RATIO BASED PARALLEL TIME INTEGRATION (RAPTI) FOR INITIAL VALUE PROBLEMS

Nabil R. Nassif

Noha Makhoul Karam

Yeran Soukiassian

Mathematics Department, American University of Beirut

IRISA, Campus Beaulieu, Université de Rennes I, Rennes, France

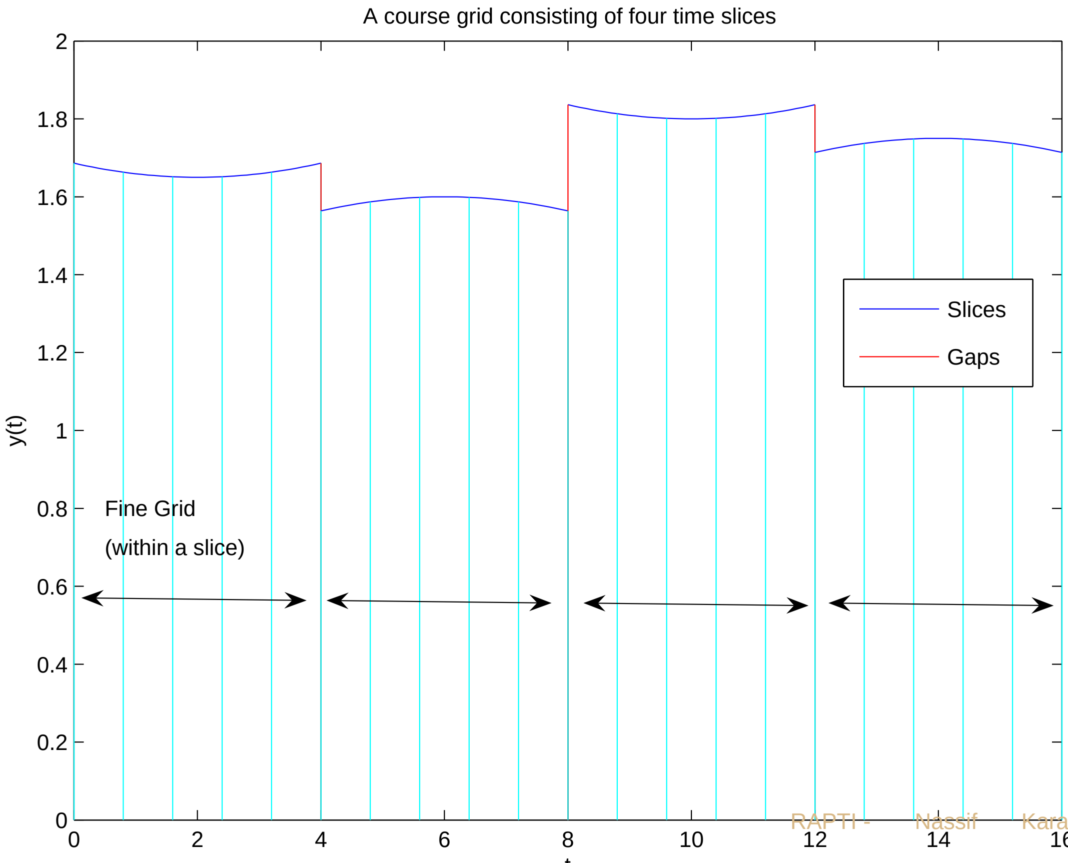
Computer Science Department, American University of Beirut, Beirut Lebanon

Outline

- Introduction and review
- Rescaling Methodology
- RAPTI for diffusion reaction problems
- RAPTI for Two Species Logistic Lotka Volterra
- RAPTI for Three Species Logistic Lotka Volterra
- Numerical Results
- Conclusion and Future Work

Introduction

A course grid consists of time slices. On each slice we intend to integrate “independently" an IVP, starting with its own initial value. This induces gaps between the end value of one slice and the starting value of the next slice.



