MAP551 - PC5: Numerical Integration of Ordinary Differential Equation (Part III)

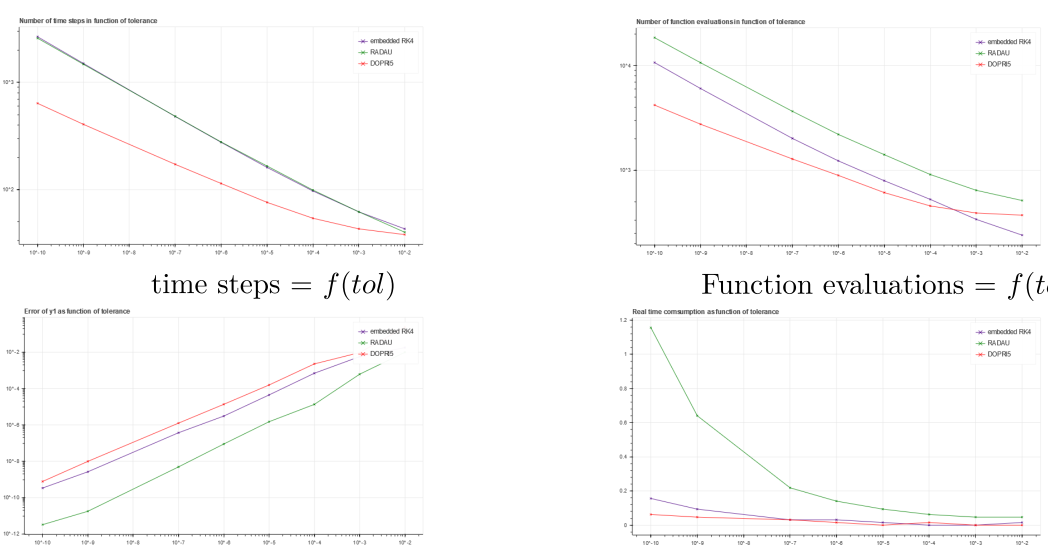
Te SUN

November 2018

1 Use of RADAU solver

1.1 Integration of Brusselator model

Shown as the figure 1, we can see on the upper right, the time steps in function of tolerance. Besides of the representation of RADAU5 method, we show, on the same time, the performance in terms of time steps of methods DOPRI5 and embedded RK4. (**2.1.1**) However, as for the fixed time-step RK4 method, we could not set the tolerance in advance. Consequently, a discussion around the time steps as function of accuracy will be made later.



time steps $= f(tol)$ | Function evaluations $= f(tol)$

$||Error_{y1}|| = f(tol)$ | Time consumption $= f(tol)$

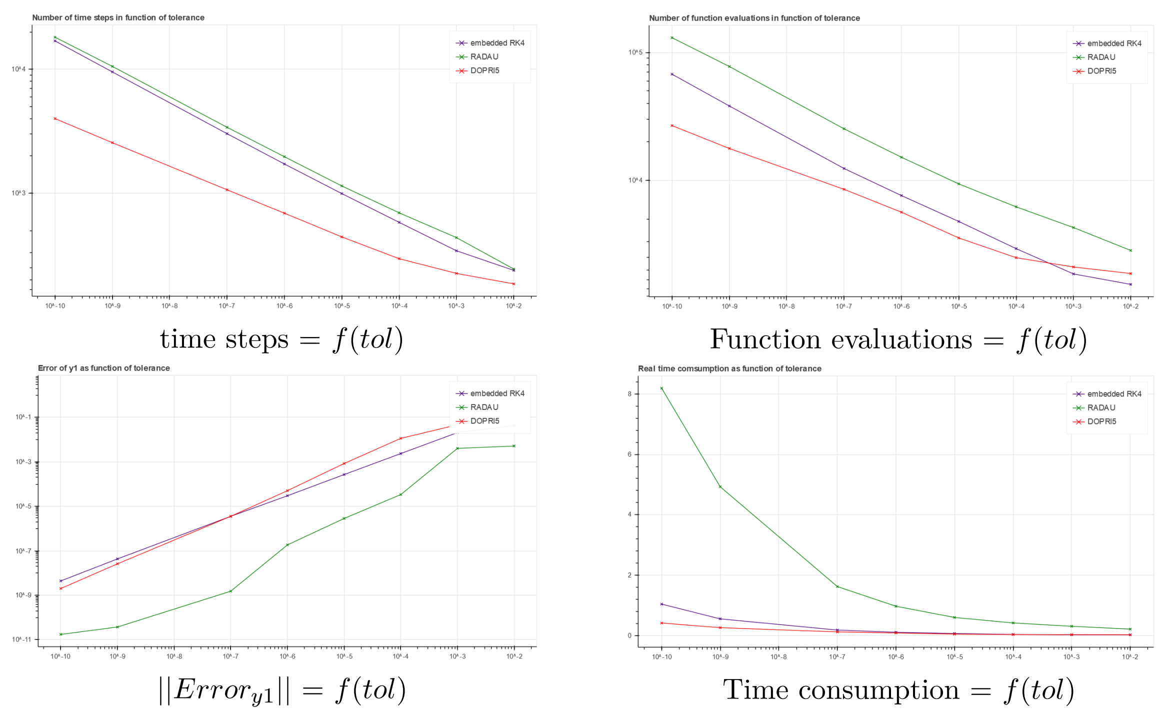
Figure 1 – Comparison from different aspect of solver : RK embed(purple), RADAU(green), DOPRI5(red)

