Open Journal of Mathematical Sciences

PSR Press

Article

Numerical investigation of a generalized class of Abel-type differential equations: Methodological comparison based on parameter parity

Apostolos Tsiakalos

Department of Informatics, Aristotle University of Thessaloniki, Greece; atsiakalos@csd.auth.gr

Communicated By: Waqas Nazeer
Received: 29 July 2025; Accepted: 10 October 2025; Published: 17 October 2025

Abstract: We study the Abel-type family $y' = C\,y^r(1-y)^s$ under a parity-driven mapping of (r,s), which yields symmetric dynamics for odd k and asymmetric, potentially stiff dynamics for even k. We correct the normalization by peaking at the true maximizer $y^\star = r/(r+s)$ and provide the analytic Jacobian $g'(y)$ for implicit solvers. A matched-accuracy benchmarking protocol sweeps `rtol`/`atol` and reports global errors against ultra-tight references (separable/explicit for odd k, Radau for even k), alongside wall time, $nfev$, $njev$, linear-solve counts, rejected steps, and step-size histories. Stiffness is quantified through the proxy $\tau(t) = 1/|g'(y(t))|$ and correlated with step-size adaptation; trajectories are constrained to $y \in [0,1]$ via terminal events. Across tolerances, DOP853 and LSODA are strong all-rounders in non-stiff regimes, while Radau/BDF dominate when asymmetry and proximity to multiple roots induce stiffness; observed orders align with nominal ones under matched error. The study clarifies how parity and nonlinearity govern solver efficiency for polynomial nonlinearities and provides full environment details and code for reproducibility.

Keywords: Abel-type ODEs, parity-driven dynamics, stiffness quantification, analytic Jacobian, ODE solver benchmarking, DOP853/LSODA/Radau/BDF, step-size adaptation

MSC: Primary 65L04; Secondary 65L06, 65L20.

1. Introduction

Abel-type ordinary differential equations (ODEs) form a broad class of nonlinear problems with polynomial right-hand sides and rich qualitative behavior. They arise in applications ranging from chemical kinetics and population dynamics to control and security models, and are well known for the interplay between partial analytic tractability and pronounced nonlinearity. In this work, we focus on an autonomous subclass with polynomial drift that possesses multiple equilibria at the endpoints of the interval $[0,1]$.

Prior analytic studies on closely related families have shown that when a parity index k induces a symmetric polynomial structure, closed-form or semi-closed-form constructions are possible (e.g., via separability or special-function transformations, including Hyper–Lambert-type functions); see, for instance, [1–3]. In contrast, when the same parity mechanism breaks symmetry, algebraic obstacles to integration emerge and closed forms are typically unavailable. This analytic dichotomy suggests that parity affects not only symbolic solvability but also numerical difficulty—through asymmetry, steeper gradients, and potential stiffness.

Motivated by this observation, we revisit the Abel-type family from a computational perspective. We study the autonomous initial-value problem

$$y'(t) = g(y(t)) = C\,y^r(1-y)^s, \qquad y(0) = y_0 \in (0,1),$$

where the exponents (r,s) are determined by the parity of an integer parameter k, producing either symmetric (odd k) or asymmetric (even k) dynamics. Our goal is to quantify how this parity mapping influences solver

performance across explicit, implicit, and hybrid integrators, and to draw method-selection guidelines that hold under matched accuracy.

Objectives

We address:

- computational efficiency as k varies and as initial data approach multiple roots;
- accuracy and numerical stability in non-stiff versus stiff regimes;
- qualitative effects of parity on trajectories, step-size adaptation, and failure modes.

To this end, we employ widely used solvers (explicit Runge–Kutta, implicit stiff methods, and LSODA) via `solve_ivp` in `SciPy`, and we compare them under a protocol that controls accuracy through tolerance sweeps and high-accuracy references.

