.50	0.25	0.079	0.197	0.9996	0.9972
Riccati I	91.3	90.6	100.0	0.56	2.11	0.035	0.469	0.9987	0.8858
Riccati II	90.3	92.1	96.4	0.58	0.27	0.071	0.279	0.9987	0.9843

6.9. **Cost, fair comparison and break-even.** Table 10 compares the cost of answering one search-then-learn query.

- *Full grid search* evaluates all $M = 1, \dots, 40$ against a reference: 40 series evaluations per query. *Linear search* stops at the first M that meets the tolerance, i.e. $\overline{M^*} = 13.8$ evaluations on average. Both are dominated by the reference solve, which is needed once per query.
- The learned surrogate replaces the query by one prediction, but its training cost must be amortized: generating the 1920 certified labels took 55 s (most of it in the 60-digit certificate) and fitting the final model 0.24 s (Gradient Boosting) or 0.49 s (Random Forest). The nested hyperparameter search took a further 48 s and 25 s respectively.

The break-even number of queries $N^* = C_{\text{train}}/(c_{\text{search}} - c_{\text{ML}})$ depends strongly on the baseline (Figure 8). Against full grid search with the certified reference, the surrogate pays off after 49 queries (70 including the tuning of the Random Forest); against full grid search with an ABM reference of step 10^{-4}, after about 500 (715); and against the cheap but inaccurate ABM reference with $h = 10^{-3}$, only after about 10^4 queries. Against the reference-free proxy, a single prediction is slower than the proxy itself (the series terms cost microseconds for a scalar equation), and break-even is reached only for batched predictions after about 7×10^5 queries. A comparison of a single prediction with a search, without amortizing the training cost, would therefore overstate the benefit of the surrogate.

In complexity terms, the first M LADM coefficients cost $O(M^2)$ operations (Cauchy products in (3.4)) and evaluating $u^{(M)}$ on n_t points costs $O(Mn_t)$. An ABM reference with N steps costs $O(N^2)$, and full grid search adds $O(M_{\max} n_t)$. A prediction costs $O(n_{\text{trees}}\,\text{depth})$ for the tree ensembles and $O(n_{\text{sv}})$ kernel evaluations for SVR. Training costs $O(n_{\text{trees}}\, n \log n)$ for Random Forest and $O(n_{\text{est}}\, n \log n)$ for Gradient Boosting (with n training samples and two features), and between $O(n^2)$ and $O(n^3)$ for SVR.

7. Riccati-type equations

To test whether the framework carries over to other equations, the construction was repeated for two fractional Riccati equations, both of the form (3.1) and therefore covered by Theorems 3.5 and 3.8:

$$\text{(I)} \quad {}^{LC}D_t^{\alpha} u = 1 - u^2, \qquad u(0) = 0.3, \qquad t \in [0, 1], \tag{7.1}$$

$$\text{(II)} \quad {}^{LC}D_t^{\alpha} u = 1 + 2u - u^2, \qquad u(0) = 0, \qquad t \in [0, 0.5], \tag{7.2}$$

Uncorrected Proof

TABLE 10. Cost of one search-then-learn query (logistic equation, $\alpha = 0.71$, $\varepsilon = 10^{-8}$) and break-even number of queries $N^* = C_{\text{train}}/(c_{\text{search}} - c_{\text{ML}})$, where C_{train} is the data-generation time plus the final fit (and, in the last column, the hyperparameter search). GB: Gradient Boosting; RF: Random Forest; "never": the search is cheaper per query than the prediction. Hardware: Intel Xeon Processor @ 2.10GHz, one thread; Python 3.13.16, NumPy 2.5.3, SciPy 1.18.1, scikit-learn 1.9.1.