Introduction: General steps of a predictor-corrector scheme for parallel time integration

1. Choice of a **coarse grid** that defines “large time-slices”.
2. Prediction of the initial values of the solution on each time-slice, using for example a Euler-implicit scheme.
3. Iteration until convergence of the following process:
   - Solving independent evolution problems on each time-slice, using any method on a **fine grid**, leading to discontinuity “jump ” between the end value of one slice and the predicted initial value on the next one.
   - Correction of the slices initial values by propagation of the jumps,

Introduction: Review of Previous Work

- The first implementation has been suggested by Nievergelt and led to multiple shooting methods (1964). Variants of this method were then developed by several authors, in particular by Chartier and Philippe (1993).
- Lions, Maday and Turinici proposed a new version the “parareal algorithm” in 2001, updated by Maday and Bal in 2002, Farhat-Chandesris in 2003 and Tromeur-Dervout-Guibert in 2005.
- Farhat et al in 2005 proposed a change to improve the performance of the parareal algorithm for second-order hyperbolic systems.

Rescaling Methodology...

Find $y : [0, T_b) \to \mathbb{R}$, $0 < T_b \leq \infty$, such that:

(1) $$\begin{cases} \frac{dy}{dt} = f(y), \, 0 < t < T_b \\ y(0) = y_0 > 0. \end{cases}$$

(2) $$t = T_{n-1} + \beta_n s, \; y(t) = y_{n-1} + \alpha_n z(s).$$

(3) $$\begin{cases} \frac{dz}{ds} = g_n(z) := \frac{\beta_n}{\alpha_n} f(y_{n-1} + \alpha_n z(s)), \, 0 < s < s_n \\ z(0) = 0. \\ z(s_n) = S. \end{cases}$$

(4) $y(T_n) = y_{n-1} + y_{n-1}S = y_{n-1}(1+S) \equiv y_n = (1+S)^n; \; T_n = T_{n-1} + \beta_n s_n.$

(5) $$\frac{y_n}{y_{n-1}} = \frac{y(T_n)}{y(T_{n-1})} = 1 + S.$$

Rescaling Methodology

Two main advantages:

1. Extremely accurate numerical approach, since each IVP (3) can be solved up to any precision, with starting homogeneous condition $z(0) = 0$ and a fixed ending condition $z(s_n) = S$ (met with an adaptive procedure)
2. The sequence of IVP (3) are defined independently and can be implemented either sequentially or in parallel.

RAPTI for Diffusion-Reaction Problems...

Consider the Cauchy problem where one seeks a vector function $U : [0, \infty) \to \mathbb{R}^k$, that verifi es:

(6) $$\begin{cases} \frac{dU}{dt} = AU + F(U) \\ U(0) = U_0 \in \mathbb{R}^k \end{cases}$$

(7) $$\begin{cases} \frac{\partial u}{\partial t} - \Delta u = au^p,\ p \leq 1, a > 0 & \text{for } x \in \Omega =]-1;1[\text{ and } 0 < t < \infty \\ u(x,t) = 0 & \text{for } x \in \partial\Omega \text{ and } 0 < t < \infty \\ u(x,0) = p(x) > 0 & \text{for } x \in \Omega. \end{cases}$$

RAPTI for Diffusion-Reaction Problems...

(8)
$$\begin{cases} U(t) = U_{n-1} + diag(\alpha_n)Z(s) & \alpha_n \in \mathbb{R}^k \\ t = T_{n-1} + \beta_n s & 0 \leq s \leq s_n, T_{n-1} \leq t \leq T_n \end{cases}$$

(9)
$$\begin{cases} \beta_n = \frac{1}{||U_{n-1}||_\infty^{p-1}} \\ \alpha_n = U_{n-1} \end{cases}$$

(10)
$$\begin{cases} \frac{dZ}{ds} = G_n(Z) = F(U_{n-1} + diag(\alpha_n)Z(s)) & 0 \leq s < s_n \\ Z(0) = 0 \\ \|Z(s_n)\| = S & \text{(S being the cut-off value)} \end{cases}$$