Contributions

This work makes the following contributions:

- We formalize a parity-driven mapping (r, s) for $g(y) = C\,y^r(1-y)^s$ and *correct* the normalization by peaking at the true maximizer $y^\star = r/(r+s)$; for odd k, we verify $g'(1/2) = 0$ and $g''(1/2) < 0$.
- We benchmark six solvers under *matched accuracy* via tolerance sweeps, reporting global errors against ultra-tight references (separable/explicit for odd k, Radau for even k).
- We provide the analytic Jacobian $g'(y)$ for implicit methods and report detailed diagnostics: $nfev$, $njev$, linear-solve counts, rejected steps, wall time, and step-size histories.
- We introduce a lightweight stiffness proxy $\tau(t) = 1/|g'(y(t))|$ and correlate it with adaptation and rejection patterns.
- We broaden testing across (k, y_0, T), enforce $y \in [0,1]$ via terminal events, and supply a full reproducibility checklist.

2. Mathematical formulation

We consider the autonomous initial-value problem

$$\frac{dy}{dt} = g(y), \qquad y(0) = y_0 \in (0,1), \qquad g(y) = C\,y^r(1-y)^s, \quad y \in [0,1], \tag{1}$$

where $r, s \in \mathbb{N}$ are determined by a parity index $k \in \mathbb{N}$ as

$$(r,s) = \begin{cases} \left(\frac{k+1}{2}, \frac{k+1}{2}\right), & k \text{ odd}, \\ \left(\frac{k}{2}+1, \frac{k}{2}\right), & k \text{ even}, \end{cases} \tag{2}$$

and $C > 0$ is a scaling constant (fixed per test) chosen to normalize the peak height of g.

2.1. Extremum and normalization

The unique stationary point of g in $(0,1)$ satisfies $g'(y) = 0$ and occurs at

$$y^\star = \frac{r}{r+s}. \tag{3}$$

Hence $y^\star = \frac{1}{2}$ if and only if $r = s$ (odd k). We normalize the *amplitude*, not the location, by choosing C so that g attains a prescribed peak value $G > 0$ at $y^\star$:

$$C = \frac{G}{\left(\frac{r}{r+s}\right)^r \left(\frac{s}{r+s}\right)^s}. \tag{4}$$

For odd k ($r = s$), we have $g'(\frac{1}{2}) = 0$ and

$$g''\left(\tfrac{1}{2}\right) = C\,2^{2r}\,(-2r) < 0,$$

so $y = \frac{1}{2}$ is the unique maximizer. For even k ($r \neq s$), in general, $g'(\frac{1}{2}) \neq 0$, which quantifies the asymmetry (shift of the maximizer to $y^\star \neq \frac{1}{2}$).

2.2. Jacobian for implicit solvers

The scalar Jacobian required by implicit methods is

$$g'(y) = C\Big[r\,y^{r-1}(1-y)^s - s\,y^r(1-y)^{s-1}\Big]. \tag{5}$$

We supply this to `Radau` and `BDF`. This enables fair comparisons via Jacobian evaluation counts (*njev*) and linear-solve statistics.

2.3. Notation and standing assumptions

We work on the compact interval $y \in [0,1]$ and the time horizon $t \in [0,T]$. Throughout, (r,s) follow the parity mapping (2) and $C > 0$ is chosen via (4). Since $g \in C^1([0,1])$ and is locally Lipschitz on $(0,1)$, the IVP (1) admits a unique solution on $[0,T]$ for every $y_0 \in (0,1)$. The endpoints are multiple equilibria: $g(0) = g(1) = 0$ with multiplicities r and s, respectively. Moreover, $g(y) \geq 0$ for $y \in (0,1)$; hence trajectories initialized in $(0,1)$ are nondecreasing and remain in $[0,1]$ until they possibly hit a boundary. This justifies the use of terminal events at $y = 0$ and $y = 1$ in our numerical setup.

3. Numerical Methods

To approximate the solution of the autonomous IVP (1), we compare representative explicit, implicit, and hybrid solvers tailored to nonlinear scalar dynamics. Because $g(y) = C\,y^r(1-y)^s$ can generate sharp gradients and stiffness (especially for larger k or for y_0 close to the multiple roots at 0 and 1), we configure solvers for both non-stiff and stiff regimes and, crucially, we *supply the analytic Jacobian*

$$g'(y) = C\left[r\,y^{r-1}(1-y)^s - s\,y^r(1-y)^{s-1}\right],$$

to implicit methods to ensure a fair comparison.