Method	ms/query	N^* GB single	N^* GB batch	N^* RF batch	N^* RF batch+tuning
Full grid search ($M = 1, \ldots, 40$), certified reference	1 149	49	49	49	70
Full grid search, ABM reference, $h = 10^{-4}$	112	495	494	496	715
Full grid search, ABM reference, $h = 10^{-3}$	5.31	11 010	10 413	10 501	15 139
Linear search (stop at first hit), ABM, $h = 10^{-3}$	5.16	11 342	10 709	10 801	15 572
Two-term tail proxy (no reference)	0.0819	never	722 734	1 002 955	1 446 012
Gradient Boosting: single / batched prediction	0.293 / 0.00547	training: data 55.0 s + fit 0.24 s + tuning 47.9 s			
Random Forest: single / batched prediction	18.5 / 0.0266	training: data 55.0 s + fit 0.49 s + tuning 24.5 s			
kNN: single / batched prediction	0.724 / 0.00256	training: data 55.0 s + fit 0.00 s + tuning 0.3 s			

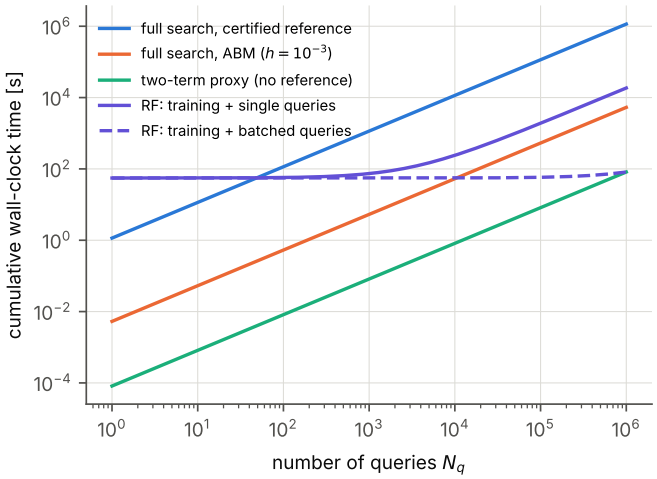

FIGURE 8. Cumulative wall-clock time versus the number of search-then-learn queries (logistic equation; data of Table 10).

the latter being the benchmark of [19]. The recursion has the same form, but *a new dataset and newly trained models are required for each equation*; the trained logistic models are not transferable.

7.1. **Convergence region.** For both equations $\gamma_0 > 1$ on the whole α range (Figure 4(b)), so convergence must be established through Theorem 3.8. For (I) on $[0, 1]$, convergence is certified for $\alpha \geq 0.73$ (validated $q_{\text{cert}} = 0.993$ at $\alpha = 0.73$), and for $\alpha \lesssim 0.70$ convergence is not certified, and the finite-N root-test estimate exceeds one, which is numerical evidence of divergence but not a proof. A grid over $\alpha \in [0.1, 1]$ would therefore consist largely of cases outside the certified region. The datasets are restricted to the certified region: $\alpha \in \{0.765, 0.770, \ldots, 1\}$ for (I) and 48 equispaced $\alpha \in [0.65, 1]$ for (II), each with $M = 1, \ldots, 40$, i.e. 1920 labels per equation, the same size as the logistic dataset.

7.2. **Results.** Tables 11 and 12 show that predict-and-skip again reaches $R^2 = 0.9987$ with Gradient Boosting on both equations. Here, however, the tree ensembles are clearly superior to the smooth models ($R^2 \approx 0.95$–0.99): near the

Uncorrected Proof

boundary of the convergence region the error is not monotone in M, and its oscillations are captured by trees but not by a polynomial, SVR or kNN. For (I) the learning curve (Figure 6) is still rising at the full size (Gradient Boosting: $R^2 = 0.993$ with 474 samples, 0.9987 with 1536), so a denser grid would still help slightly. For search-then-learn all models reach $R^2 \geq 0.990$; the rounded-up prediction met the tolerance in 91.3% (I) and 90.3% (II) of the queries, and with one extra term in 99.2% and 98.1%.