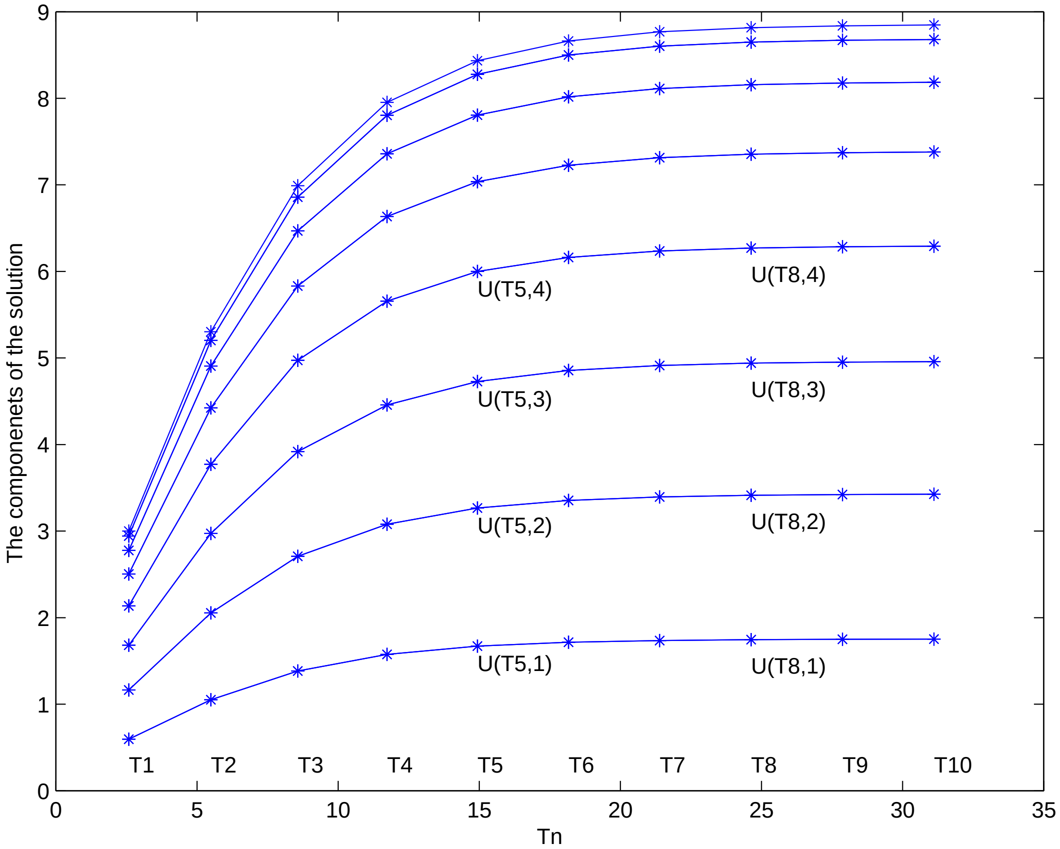
$$U_n = U(T_n) = U_{n-1} + diag(U_{n-1})Z(s_n) = diag(U_{n-1})(e + Z(s_n));\ T_n = T_{n-1} + \beta_n s_n.$$
(11)

(12)
$$r_n = (e + Z(s_n)).$$

(13)
$$U_n = diag(r_n)U_{n-1}.$$

RAPTI for Diffusion-Reaction Problems

Time versus the solution U_n for the diffusion reaction problem, where each solution is a vector consisting of 8 components.



RAPTI for diffusion-reaction problem: Algorithm

1. Sequential run on n_s slices, until transition vector r_n stabilizes.

2. For following slices, $n > n_s$, estimate ratios r_n^p, based on r_n^e $(n \leq n_s)$ and get **predicted initial values** for each slice $n > n_s$.

3. Execute parallel computations on each n^{th} slice, $n > n_s$, with starting value U_{n-1}^p leading to an end value U_n^c. Compute the sequence G_n of gaps at the end of each n^{th} slice, as $G_n = U_n^c - U_n^p$.