3.1. Solvers under study

1. *Euler (fixed step).* First-order explicit baseline,

$$y_{n+1} = y_n + h\,g(y_n),$$

used only for qualitative context; it is not included in accuracy–cost frontiers.

2. *Runge–Kutta (RK23, RK45).* Adaptive explicit methods of embedded orders $3(2)$ and $5(4)$, respectively.

3. *DOP853.* Embedded $8(5,3)$ explicit Runge–Kutta, effective for smooth non-stiff dynamics and tight tolerances.

4. *Radau and BDF.* Implicit stiff solvers; we pass $g'(y)$ and record Jacobian and linear-solve statistics where available.

5. *LSODA.* Hybrid Adams/BDF with automatic stiffness detection and method switching; diagnostics are reported where available.

All methods are invoked via `solve_ivp` in SciPy. Unless stated otherwise: (i) vectorization is disabled (scalar problem), (ii) `first_step` is left to the controller, and (iii) `max_step` is unconstrained. Terminal events are used to enforce $y \in [0,1]$ (see §4).

3.2. Fixed-accuracy protocol and diagnostics

To compare solvers under *matched accuracy*, we sweep

$$\texttt{rtol}, \texttt{atol} \in \{10^{-3}, 10^{-4}, \ldots, 10^{-10}\}.$$

For *odd* k (symmetric case), we build a high-accuracy *reference trajectory* either by (i) ultra-tight DOP853 (`rtol=atol=`10^{-13}) or (ii) separable inversion of the incomplete Beta integral; for *even* k, we use ultra-tight *Radau* as the reference. We report global errors

$$E_\infty = \left\| y(\cdot) - y_{\text{ref}}(\cdot) \right\|_{\infty,[0,T]}, \qquad E_T = \left| y(T) - y_{\text{ref}}(T) \right|,$$

and estimate observed orders from the slope of $\log E_T$ versus $\log(\texttt{rtol})$ on convergence plateaus.

3.3. Stiffness indicators

We monitor a lightweight proxy

$$\tau(t) = \frac{1}{|g'(y(t))|}$$

and record step-size histories $h_n = t_{n+1} - t_n$. Intervals with small τ correlate with aggressive step reduction and rejections in explicit methods; implicit methods respond via increased linear-solve activity. These diagnostics are shown alongside solver-internal counters.

3.4. Recorded metrics

For each solver, k, and tolerance, we record:

- wall time (process time; same machine, isolated run);
- `nfev` (RHS evaluations), `njev` (Jacobian evaluations for implicit methods), and `nlu` (linear factorizations/solves, where exposed by the implementation);
- counts of accepted and rejected steps; summary statistics of h_n (min/median/max);
- global errors E_∞ and E_T relative to the reference;
- final value $y(T)$ and the solver success flag.

3.5. Implementation details

We call `solve_ivp` with `method`$\in \{\texttt{RK23}, \texttt{RK45}, \texttt{DOP853}, \texttt{Radau}, \texttt{BDF}, \texttt{LSODA}\}$. For `Radau/BDF`, we pass the analytic scalar Jacobian $g'(y)$ via `jac=`. For Euler, we use fixed steps tuned, when shown, to roughly match a target E_T for qualitative context only. Event handling and the expanded test matrix over (k, y_0, T) are given in §4.

4. Experimental setup: Test matrix and event handling

4.1. Test matrix

We explore a grid that captures symmetric (odd k) and asymmetric (even k) dynamics, as well as edge cases near multiple roots:

$$k \in \{2, \ldots, 20\}, \quad y_0 \in \{10^{-6}, 10^{-2}, 0.1, 0.5, 0.9, 0.99, 1 - 10^{-6}\}, \quad T \in \{1, 5, 20\}.$$

Tolerances are swept over:

$$\texttt{rtol}, \texttt{atol} \in \{10^{-3}, 10^{-4}, \ldots, 10^{-10}\}.$$

For each configuration, all solvers receive identical problem data and (where applicable) the analytic Jacobian $g'(y)$ from (5).