We continue to discuss the graphic result in 1 : **upper right** shows the time steps in function of tolerance. With a more demanded accuracy, the needed time steps increase to ensure the satisfaction of tolerance condition. The embedded and RADAU method have almost the same performance while the DOPRI5 needs a bit less of time steps. As for function evaluation times, **upper left** shows that the DOPRI5 prevails again. And the RADAU method presents nothing better than the other two. When we concentrate on the accuracy, for instant, we could observe the error for y_1. It comes as no surprise that, according to **bottom left**, the RADAU method wins since it is an implicit method which involves more stability and accuracy proprieties. However, if we look at the real time consumption. **bottom right** figure shows the implicity of RADAU, which introduces a huge computational effort to solve equations. Untill then, we can see with a "not so big gap" in terms of function evaluation times, the RADAU method needs a great amount of time comparing to the other two methods. That is due to the resolution of the non-linear system, which includes calculation of Jacobean matrix, and iteration of Newton resolution. **2.1.2-2.1.3**

Therefore, to solve this dynamical system, the implicit method, i.e. RADAU5, takes significantly an extra effort to reach a not very remarkable improvement of accuracy. **Then, for such a mildly stiff model, we might have a preference towards the explicit methods.** Because even with a much more demanded tolerance, they could save us much time. Yet, when we need a very precis result, we could also try the implicit method to have more accuracy with the same resolution tolerance. (**2.1.4**)

1.2 Integration of van der Pol oscillator model

With a mild stiffness, similar to the previous results, **DOPRI5 needs less time steps to solve numerically the system. The implicit RADAU calculates more often the evaluation function and gains more accuracy**.



time steps $= f(tol)$ Function evaluations $= f(tol)$

$||Error_{y1}|| = f(tol)$ Time consumption $= f(tol)$

Figure 2 – $\varepsilon = 1$, Comparison from different aspects of solver : RK embed(purple), RADAU(green), DOPRI5(red)

When we look at the fixed time step RK4 method, **it always needs more time steps to**

obtain results with same accuracy. (**2.2.1-2.2.2**)

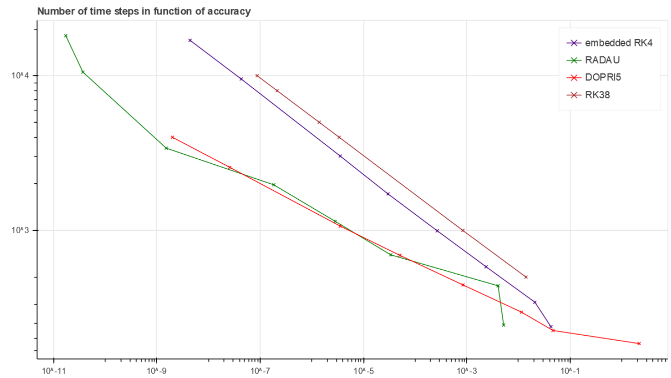
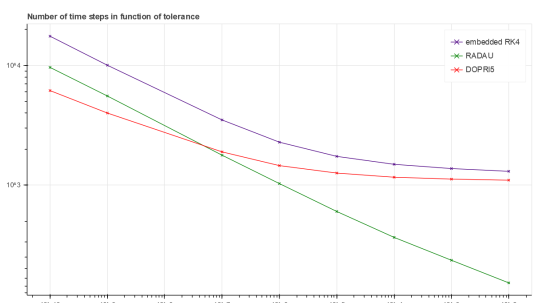
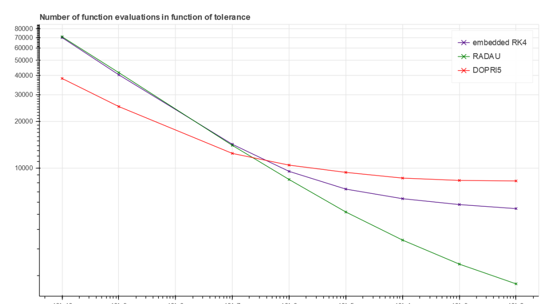


Figure 3 – $\varepsilon = 1$, number of time steps in function of accuracy for y_1 : fixed time step RK4(brown), RK embed(purple), RADAU(green), DOPRI5(red)

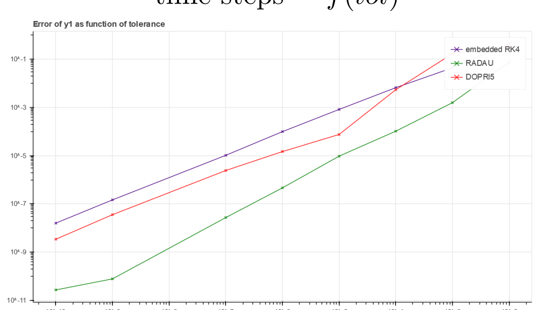
If we increase the stiffness of the system, the advantage of the implicit method appears : with $\epsilon = 20$, we can see that with same tolerance, the accuracy of implicit RADAU method allows less time steps and less function evaluation times, which narrows the gap of time consumption between implicit and explicit methods.