4. Test convergence: $n_s < n \leq n_{conv}$, $||G_n||_\infty \leq \epsilon_{tol}^g$.

5. Update n_s by n_{conv} and repeat steps 3 to 5 until a maximum time T is reached.

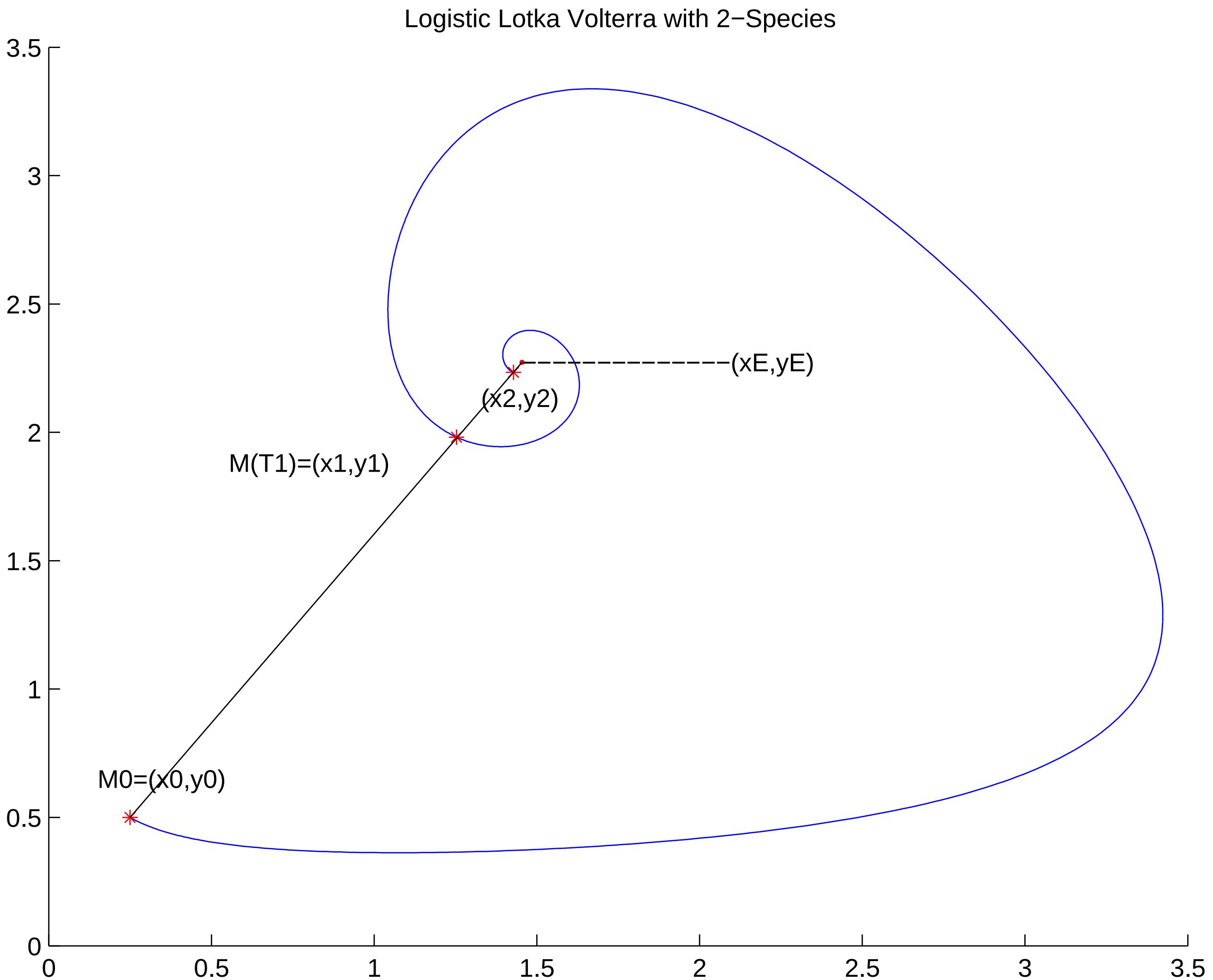
RAPTI for Two Species Logistic Lotka-Volterra...

(14)
$$\begin{cases} \frac{dx}{dt} = ax - bxy - ex^2, \\ \frac{dy}{dt} = -cy + dxy - fy^2, \\ x(0) = x_0,\ y(0) = y_0. \end{cases}$$

where $a, b, c, d, e, f, x_0, y_0$ are given positive parameters. The stable equilibrium point of the above system is given by:

$$M_E = (x_E, y_E) = \left(\frac{af+bc}{ef+bd}, \frac{ad-ec}{ef+bd}\right)$$

RAPTI for Two Species Logistic Lotka-Volterra



RAPTI for 2-Species Logistic Lotka-Volterra: Rescaling

(15) $$\begin{cases} \frac{dU}{dt} = AU + F(U) \\ U(0) = U_0 \end{cases}$$

with $t = T_{n-1} + \beta_n s,\ U(t) = U_{n-1} + diag(\alpha_n)Z(s)\ (\alpha_n \in \mathbb{R}^2)$

$$U = \begin{pmatrix} x \\ y \end{pmatrix}, U_0 = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}, A = \begin{pmatrix} a & 0 \\ 0 & -c \end{pmatrix}, F(x,y) = \begin{pmatrix} -bxy - ex^2 \\ dxy - fy^2 \end{pmatrix}$$