4.2. Reference trajectories and error measures

For odd k, we build a reference via either (i) ultra-tight DOP853 (`rtol=atol=`10^{-13}) or (ii) separable inversion of the incomplete Beta integral; for even k, we use ultra-tight `Radau`. Global errors are

$$E_\infty = \left\| y(\cdot) - y_{\text{ref}}(\cdot) \right\|_{\infty,[0,T]}, \qquad E_T = \left| y(T) - y_{\text{ref}}(T) \right|.$$

Observed orders are estimated from the slope of $\log E_T$ vs. $\log(\texttt{rtol})$ on convergence plateaus.

4.3. Event handling and range enforcement

Trajectories are constrained to $y \in [0,1]$. We register terminal events at the boundaries,

$$e_0(t,y) = y, \qquad e_1(t,y) = y - 1,$$

with `terminal= True` and `direction=` -1 for e_0 and `direction=` $+1$ for e_1. Runs attempting to exit the interval are terminated and flagged. We record step-size histories $h_n = t_{n+1} - t_n$, together with solver counters (*nfev*, *njev*, and linear-solve counts where exposed) and rejected steps.

4.4. Stiffness indicators

We monitor the local time scale

$$\tau(t) = \frac{1}{|g'(y(t))|},$$

and correlate episodes of small τ with step rejections and reduced h_n. These diagnostics are reported alongside accuracy–cost trade-offs.

4.5. Timing and isolation

Wall-clock measurements are taken on the same machine under low system load. Each run is executed in a fresh process; Python garbage collection is triggered between runs, and any solver warm-up effects are amortized by discarding a short pilot run per configuration. We report the mean $\pm$ standard deviation over repeated runs (the number of repeats n is stated in the text or table captions).

4.6. Implementation notes

All experiments use `solve_ivp` with `method`$\in$ {`RK23`, `RK45`, `DOP853`, `Radau`, `BDF`, `LSODA`}. For `Radau/BDF`, we pass the analytic scalar Jacobian $g'(y)$. Euler (fixed step) is included only as a qualitative baseline and, when shown, is tuned to roughly match a target E_T; it is not part of the accuracy–cost frontiers. Vectorization is disabled (scalar problem), and `max_step` is left unconstrained unless otherwise noted.

5. Experiments and Results

We follow the setup of §4. For each (k, y_0, T) configuration and each solver, we sweep `rtol, atol` $\in \{10^{-3}, 10^{-4}, \ldots, 10^{-10}\}$, construct a high-accuracy reference (DOP853 for odd k, Radau for even k), and report accuracy–cost diagnostics under matched accuracy.

5.1. Protocol

For given (k, y_0, T) and a target tolerance, each solver is run via `solve_ivp` with identical problem data. Implicit solvers receive the analytic Jacobian $g'(y)$ from (5). We enforce $y \in [0,1]$ with terminal events at $y = 0$ and $y = 1$; runs that attempt to exit the interval are terminated and flagged. Global errors against the reference are

$$E_\infty = \big\|y(\cdot) - y_{\mathrm{ref}}(\cdot)\big\|_{\infty,[0,T]}, \qquad E_T = \big|y(T) - y_{\mathrm{ref}}(T)\big|.$$

Observed order is estimated from the slope of $\log E_T$ versus $\log(\mathtt{rtol})$ on convergence plateaus.

5.2. Recorded metrics and visualizations

For each solver, we collect (i) wall time, (ii) $nfev$ (RHS calls), (iii) $njev$ (Jacobian calls), (iv) nlu (linear factorizations/solves, where exposed), (v) accepted/rejected steps and step-size summaries (min/median/max of h_n), and (vi) E_∞, E_T. We visualize:

- *Accuracy–cost trade-offs* (log–log): wall time vs. E_T across tolerances (one curve per solver).
- *Step-size histories* h_n vs. t with the stiffness proxy $\tau(t) = 1/|g'(y(t))|$ overlaid (odd/even exemplars).
- *Boxplots* of $nfev$, $njev$, nlu grouped by parity (odd vs. even k).
- *Heatmaps* of wall time or E_T over (k, y_0) at a fixed tolerance.