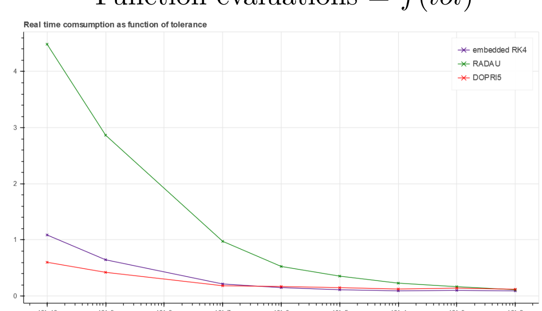
time steps $= f(tol)$



Function evaluations $= f(tol)$



$||Error_{y1}|| = f(tol)$



Time consumption $= f(tol)$

Figure 4 – $\varepsilon = 20$, Comparison from different aspects of solver : RK embed(purple), RADAU(green), DOPRI5(red)

By comparing the time steps as a function of accuracy, the advantage of RADAU is even more remarkable. And the fixed step RK38 is no more usable to solve the problem. **2.2.2**

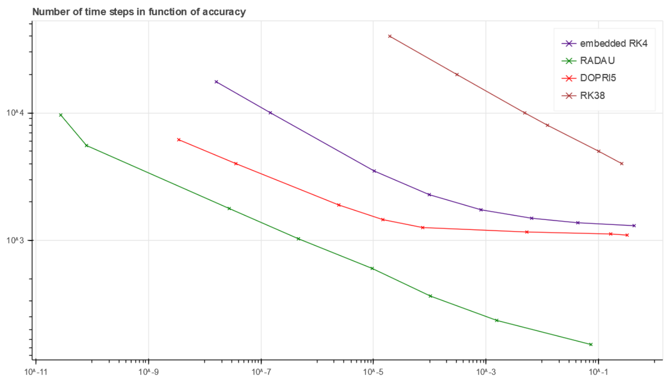
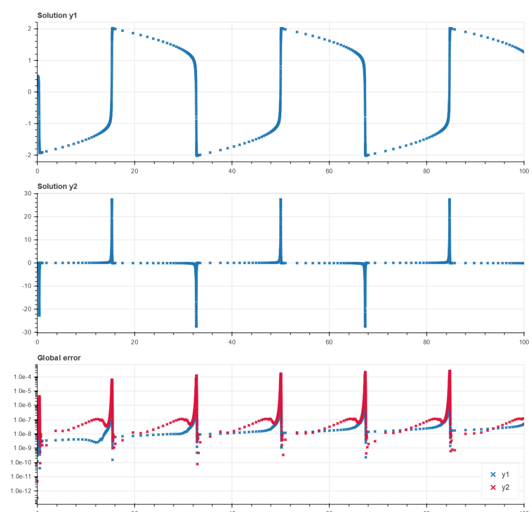


Figure 5 – $\varepsilon = 20$, number of time steps in function of accuracy for y_1 : fixed time step RK4(brown), RK embed(purple), RADAU(green), DOPRI5(red)



y_1, y_2 and global error

Here, we take the RADAU method as example : **even with an important stiffness, the global error does not accumulate. That is thanks to the fact that the system is dissipative.** The tolerance and the global error is positively correlated, when we decrease the tolerance, the global error does as well (bottom left of Fig 4). (**2.2.3**)

Therefore, when the stiffness is mild, we have plenty of reasons using the explicit methods : to **use less memory and time** while obtaining a high enough precision. In the contrary, **when the stiffness is important, the implicit method could be time saving** to solve the same problem with same tolerance. (**2.2.4**)

2 Operator splitting techniques for stiff ODEs

The operator splitting strategy for ODEs will be experimented on the Oregonator system of equations. The system of ordinary differential equations reads :

$$
\begin{aligned}
\mathrm{d}_\tau y_1 &= y_2 - y_1, && (1)\\
\epsilon\, \mathrm{d}_\tau y_2 &= q\, y_3 - y_3\, y_2 + y_2(1 - y_2), && (2)\\
\mu\, \mathrm{d}_\tau y_3 &= -q\, y_3 - y_3\, y_2 + f\, y_1, && (3)\\
& && (4)
\end{aligned}
$$

with paremeters

$$\epsilon = 10^{-2}, \quad \mu = 10^{-6}, \quad f = 3, \qquad q = 2.10^{-4}. \tag{5}$$

In general $\mu \ll \epsilon \ll 1$ and $q \ll 1$.