RAPTI for 2 Species Logistic Lotka-Volterra: Rescaling

By selecting $\alpha_n = U_{n-1}$, $\beta_n = 1$, then $Z(s)$ verifi es:

(16)
$$\begin{cases} \frac{dZ}{ds} = diag(\frac{1}{\alpha_n})A(U_{n-1} + diag(\alpha_n)Z(s)) + diag(\frac{1}{\alpha_n})F(U_{n-1} + diag(\alpha_n)Z(s)), 0 < s < s_n \\ Z(0) = 0 \\ Arg(M_E\overrightarrow{M}(T_n)) = Arg(M_E\overrightarrow{M}(T_{n-1})) + 2\pi \end{cases}$$

Stopping criterion is based on fi nding a “period” T such that:

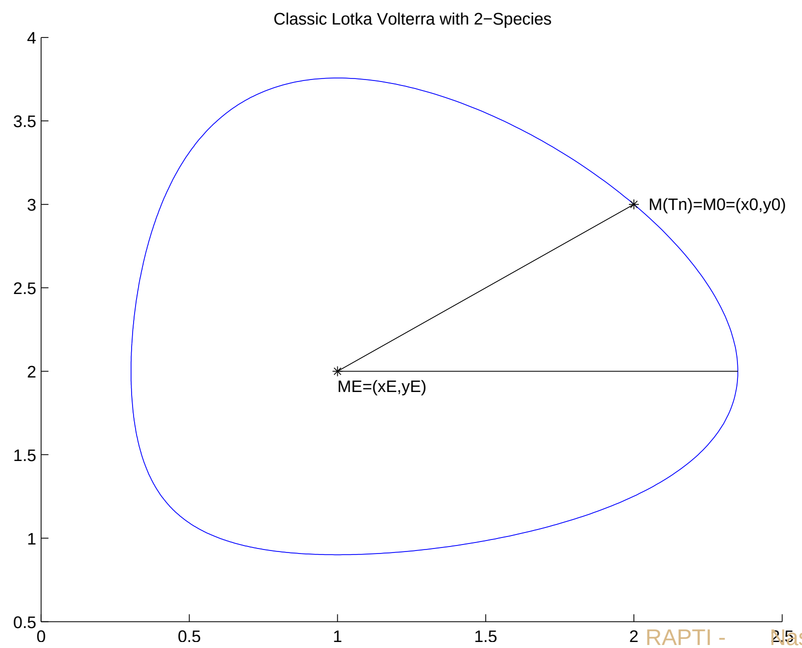
(17)
$$Arg(M_E\overrightarrow{M}(nT)) = Arg(M_E\overrightarrow{M}(0)) + 2\pi \, \forall n \geq 1$$

RAPTI for 2-species Logistic Lotka-Volterra: Defining the ratios

(18)
$$r_n = \frac{\left|\overrightarrow{M_E M}(T_n)\right|}{\left|\overrightarrow{M_E M}(T_{n-1})\right|}, n = 1, 2, \cdots$$

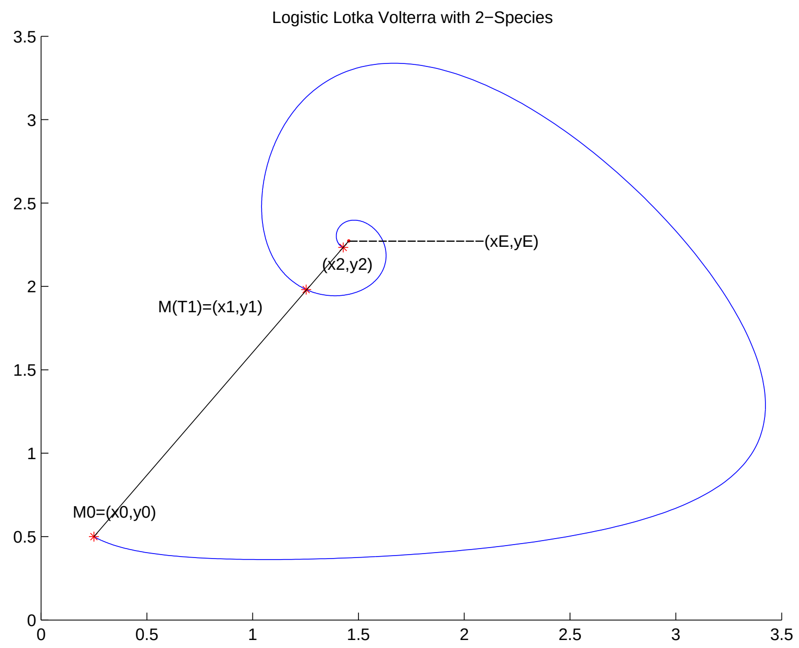
Choice based on the fact that in the classic Lotka-Volterra model, the ratios