TABLE 11. Riccati equation I (${}^{LC}D_t^{\alpha} u = 1 - u^2$, $u_0 = 0.3$, $T = 1$, $\alpha \in [0.765, 1]$): nested 5-fold cross-validated performance.

	Predict-and-skip ($n = 1920$, $\log_{10} E$ units)			Search-then-learn ($n = 855$, units of M)			
Model	MAE	RMSE	R^2	MAE	RMSE	MAPE (%)	R^2
Random Forest	0.078 ± 0.007	0.120 ± 0.018	0.9954 ± 0.0014	0.66 ± 0.05	0.93 ± 0.07	3.2	0.9917 ± 0.0014
Gradient Boosting	0.035 ± 0.002	0.063 ± 0.009	0.9987 ± 0.0004	0.66 ± 0.03	0.83 ± 0.04	3.9	0.9934 ± 0.0008
SVR	0.247 ± 0.011	0.390 ± 0.023	0.9530 ± 0.0055	0.56 ± 0.01	0.69 ± 0.02	3.5	0.9956 ± 0.0004
kNN	0.304 ± 0.010	0.384 ± 0.017	0.9545 ± 0.0041	0.73 ± 0.04	0.94 ± 0.07	4.4	0.9915 ± 0.0014
Polynomial LS	0.271 ± 0.010	0.362 ± 0.020	0.9596 ± 0.0044	0.55 ± 0.01	0.68 ± 0.02	3.6	0.9956 ± 0.0005

TABLE 12. Riccati equation II (${}^{LC}D_t^{\alpha} u = 1 + 2u - u^2$, $u_0 = 0$, $T = 0.5$, $\alpha \in [0.65, 1]$): nested 5-fold cross-validated performance.

	Predict-and-skip ($n = 1920$, $\log_{10} E$ units)			Search-then-learn ($n = 1004$, units of M)			
Model	MAE	RMSE	R^2	MAE	RMSE	MAPE (%)	R^2
Random Forest	0.134 ± 0.004	0.208 ± 0.004	0.9966 ± 0.0002	0.64 ± 0.02	0.94 ± 0.09	3.8	0.9922 ± 0.0015
Gradient Boosting	0.071 ± 0.005	0.129 ± 0.009	0.9987 ± 0.0002	0.69 ± 0.04	0.90 ± 0.08	4.8	0.9928 ± 0.0009
SVR	0.280 ± 0.020	0.461 ± 0.029	0.9834 ± 0.0017	0.59 ± 0.04	0.74 ± 0.05	4.4	0.9952 ± 0.0006
kNN	0.237 ± 0.008	0.321 ± 0.014	0.9920 ± 0.0006	0.70 ± 0.03	0.91 ± 0.08	5.5	0.9926 ± 0.0015
Polynomial LS	0.328 ± 0.007	0.416 ± 0.016	0.9865 ± 0.0007	0.74 ± 0.04	1.06 ± 0.16	5.8	0.9901 ± 0.0023

Uncorrected Proof

7.3. **Comparison with an MLP used as the solver.** Machine learning has also been used to *solve* the Riccati equation directly, by training an MLP on the residual [21]. The two uses are complementary: an MLP solver replaces the series, whereas our surrogate only selects its truncation length. Table 13 contrasts them on equation (I) with a small MLP solver (our implementation, not that of [21]). The MLP reaches errors of 4×10^{-4} to 3×10^{-3}, limited by the L1 discretization of the derivative, after 0.1–20 s of training per value of α. As [21] shows, the activation function and the network design strongly affect the accuracy of such solvers, so these figures describe one simple design, not the best accuracy an MLP solver can reach. The LADM with a regressor-selected length meets the tolerance 10^{-4} in under 1 ms per query, once the regressor has been trained. The MLP solver, on the other hand, needs no convergence region and no series structure; this is its advantage whenever the decomposition series diverges.