2.1 Singular perturbation analysis and various forms of the system involving two operators

The system can be rewritten as follow (**3.1.1**) :

$$\mathrm{d}_\tau \mathcal{U} = \mathrm{d}_\tau \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} y_2 - y_1 \\ \frac{1}{\varepsilon}\Big(q\, y_3 - y_3\, y_2 + y_2(1-y_2)\Big) \\ 0 \end{bmatrix} + \frac{1}{\mu}\begin{bmatrix} 0 \\ 0 \\ -q\, y_3 - y_3\, y_2 + f\, y_1 \end{bmatrix}$$

In the limit $\mu \to 0$, the second term $\mathcal{B}(\mathcal{U})$ must be zero or else, the velocity of the system will be infinity. As a result, the only remaining term lets us reformulate the equation as :

$$\mathrm{d}_\tau \begin{bmatrix} \bar{y_1} \\ \bar{y_2} \\ \bar{y_3} \end{bmatrix} = \begin{bmatrix} \bar{y_2} - \bar{y_1} \\ \frac{1}{\varepsilon}\Big(q\, \bar{y_3} - \bar{y_3}\, \bar{y_2} + \bar{y_2}(1-\bar{y_2})\Big) \\ 0 \end{bmatrix}$$

Because $\mathrm{d}_\tau \bar{y_3} = 0$ and $\mathcal{B}(\mathcal{U}) = 0$, y_3 is a constant and can be written as :

$$y_3 = \frac{f\, y_1}{q + y_2}$$

By replacing y_3 with the former result, we can obtain the demanded equations. **3.1.2**

We have discussed that when the system is not very stiff the explicit method can be well adapted to resolve the system. Here, we could choose the **DOPRI5 method for the slow dynamics**. However we should use the **implicit RADAU** to guarantee the accuracy for the full system.

As we expected, the smaller μ is, the more accurate the approximation will be. We show in the follow two solution of y_1 :

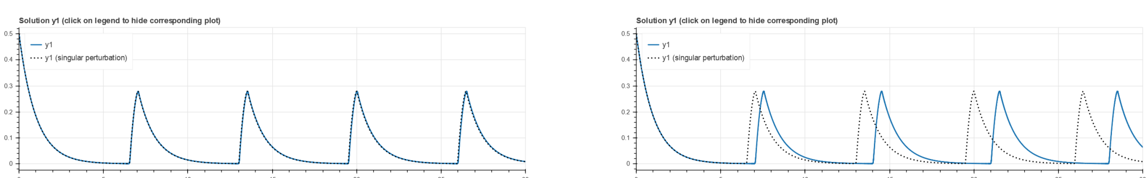


Figure 6 – y_1 with $\mu = 10^{-6}$ and $\mu = 10^{-4}$

Visually, when $\mu = 10^{-4}$, the approximation is no more acceptable. If we calculate the global error (norm 2) of y_1, $\mu = 10^{-6}$ leads us to an error of 0.00393, while $\mu = 10^{-4}$ gives us an error of 0.09711 for $t \in [0, 30]$.(**3.1 2**)

$$\mathrm{d}_\tau \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \underbrace{\begin{bmatrix} y_2 - y_1 \\ \frac{1}{\varepsilon}\Big(y_3^{eq}(y_1, y_2)(q - y_2) + y_2(1-y_2)\Big) \\ 0 \end{bmatrix}}_{\mathcal{A}_2} + \underbrace{\frac{1}{\mu}\begin{bmatrix} 0 \\ \frac{\mu}{\varepsilon}\Big((y_3 - y_3^{eq})(q - y_2) + y_2(1-y_2)\Big) \\ -q\, y_3 - y_3\, y_2 + f\, y_1 \end{bmatrix}}_{\mathcal{B}_2}$$

We suppose then the $\mathcal{A}_2$ has a slow dynamic. As a result the $\mathcal{B}_2$ represents the stiff part. **The hypothesis of stiffness of second part is valid only when** $\varepsilon << 1$. If not, the second component of $\mathcal{B}_2$ is still very influential to the slow dynamical term. And this reformulation reduce stiffness of ordinary differential equations when $\mu << \varepsilon$. Imagine when $\mu >> \varepsilon$, we would create a stiffness term by introducing the second component of $\mathcal{B}_2$: $\frac{\mu}{\varepsilon}\Big((y_3 - y_3^{eq})(q - y_2) + y_2(1 - y_2)\Big)$ **(3.1.3)**

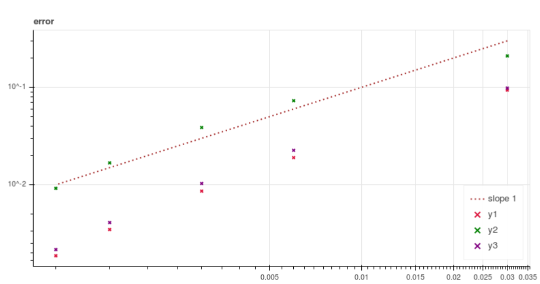
2.2 Integration of the system using operator splitting