$$r_n = \frac{\left|\overrightarrow{M_E M}(T_n)\right|}{\left|\overrightarrow{M_E M}(T_0)\right|} = 1 \ \forall n$$



RAPTI for 2-Species Logistic Lotka-Volterra: Stopping criterion

The trajectories of the Logistic Lotka-Volterra with 2-species in the xy-plane circulate counterclockwise and spiral inwards towards the equilbrium point (x_E, y_E). The ratios stabilize $r_n = \frac{\left|M_E\overrightarrow{M}(T_n)\right|}{\left|M_E\overrightarrow{M}(T_{n-1})\right|} \approx C$ for $n \geq n_s$ allowing us to predict the initial values at the beginning of each slice.



RAPTI for 2-Species Logistic Lotka-Volterra: Algorithm

1. Sequential run on n_s slices, until the ratios r_n stabilize
2. Determine the total number of slices n_t given by:
   $$n_t = n_s + \left\lfloor \frac{\log \frac{\epsilon^s_{tol}}{r_{n_s}}}{\log C} \right\rfloor$$
3. For following slices, $n > n_s$, estimate ratios r^p_n, based on r^e_n ($n \leq n_s$) and get **predicted initial values** for each slice $n > n_s$ given by:
   U^p_n $= M_E + C^{n-n_s} \, |U_{n_s} - M_E| \, sign(U_{n_s} - M_E)$
   using the following conditions:
   - $\forall n \; M_E\overrightarrow{M}(T(n)$ has the same direction of $M_E\overrightarrow{M}(T(0)$
   - $|M_E M(T(n)|^p = |M_E M(T(n_s)| \times (C)^{n-n_s} \; \forall n > n_s$
4. Execute parallel computations on each n^{th} slice, $n > n_s$, with starting value U^p_{n-1} leading to an end value U^c_n. Compute the sequence G_n of gaps at the end of each n^{th} slice, as $G_n = U^c_n - U^p_n$.
5. Test convergence: $n_s < n \leq n_{conv}$, $||G_n||_\infty \leq \epsilon^g_{tol}$.
6. Update n_s by n_{conv} and repeat steps 2 to 6 until a maximum time T is reached.

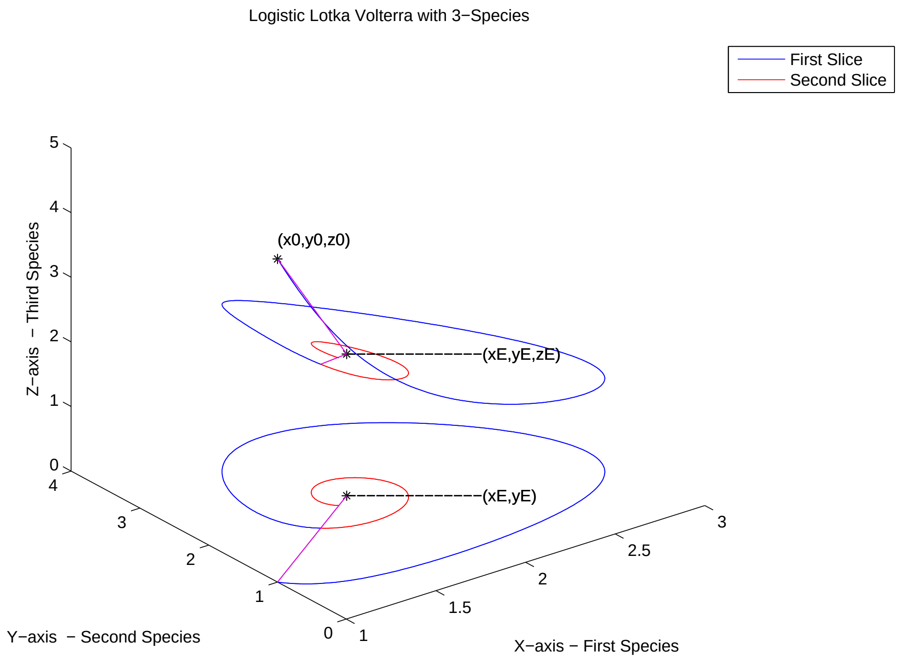
RAPTI for 3-Species Logistic Lotka-Volterra...