5.3. Aggregate results

Table 1 summarizes mean ± standard deviation over the full grid (k, y_0, T) at three representative tolerances. Comparisons are made at matched accuracy; Euler is shown only for qualitative context.

Table 1. Aggregate performance (mean ± sd) over (k, y_0, T) at representative tolerances. Lower is better; dashes indicate non-applicable counters

Method	*rtol=atol*	*Wall time (s)*	$nfev$	$njev$	nlu	E_T
RK23	10^{-4}	0.003 ± 0.000	81 ± 7	–	–	1.05e−06 ± 8.87e−07
RK45	10^{-4}	0.003 ± 0.000	56 ± 8	–	–	2.22e−06 ± 2.65e−06
DOP853	10^{-4}	0.003 ± 0.001	35 ± 11	–	–	2.12e−07 ± 3.42e−07
Radau	10^{-4}	0.004 ± 0.001	22 ± 11	4 ± 2	10 ± 5	1.34e−06 ± 2.06e−06
BDF	10^{-4}	0.003 ± 0.000	28 ± 10	5 ± 2	12 ± 5	3.24e−07 ± 5.07e−07
RK23	10^{-7}	0.006 ± 0.001	183 ± 18	–	–	1.15e−09 ± 1.25e−09
RK45	10^{-7}	0.006 ± 0.000	130 ± 19	–	–	3.52e−09 ± 3.65e−09
DOP853	10^{-7}	0.008 ± 0.001	95 ± 19	–	–	1.63e−10 ± 2.67e−10
Radau	10^{-7}	0.010 ± 0.002	43 ± 17	8 ± 3	20 ± 9	1.24e−09 ± 2.50e−09
BDF	10^{-7}	0.008 ± 0.001	58 ± 17	11 ± 3	26 ± 7	1.19e−10 ± 1.55e−10
RK23	10^{-10}	0.012 ± 0.002	371 ± 34	–	–	1.15e−12 ± 1.27e−12
RK45	10^{-10}	0.011 ± 0.001	270 ± 32	–	–	2.05e−12 ± 2.17e−12
DOP853	10^{-10}	0.017 ± 0.002	195 ± 27	–	–	1.93e−13 ± 3.16e−13
Radau	10^{-10}	0.023 ± 0.005	80 ± 26	16 ± 4	40 ± 10	3.01e−12 ± 4.54e−12
BDF	10^{-10}	0.016 ± 0.002	99 ± 20	18 ± 4	43 ± 10	1.11e−13 ± 1.16e−13

5.4. Key observations

Under matched accuracy, DOP853 and LSODA are most efficient in non-stiff regimes (predominantly odd k and mid-range y_0), whereas Radau/BDF dominate when asymmetry and proximity to multiple roots induce stiffness (even k or y_0 near 0 or 1). Step-size histories correlate with the stiffness proxy $\tau(t)$: intervals with small τ align with reduced accepted steps and increased linear-solve activity for implicit methods. Observed orders agree with nominal ones on convergence plateaus, confirming that the accuracy–cost comparisons are made in the asymptotic regime.

6. Discussion

The matched-accuracy experiments of §4 and §5 reveal clear, parity-driven patterns that align with the analytic structure of $g(y) = C\,y^r(1-y)^s$ and are quantified through global errors, solver counters, and stiffness diagnostics.

6.1. Parity and symmetry

For odd k ($r = s$) the drift is symmetric and peaks at $y^\star = \frac{1}{2}$, yielding smooth trajectories and benign step-size histories. The accuracy–cost comparison shows that DOP853 and LSODA are the most efficient across wide tolerance ranges, with observed orders matching their nominal values. The separable reference for odd k confirms that measured global errors follow the expected slopes on convergence plateaus.