1. First Lie formulae : To obtain $\mathcal{U}^{n+1}$, we first integrate during dt and only for y_1, the slow dynamics (supposing $\mathrm{d}_\tau\mathcal{U} = \mathcal{A}_1(\mathcal{U})$) using radau5 method, result as $\mathcal{U}^{n+1/2}$; Then we take $\mathcal{U}^{n+1/2} = (y_1^{n+1/2}, y_2^n, y_3^n)^t$ as initial condition to integrate stiff term for y_2 and y_3 (supposing $\mathrm{d}_\tau\mathcal{U} = \frac{1}{\mu}\mathcal{B}_1(\mathcal{U})$), which gives by radau5 method the $\mathcal{U}^{n+1}$.
2. First Strang formulae : We take $\mathcal{U}^n = (y_1^n, y_2^n, y_3^n)$ as initial datum and firstly integrate the stiff part during $\frac{1}{2}dt$ to obtain only the evolution of y_3, noted $y_3^{n+1/3}$. Then integrate for (y_1, y_2) the $\mathcal{A}_1$ during dt (while $y_3 remains unchanged$), and finally integration for y_3 of $\mathrm{d}_\tau\mathcal{U} = \frac{1}{\mu}\mathcal{B}_1(\mathcal{U})$ during the last $\frac{1}{2}dt$
3. Second Lie formulae with the same logic, we integrate the plate part and then the stiff term.
4. Second Strang, we integrate B_2 within the first $0.5\,dt$, then A_2 during dt and B_2 for the last $0.5\,dt$

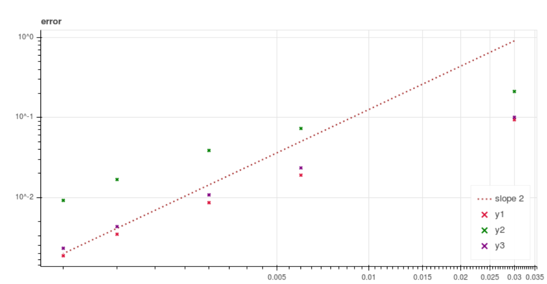
In general, all methods integrate during a $2dt$ period. And **we use every time the integrator RADEAU5 during one integration**. **And we calculate Jacobean matrix after every** $0.5dt$. Therefore, for a dt time integration, we calculate Jacobean two times. **3.2.1**.

We take $\mu = 10^{-4}$ that is absolutely not acceptable for a singular perturbation analysis as discussed before. Here, the method that we used is Lie for first splitting strategy.

So, we show below the relation between number of time steps and the global error for three variables. In the aim of inspecting the order of the method, we use the step length as the X axis :



Lie splitting

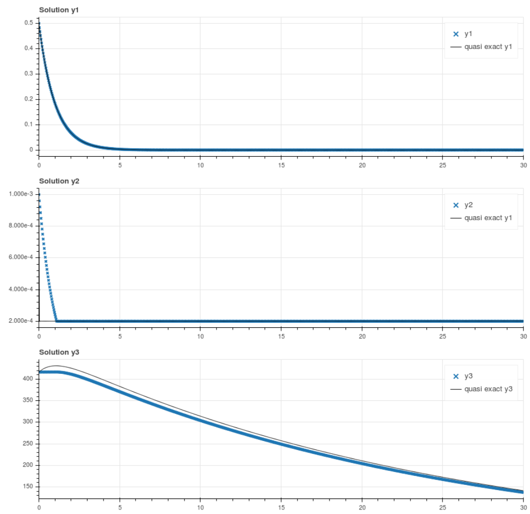


Strang splitting

Figure 7 – Global error in function of step length for different Splitting

We can see from figure 7 that the Lie method is a first order while the Strang is of second order. 3.2.3

Although the method are theoretically and respectively order 1 and 2, we can see the error function is not exactly parallel to line with slope 1 (or 2). That might be resulted from the non-commutative propriety between $\mathcal{A}$ and $\mathcal{B}$. And the value of $\mathcal{A}\,\mathcal{B} - \mathcal{B}\,\mathcal{A}$ evaluate through time. So the order of method is only asymptotically valid.



numerical solution for y_1, y_2 and y_3 with second split strategy and strang method ; $\mu = \varepsilon = 10^{-2}$

As speculated, when μ is too large to be reduced as two split part, we have created a non-existing stiffness (especially for y_2). Actually, the full dynamics are no more stiff when μ is large.

In conclusion, the splitting method could be taken only if the low and high dynamics behaviors are appropriately separated, that is, to reduce as much as possible the error introduced by the splitting. (3.2.3)

3 An ODE system derived from the semi-discretization in space of the heat equation

We will derive a discretization through the method of lines (MOL) of the heat equation :

$$\partial_t T - \partial^2_{xx} T = 0 \tag{6}$$

with Neumann boundary : $\partial_x T = 0$ on the boundary.