8. Extending the feature set

The regressors of Sections 6–7 use only (α, M), with r, u_0 and T fixed, so they cannot be applied to other parameter values without retraining. Two remedies are available. First, by Proposition 3.3 the interval length is not an independent parameter for the logistic equation: the truncation error on $[0, T]$ with rate r equals the truncation error on $[0, 1]$ with the effective rate $r_{\text{eff}} = rT^{\alpha}$. A model trained on $[0, 1]$ with r as a feature therefore covers any interval $[0, T]$, provided it is queried with r_{eff} in place of r and r_{eff} lies within the training range of r. This identity was confirmed numerically ($\alpha = 0.6$, $u_0 = 0.3$, $r = 0.8$ on $[0, 2]$ against $r_{\text{eff}} = 0.8 \cdot 2^{0.6}$ on $[0, 1]$) to a relative difference of 6×10^{-15}. Second, u_0 and r can be added as features. We drew 300 random configurations $(\alpha, u_0, r) \in [0.05, 1] \times [0.05, 0.95] \times [0.1, 1.5]$, keeping only those whose numerically located convergence factor was below 0.9 on $[0, 1]$ (the validated q_{cert} of Section 4 is at most 0.91), and computed labels for $M = 1, \ldots, 40$ ($n = 12000$). The labels are certified exactly as in Section 4: the remainder is at most $8.5 \times 10^{-7} E(40)$ for every configuration, and the grid maximum lies within 2.6×10^{-5} (relative) of the supremum. Table 14 evaluates predict-and-skip with a five-fold split over *configurations*, so that every test

TABLE 13. Riccati equation I on $[0, 1]$, $u_0 = 0.3$: an MLP used as the solver (one hidden layer of 20 tanh units, trial function $u_0 + t\mathcal{N}(t)$, L1 discretization of the Liouville–Caputo derivative, L-BFGS, best of 3 restarts) versus the LADM with the decomposition length selected by the trained search-then-learn regressor. Error: maximum deviation from the certified reference on $[0, 1]$; time: training (MLP) or query-plus-evaluation (LADM), measured on the hardware and software of Table 10 (code: `cost_experiments.py`); timings are machine-dependent.

Approach	α	max error	time
MLP solver ($N = 100$ collocation pts)	0.85	2.96×10^{-3}	5.5 s
MLP solver ($N = 400$ collocation pts)	0.85	8.95×10^{-4}	19.9 s
MLP solver ($N = 100$ collocation pts)	0.90	2.26×10^{-3}	5.0 s
MLP solver ($N = 400$ collocation pts)	0.90	6.26×10^{-4}	16.5 s
MLP solver ($N = 100$ collocation pts)	1.00	1.67×10^{-3}	0.1 s
MLP solver ($N = 400$ collocation pts)	1.00	4.18×10^{-4}	2.7 s
LADM, $M = 37$ selected by GB ($\varepsilon = 10^{-4}$)	0.85	6.95×10^{-5}	0.77 ms
LADM, $M = 27$ selected by GB ($\varepsilon = 10^{-4}$)	0.90	2.52×10^{-5}	0.82 ms
LADM, $M = 18$ selected by GB ($\varepsilon = 10^{-4}$)	1.00	4.03×10^{-5}	0.78 ms

configuration is unseen. Gradient Boosting reaches $R^2 = 0.991$ (MAE 0.58 decades). This accuracy is lower than on the fixed-parameter grid, but the result shows that the framework can be queried for new (u_0, r, T) without retraining.

TABLE 14. Predict-and-skip with the extended feature set (α, M, u_0, r) (logistic equation, $T = 1$; 300 random parameter configurations, $n = 12000$). Five-fold `GroupKFold` over configurations: every test configuration (α, u_0, r) is unseen during training.

Model	MAE ($\log_{10} E$)	RMSE	R^2
Random Forest	0.780 ± 0.080	1.155 ± 0.155	0.9822 ± 0.0042
Gradient Boosting	0.583 ± 0.031	0.823 ± 0.063	0.9910 ± 0.0014
kNN	1.214 ± 0.157	2.001 ± 0.275	0.9463 ± 0.0131

Uncorrected Proof

9. DESIGN CHOICES AND LIMITATIONS

This section explains the choice of models, delimits the scope of the claims, and relates the surrogate to classical parameter-search methods.