$$(19) \quad \begin{cases} \frac{dx}{dt} = ax - bxy - hx^2 \\ \frac{dy}{dt} = -cy + dxy - eyz - iy^2 \\ \frac{dz}{dt} = -fx + gyz - jz^2 \\ x(0) = x_0, \; y(0) = y_0, \; z(0) = z_0. \end{cases}$$

where $a, b, c, d, e, f, g, h, i, j, x_0, y_0, z_0$ are given positive parameters.
A study of the equilibrium points of the system lead to 8 of them, of which there exists a unique stable point given by:

$$N_E = (x_E, y_E, z_E) = \left(\frac{bjc - bfe + age + aij}{ghe + hij + dbj}, \frac{-jhc + jda + fhe}{ghe + hij + dbj}, \frac{-hif - ghc + gda - dbf}{ghe + hij + dbj}\right)$$

RAPTI for 3-Species Logistic Lotka-Volterra



The upper part represents the trajectories of the Logistic Lotka-Volterra with 3-species in the xyz-plane. The lower part represents the projection of the solution in the xy-plane which has a similar behaviour to that of the 2-species logistic Lotka-Volterra system. The ratios $r_n = \frac{\left|M_E \overrightarrow{M}(T_n)\right|}{\left|M_E \overrightarrow{M}(T_{n-1})\right|} \approx C, \rho_n = \frac{\left|N_E \overrightarrow{N}(T_n)\right|}{\left|N_E \overrightarrow{N}(T_{n-1})\right|} \approx D$ for $n \geq n_s$

RAPTI for 3-species Lotka-Volterra: Algorithm

- Sequential run on n_s slices, until the ratios r_n and ρ_n stabilize
- Determine the total number of slices n_t given by
$$n_t = n_s + max\left(\left\lfloor \frac{\log \frac{\epsilon_{tol}^s}{r_{n_s}}}{\log C} \right\rfloor, \left\lfloor \frac{\log \frac{\epsilon_{tol}^s}{\rho_{n_s}}}{\log D} \right\rfloor\right)$$
- For following slices, $n > n_s$, estimate ratios r_n^p and ρ_n^p, based on r_n^e and r_n^e ($n \leq n_s$) and get **predicted initial values** for each slice $n > n_s$ given by:
U_n^p =$M_E + C^{n-n_s}\, |U_{n_s} - M_E|\, sign(U_{n_s} - M_E)$
using the following conditions:
  - $\forall n \ M_E\overrightarrow{M}(T(n)$ has the same direction of $M_E\overrightarrow{M}(T(0)$
  - $|M_E M(T(n)|^p = |M_E M(T(n_s)| \times (C)^{n-n_s} \ \forall n > n_s$
  - $|N_E N(T(n)|^p = |N_E N(T(n_s)| \times (D)^{n-n_s} \ \forall n > n_s$

  lead to predict the starting value at every slice n,$\forall n > n_s$ (Note that for $n = n_s + 1$, $U_{n-1}^p = U_{n-1}^e$)
- Execute parallel computations on each n^{th} slice, $n > n_s$, with starting value U_{n-1}^p leading to an end value U_n^c. Compute the sequence G_n of gaps at the end of each n^{th} slice, as $G_n = U_n^c - U_n^p$.
- Test convergence: $n_s < n \leq n_{conv}$, $||G_n||_\infty \leq \epsilon_{tol}^g$.
- Update n_s by n_{conv} and repeat steps 2 to 6 until a maximum time T is reached.

Numerical Results

List of tolerances used in the *RaPTI* algorithm.