3.1 Properties of the original system of PDEs and of the discretized one

We try to obtain certain proprieties of $T(t, x)$ by finding its Fourier transformation.

$$\tilde{T}(t, s) = \int_{\mathbb{R}} e^{-2\pi i\, xs} T(t, x) \mathrm{d}x$$

Therefore

$$\partial_t \tilde{T}(t, s) + 4\pi^2 s^2 \tilde{T}(t, s) = 0$$
$$\tilde{T}(t, s) = \hat{g}(s) e^{-4\pi^2 s^2 t}$$

Where $\hat{g}(s) = \tilde{T}(0, s)$.

If we have only one Fourier mode, the temperature function becomes Dirac function. Knowing that the Fourier transform of function δ is 1. ($\hat{g}(s) = 1$) By the inverse of Fourier, the dynamic of single Fourier mode is :

$$T(x, t) = \frac{1}{2\sqrt{\pi t}} \exp\left(\frac{-x^2}{4t}\right), x \in \mathbb{R},\ t > 0$$

As well, we show the Fourier transformation of the initial data :

$$\begin{aligned}
\tilde{T}(0,s) &= \int_{\mathbb{R}} \frac{1}{2\sqrt{\pi t_0}} e^{-ixs} \exp(-x^2/(4t_0))\mathrm{d}x \\
\frac{dT}{dx}(0,x) &= -\frac{x}{2t_0} T(0,x) \\
\widetilde{\frac{dT}{dx}}(0,s) &= -(\widetilde{\frac{x}{2t_0}T})(0,s) \\
i\,s\,\widetilde{T}(0,s) &= -i/(2t_0)\frac{\mathrm{d}\widetilde{T}(0,s)}{\mathrm{d}s} \\
-2t_0\,s\,\widetilde{T}(0,s) &= \frac{\mathrm{d}\widetilde{T}(0,s)}{\mathrm{d}s} \\
\widetilde{T}(0,s) &= \exp(-t_0 s^2)
\end{aligned}$$

With the Fourier transform of initial data, we can see that if t_0 tends to 0, the non zero value of function T will centralize around the zero and tend to be Dirac function, which is a single mode Fourier mode. The irregularity of δ function introduce then the stiffness to the system. The smaller t_0 is, the stiffer the system is. 4.1.1

Stiffness can also arise in linear or linearized systems when eigenvalues exist with significantly different magnitudes. Since the timestep must be set so that both eigenvalues are stable, the larger eigenvalue will dominate the time step. Knowing that the largest eigenvalues (in absolute value) are related to the high frequency, therefore,

$$Stiffness = \frac{max|\lambda_m|}{min|\lambda_n|} = \frac{\sin^2(N\pi/2(N+1))}{\sin^2(\pi/2(N+1))} \approx_{N\to\infty} (\frac{2N+2}{\pi})^2$$

When N gets larger to refine the discretization of x, we add also the stiffness of the numerical solution in terms of time t. (4.1.2)

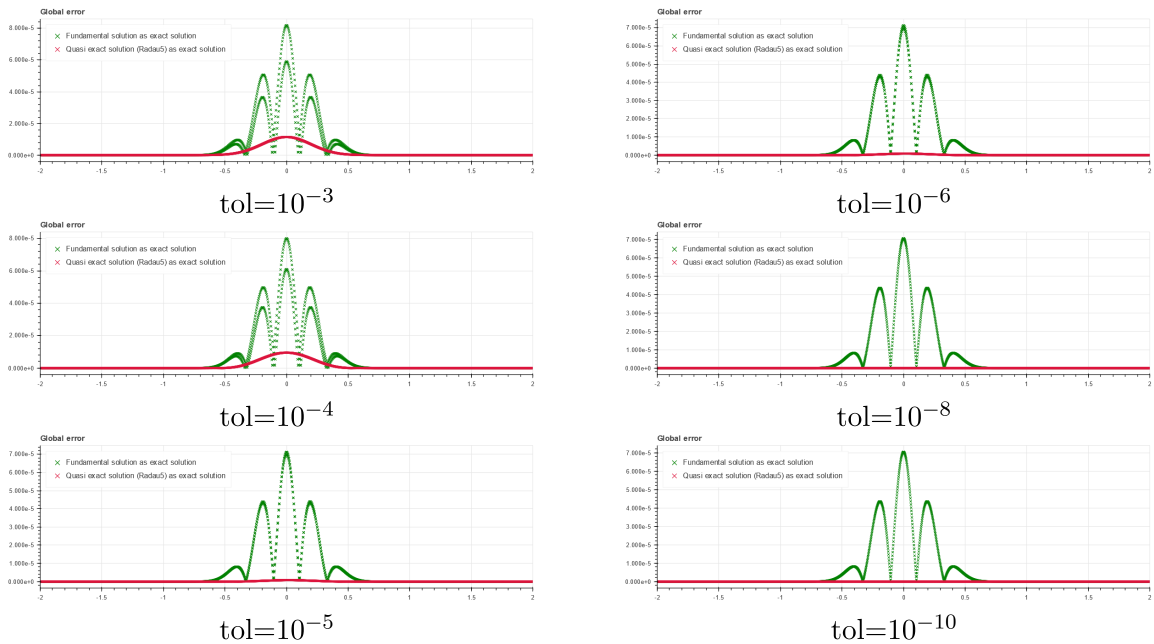
3.2 Integration of the dynamics for a given space discretization

At the beginning, tolerance is large, and the length time step is consequently big. Therefore, the error due to the time discretization is still strong and obvious. When we set a very fine tolerance (smaller than 10^{-6} for example), **there is no more significant improvement in terms of global error.** That should be a consequence of stiffness when we proceed the spatial discretization (the stiffness of the matrix A_N).