9.1. **Choice of models.** Random Forest, Gradient Boosting and SVR were chosen as representatives of three distinct regression paradigms, and kNN and polynomial least squares as classical baselines; all were tuned by nested cross-validation (Table 2). Newer boosting libraries and neural-network regressors were not evaluated. For the two- to four-dimensional inputs considered here we would not expect large gains, but we have not tested this.

9.2. **Scope of the claims.** The convergence analysis covers equations with a quadratic right-hand side; the numerical study covers the logistic equation and two Riccati equations, within their certified convergence regions. We make no claim for other nonlinearities, for parameter ranges outside the certified convergence region, or for equations with spatial derivatives, each of which would require its own analysis. For the scalar equations studied here, the series terms are so cheap that a two-term tail proxy is a strong reference-free selector of M (Section 6.8). The surrogate's advantages are its markedly more accurate error prediction, batched queries, and settings in which each solver evaluation is expensive.

9.3. **Relation to classical parameter-search methods.** The surrogate was compared with full and linear grid search, the analytical bound and a tail proxy. Metaheuristic search (Bayesian optimization, genetic algorithms, particle swarm optimization) was not evaluated; for a one-dimensional integer search over at most 40 values, exhaustive search is already cheap relative to the reference solve.



10. Conclusion

For Liouville–Caputo fractional equations with a quadratic nonlinearity, including the logistic and Riccati equations, the LADM approximation of order M is the degree-M partial sum of a power series in t^α. This structure yields a rigorous *a priori* convergence theorem with an error bound explicit in the data, and a sharp convergence criterion with a computable certificate. The certificate also explains the behaviour observed in practice, including convergence when the *a priori* parameter exceeds one. Using certified truncation errors as labels, tuned tree ensembles predict $\log_{10} E(M,\alpha)$ with $R^2 = 0.9996$ (logistic) and 0.9987 (both Riccati equations). On the logistic grid the predictions are a median of about seven orders of magnitude sharper than the *a priori* bound, and on all three equations they are 2.5 to 13 times more accurate than a reference-free tail proxy. The minimal decomposition length is predicted with $R^2 \geq 0.99$, and with a one-term safety margin it meets the tolerance in 98–100% of the queries. Once training is amortized, the surrogate is cheaper than full grid search with an accurate reference after about 50 (certified reference) to 500 (fine-step Adams reference) queries, but it does not beat the tail proxy as a selector of M for these scalar problems. The ML grid fixes the parameters other than (α, M); adding (u_0, r) as features retains $R^2 = 0.99$ on unseen configurations, and the interval length enters only through rT^α. Future work includes divergence-aware targets, Padé-type continuation outside the convergence region, systems and equations with spatial derivatives where solver evaluations are expensive, and the modified Atangana–Baleanu–Caputo derivative [4].

Uncorrected Proof

Acknowledgment

The authors thank the Editor and the anonymous reviewers for their constructive comments, which improved the paper.