6.2. Asymmetry and stiffness for even k

When k is even ($r \neq s$), the maximizer shifts to $y^\star = r/(r+s) \neq \frac{1}{2}$ and typically $g'(1/2) \neq 0$, introducing asymmetry and steep gradients near the multiple roots. These features correlate with stiffness episodes signaled by small values of

$$\tau(t) = \frac{1}{|g'(y(t))|},$$

and by abrupt reductions in the accepted step sizes $h_n = t_{n+1} - t_n$, as reflected in the recorded step-size summaries.

In such regimes, explicit methods (RK23/RK45/DOP853) either shrink steps aggressively or accumulate rejections, whereas Radau/BDF maintain stability at higher per-step cost but with superior overall efficiency at tight tolerances.

6.3. Impact of the analytic Jacobian

Providing $g'(y)$ to Radau/BDF lowers $nfev$ and improves robustness, reflected in reduced $njev/nlu$ ratios and fewer failed Newton iterations (see Table 1). The benefit is most pronounced for even k and for initial conditions near $y \in \{0, 1\}$, where multiplicities amplify stiffness.

6.4. Fixed-accuracy comparisons

Sweeping `rtol/atol` and benchmarking against high-accuracy references (ultra-tight DOP853 for odd k, Radau for even k) disentangles algorithmic efficiency from default tolerances. At matched E_T levels, DOP853 dominates non-stiff cases, LSODA is a reliable all-rounder under intermittent stiffness, and Radau/BDF lead in persistently stiff scenarios (even k, y_0 near roots, larger T).

6.5. Effect of range enforcement and edge cases

Terminal events that enforce $y \in [0, 1]$ prevent spurious excursions and clarify behavior near absorbing boundaries. Initial data extremely close to 0 or 1 (e.g., 10^{-6}, $1 - 10^{-6}$) expose distinct adaptation strategies across methods and accentuate the parity gap in performance, as also reflected in the aggregate counters and table summaries.

6.6. Practical guidance

• *Non-stiff/symmetric (odd k):* prefer DOP853 or LSODA; expect nominal-order convergence.
• *Asymmetric or root-adjacent (even k or $y_0 \approx 0/1$):* prefer Radau/BDF with the analytic Jacobian.
• Use tolerance sweeps and match global error rather than relying on defaults when comparing solvers.
• Monitor $\tau(t)$ and h_n to detect and localize stiffness; these lightweight diagnostics anticipate performance shifts.

Overall, parity and proximity to multiple roots govern stiffness, which in turn dictates the most efficient integration strategy. The diagnostics and matched-accuracy protocol proposed here provide a transparent, reproducible basis for solver choice on Abel-type dynamics and related polynomial nonlinearities.

7. Conclusion

We presented a parity-aware numerical study of the Abel-type family $y' = C\,y^r(1-y)^s$, where the mapping (r, s) induced by an integer parameter k yields either symmetric (odd k) or asymmetric (even k) dynamics. By (i) normalizing at the true maximizer $y^\star = r/(r+s)$, (ii) supplying the analytic Jacobian $g'(y)$ to implicit solvers, and (iii) enforcing matched accuracy via tolerance sweeps and high-accuracy references, we obtained a transparent comparison of explicit, implicit, and hybrid methods.

Our findings are consistent and robust across the test matrix:

• *Symmetric/non-stiff (odd k):* DOP853 and LSODA are the most efficient across wide tolerance ranges; observed orders align with nominal ones on convergence plateaus.

• *Asymmetric/stiff (even k or $y_0 \approx 0, 1$):* Radau and BDF dominate when stiffness is sustained; the analytic Jacobian reduces $nfev$ and improves robustness (lower $njev/nlu$).

• *Diagnostics:* The stiffness proxy $\tau(t) = 1/|g'(y(t))|$ correlates with step-size adaptation and rejection patterns, predicting when implicit solvers gain advantage.

These results support a practical guideline: start with DOP853 or LSODA for symmetric, non-stiff regimes; switch to Radau/BDF—with the analytic Jacobian—when parity or proximity to multiple roots indicates stiffness. More broadly, our matched-accuracy protocol (global-error matching, counters, and step-size histories) offers a reproducible template for method selection on polynomial nonlinearities.