If we compare the ROCK4 with RADAU5, we can see again that the difference is no more visible when the tolerance is small enough. As we know, RADAU5 is an implicit method with great precision. So we can draw conclusion that the error is coming from the discretization stiffness rather than the method it self. (Fig 8). **4.2.1-4.2.2**

So another set synthetic comparison is made as below(Fig 10) :

It is exciting to see that the ROCK4 is even more accurate (upper left Fig 10) than implicit method. However, it is at expense of more function evaluations and more time steps. We can see that, in average, the ROCK4 is a 6.5 stages method. We know that ROCK4 is made to chose suitable stages, suitable time steps and solve the problem explicitly. We can then have a larger convergence domain. **4.2.2**

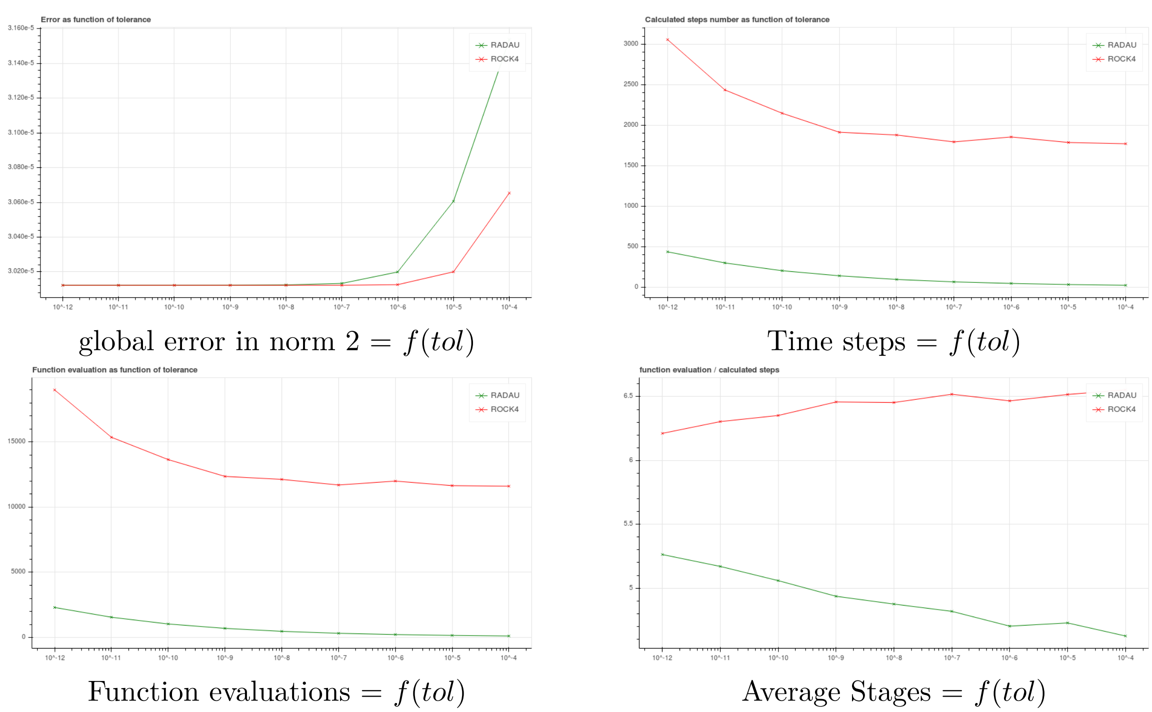


tol=10^{-3} tol=10^{-6}

tol=10^{-4} tol=10^{-8}

tol=10^{-5} tol=10^{-10}

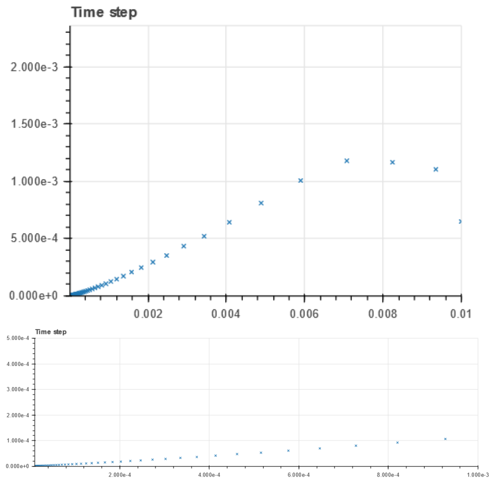
Figure 8 – Comparison between the ROCK4 solution and exact solution (Green), ROCK4 and RADAU5 method (Red)