Tolerance	**Functionality**
ϵ^r_{tol}	Determine n_s
ϵ^g_{tol}	Determine number of iterations
ϵ^l_{tol}	Determine stopping criteria within a slice
ϵ^s_{tol}	Determine number of slices

Diffusion Reaction-1

The results of running the diffusion reaction problem:

$$(20)\begin{cases} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = au^p,\ p \le 1, a > 0,\ x \in]-1;1[,\ 0 < t < \infty \\ u(x,t) = 0 \quad \text{for } x \in \partial\Omega \text{ and } 0 < t < \infty \\ u(x,0) = p(x) > 0 \quad \text{for } x \in \Omega. \end{cases}$$

$$\begin{cases} p(x) = 2 * x + 2 & \text{for } x \le -0.5 \\ p(x) = \frac{-2}{3} * x + \frac{-2}{3} & \text{for } x \succ -0.5 \end{cases}$$

$$p = 0.8, h = \frac{1}{8}, S = 2, a = 3$$

Numerical Results

Diffusion Reaction-2

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	**Number of Slices**	**Iterations**
17	10^-4	10^-8	10^-8	64	21
17	10^-4	10^-8	10^-8	128	23
17	10^-4	10^-8	10^-8	256	24
24	10^-6	10^-8	10^-8	64	14
24	10^-6	10^-8	10^-8	128	16
24	10^-6	10^-8	10^-8	256	17
32	10^-8	10^-8	10^-8	64	6
32	10^-8	10^-8	10^-8	128	8
32	10^-8	10^-8	10^-8	256	9
36	10^-9	10^-8	10^-8	64	2
36	10^-9	10^-8	10^-8	128	4
36	10^-9	10^-8	10^-8	256	5

Numerical Results

2-species Logistic Lotka Volterra(1)

$a = 1, b = 1, c = 1, d = 1, e = \frac{1}{2}, f = \frac{1}{5}, x_0 = 10, y_0 = 5, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
3	10^{-2}	10^{-8}	10^{-8}	10^{-14}	12	1
3	10^{-3}	10^{-8}	10^{-8}	10^{-14}	12	1

$a = 1, b = 1, c = 1, d = 1, e = \frac{1}{2}, f = \frac{1}{5}, x_0 = 1, y_0 = 5, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
3	10^{-2}	10^{-8}	10^{-8}	10^{-14}	12	1
4	10^{-3}	10^{-8}	10^{-8}	10^{-14}	12	1

Numerical Results

2-species Logistic Lotka Volterra(2)

$a = 2, b = 1, c = 1, d = 1, e = \frac{1}{2}, f = \frac{1}{5}, x_0 = 1, y_0 = 5, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
3	10^{-2}	10^{-8}	10^{-8}	10^{-14}	16	3
4	10^{-3}	10^{-8}	10^{-8}	10^{-14}	16	1

$a = 2, b = 1, c = 1, d = 2, e = \frac{1}{2}, f = \frac{1}{5}, x_0 = 1, y_0 = 5, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
4	10^{-2}	10^{-8}	10^{-8}	10^{-14}	24	5
5	10^{-3}	10^{-8}	10^{-8}	10^{-14}	24	3

Numerical Results

3-species Logistic Lotka Volterra(1)

$a = 3, b = 1, c = 1, d = 1, e = 1, f = 1, g = 1, h = \frac{1}{2}, i = \frac{1}{5}, j = \frac{1}{2}, x_0 = 1, y_0 = 1, z_0 = 2, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
3	10^-2	10^-8	10^-8	10^{-14}	18	5
7	10^-3	10^-8	10^-8	10^{-14}	18	1

$a = 3, b = 1, c = 1, d = 2, e = 1, f = 1, g = 1, h = \frac{1}{2}, i = \frac{1}{5}, j = \frac{1}{2}, x_0 = 1, y_0 = 1, z_0 = 2, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
3	10^-2	10^-8	10^-8	10^{-14}	25	10
8	10^-3	10^-8	10^-8	10^{-14}	25	2

Numerical Results

3-species Logistic Lotka Volterra(2)

$a = 3, b = 1, c = 1, d = 1, e = 1, f = 1, g = 1, h = \frac{1}{2}, i = \frac{1}{5}, j = \frac{1}{2}, x_0 = 1, y_0 = 2, z_0 = 5, \tau = \frac{1}{2^7}$

n_s	ϵ^r_{tol}	ϵ^g_{tol}	ϵ^l_{tol}	ϵ^s_{tol}	Nb of Slices	Iter
4	10^-3	10^-8	10^-8	10^{-14}	18	4
12	10^-4	10^-8	10^-8	10^{-14}	18	1