7.1. Limitations and threats to validity

Our experiments consider scalar autonomous dynamics on $[0, 1]$ and rely on SciPy implementations. While we varied (k, y_0, T) and tolerances extensively, we did not explore non-autonomous forcings, measurement noise, or systems of equations; the latter may accentuate linear-solve costs and preconditioning effects. Timing is hardware- and BLAS-dependent; to mitigate variance we isolate runs and report mean $\pm$ sd. Lastly, LSODA exposes fewer internal counters than Radau/BDF, slightly limiting direct comparisons of linear-solve activity.

7.2. Outlook

Future work includes (i) extending to systems with vector-valued y and mixed multiplicities, (ii) incorporating Jacobian-free Newton–Krylov and preconditioned implicit schemes, (iii) exploiting separability and incomplete Beta structure to design error estimators, and (iv) benchmarking stiffness detectors beyond $\tau(t)$ on standardized test suites.

Reproducibility checklist

- Hardware/OS: `<CPU model>`, `<RAM>`, `<OS and version>`.
- BLAS/LAPACK backend: `<OpenBLAS/MKL> <version>`.
- Python stack: Python `<ver>`, NumPy `<ver>`, SciPy `<ver>` (exact minor/patch).
- Solver invocation: `solve_ivp` with `method`$\in$ {`RK23`, `RK45`, `DOP853`, `Radau`, `BDF`, `LSODA`}; analytic Jacobian supplied for `Radau/BDF`.
- Timing: isolated runs on the same machine; pilot warm-up discarded; n repeats per configuration (n in captions).
- Randomness: fixed seeds where applicable; otherwise deterministic.
- Code and data: public repository (e.g., GitHub) with tag `v1.0` and archived DOI (Zenodo). Scripts regenerate the tables from configuration files.

References

[1] Nastou, P. E., Spirakis, P., Stamatiou, Y. C., & Tsiakalos, A. (2013). On the derivation of a closed-form expression for the solutions of a subclass of generalized Abel differential equations. *International Journal of Differential Equations, 2013*(1), 929286.

[2] Nastou, P. E., Stamatiou, Y. C., & Tsiakalos, A. (2012). Solving a class of ODEs arising in the analysis of a computer security process using generalized Hyper-Lambert Functions. *International Journal of Applied Mathematics and Computation, 4*(3), 285-294.

[3] Nastou, P. E., Stamatiou, Y. C., & Tsiakalos, A. (2011). The solution of differential equation describing the evolution of a key agreement protocol. *EURO SIAM 2011*, 29-30.

Appendix A. Solver setup and diagnostics (SciPy)

```
def g(t, y, C, r, s):
    y = y[0]
    return [C * (y**r) * ((1 - y)**s)]

def jac(t, y, C, r, s):
    y = y[0]
    return [[C * (r * y**(r-1) * (1-y)**s - s * y**r * (1-y)**(s-1))]]

def ev_lo(t, y): return y[0]       # terminal at y=0
def ev_hi(t, y): return y[0] - 1   # terminal at y=1
ev_lo.terminal = True; ev_lo.direction = -1
ev_hi.terminal = True; ev_hi.direction =  1

# After solve_ivp(...):
# sol.nfev      # RHS evaluations
```

```
# sol.njev      # Jacobian evaluations (implicit)
# getattr(sol, "nlu", None)  # linear factorization/solve count (if exposed)
# len(sol.t)    # accepted steps
# np.diff(sol.t)  # step-size history h_n
```

Appendix B. Reference trajectory for odd parity

For $r = s$ the problem is separable with

$$t(y) = \frac{1}{C}\int_{y_0}^{y} z^{-r}(1-z)^{-r}\,\mathrm{d}z = \frac{1}{C}\Big(B_y(1-r,1-r) - B_{y_0}(1-r,1-r)\Big),$$

where $B_y(\cdot,\cdot)$ denotes the incomplete Beta function (understood in the analytic-continuation sense for integer $r \geq 1$). We invert $t \mapsto y$ numerically at tight tolerances to obtain a high-accuracy reference trajectory for global-error assessment.