global error in norm 2 = $f(tol)$ Time steps = $f(tol)$

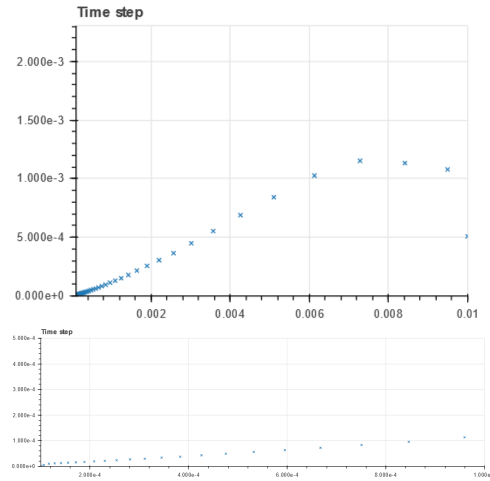
Function evaluations = $f(tol)$ Average Stages = $f(tol)$

Figure 9 – Comparison between ROCK4 and RADAU5 when $t_0 = 0.00001$



$t_0 = 10^{-5}$

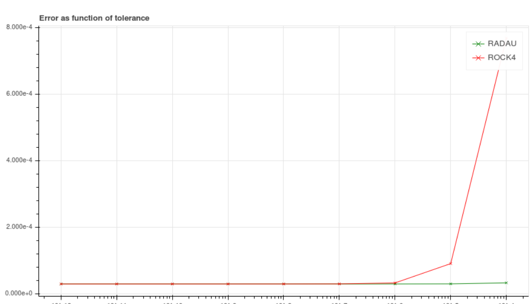
Number of function evaluations : 11996
Number of computed steps : 1855
Number of accepted steps : 1551
Number of rejected steps : 304
Maximum number of stage used : 10



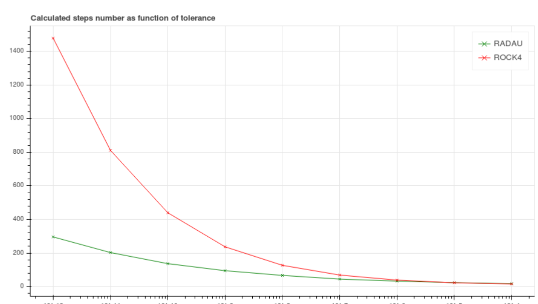
$t_0 = 10^{-4}$

Number of function evaluations : 965
Number of computed steps : 39
Number of accepted steps : 39
Number of rejected steps : 0
Maximum number of stage used : 63

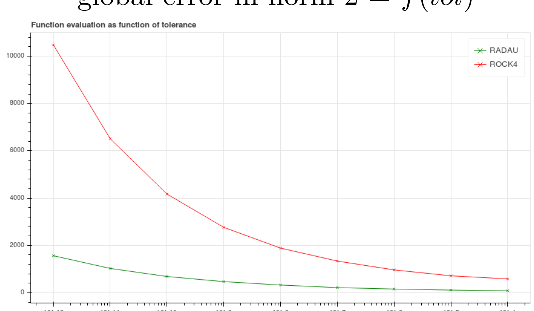
These two pictures (also with zoom of the interval $t \in [t_0, 0.001]$) show us that when t_0 is small (stiff system), ROCK4 computes with more time steps than to achieve a tolerance of 10^{-6}. And the time step gets much larger when system gets less stiffer.



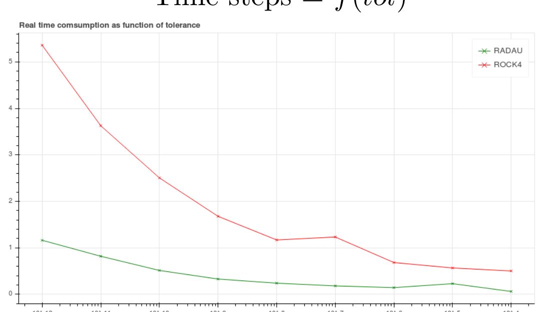
global error in norm 2 $= f(tol)$



Time steps $= f(tol)$



Function evaluations $= f(tol)$



Real time consumption $= f(tol)$

Figure 10 – Comparison between ROCK4 and RADAU5 when $t_0 = 0.0001$

Actually, when we set the same stiffness for both methods, implicit RADAU evaluates still less often than ROCK (bottom left of Fig 10). Because ROCK method use adaptive stages and time step length, with the same level of time steps number, ROCK could calculate more often evaluation function than RADAU could. Therefore, the main

advantage of ROCK4 lies on the fact that it is an explicit method which allows us use less memory. We should always compromise between the time and memory consumption. **4.2.4**

♠
♢ ♡
♣