References

[1] K. Abbaoui and Y. Cherruault, *Convergence of Adomian's method applied to differential equations*, Comput. Math. Appl., *28*(5) (1994), 103–109.
[2] G. Adomian, *Solving Frontier Problems of Physics: The Decomposition Method*, Kluwer Academic Publishers, Dordrecht, 1994.
[3] S. Ahmed, S. Jahan, and K. S. Nisar, *Hybrid Fibonacci wavelet method to solve fractional-order logistic growth model*, Math. Methods Appl. Sci., *46*(15) (2023), 16218–16231.
[4] M. Al-Refai, M. I. Syam, and D. Baleanu, *Analytical treatments to systems of fractional differential equations with modified Atangana–Baleanu derivative*, Fractals, *31*(10) (2023), 2340156.
[5] I. Area and J. J. Nieto, *Power series solution of the fractional logistic equation*, Physica A, *573* (2021), 125947.
[6] A. Atangana and D. Baleanu, *New fractional derivatives with nonlocal and non-singular kernel: Theory and application to heat transfer model*, Therm. Sci., *20*(2) (2016), 763–769.
[7] L. Breiman, *Random forests*, Mach. Learn., *45*(1) (2001), 5–32.
[8] Y. Cherruault, *Convergence of Adomian's method*, Kybernetes, *18*(2) (1989), 31–38.
[9] K. Diethelm, *The Analysis of Fractional Differential Equations*, Lecture Notes in Mathematics 2004, Springer, Berlin, 2010.
[10] K. Diethelm, N. J. Ford, and A. D. Freed, *A predictor-corrector approach for the numerical solution of fractional differential equations*, Nonlinear Dynam., *29* (2002), 3–22.
[11] K. Diethelm, N. J. Ford, and A. D. Freed, *Detailed error analysis for a fractional Adams method*, Numer. Algorithms, *36*(1) (2004), 31–52.
[12] J. H. Friedman, *Greedy function approximation: A gradient boosting machine*, Ann. Statist., *29*(5) (2001), 1189–1232.
[13] J. H. Friedman and B. E. Popescu, *Predictive learning via rule ensembles*, Ann. Appl. Stat., *2*(3) (2008), 916–954.
[14] M. Izadi, *A comparative study of two Legendre-collocation schemes applied to fractional logistic equation*, Int. J. Appl. Comput. Math., *6*(3) (2020).
[15] M. Izadi and H. M. Srivastava, *A discretization approach for the nonlinear fractional logistic equation*, Entropy, *22*(11) (2020), 1328.

[16] M. Izadi and H. M. Srivastava, *Numerical approximations to the nonlinear fractional-order logistic population model with fractional-order Bessel and Legendre bases*, Chaos Solitons Fractals, *145* (2021), 110779.
[17] S. A. Khuri, *A Laplace decomposition algorithm applied to a class of nonlinear differential equations*, J. Appl. Math., *1*(4) (2001), 141–155.
[18] S. Marakonda and Y. Rajashekhar Reddy, *ML-MS-TSADM5: A machine learning assisted multi-stage two-step Adomian decomposition method with coupled parameter selection for the nonlinear Burgers equation*, IAENG Int. J. Appl. Math., *56*(10) (2026), 4586–4601.
[19] S. Momani and N. Shawagfeh, *Decomposition method for solving fractional Riccati differential equations*, Appl. Math. Comput., *182*(2) (2006), 1083–1092.
[20] A. E. Owoyemi, C. Phang, and A. Isah, *Perturbation-iteration approach for fractional-order logistic differential equations*, Nonlinear Eng., *14*(1) (2025).
[21] H. Patel, Y. F. Patel, and M. Izadi, *Multilayer perceptron-based solutions of the Riccati equation demonstrating the impact of activation functions and network design*, Contemp. Math., *7*(4) (2026), 4713–4747.
[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, J. Mach. Learn. Res., *12* (2011), 2825–2830.
[23] I. Podlubny, *Fractional Differential Equations*, Academic Press, San Diego, 1999.
[24] H. M. Srivastava, *An introductory overview of fractional-calculus operators based upon the Fox–Wright and related higher transcendental functions*, J. Adv. Eng. Comput., *5*(3) (2021), 135–166.
[25] M. I. Syam and M. Al-Refai, *Fractional differential equations with Atangana–Baleanu fractional derivative: Analysis and applications*, Chaos Solitons Fractals X, *2* (2019), 100013.
[26] H. Yépez-Martínez and J. F. Gómez-Aguilar, *Laplace variational iteration method for modified fractional derivatives with non-singular kernel*, J. Appl. Comput. Mech., *6*(3) (2020), 684–698.

Uncorrected Proof