MAP551 - PC3: Numerical Integration of Ordinary Differential Equation (Part I)

Te SUN

October 2018

1 Numerical integration in he frame work of application of Peano's theorem

1.1 Study of the ODE

We consider the dynamical system on the function $u(t) \in \mathbb{R}$:

$$\mathrm{d}_t u = f(t,u), \quad u(0) = 0.$$

with

$$f(t,u) = 4\left(\operatorname{sign}(u)\sqrt{|u|} + \max\left(0, t - \frac{|u|}{t}\right) \cos\left(\frac{\pi \log(t)}{\log 2}\right)\right), \quad t \neq 0$$

and

$$f(t,u) = 4\left(\operatorname{sign}(u)\sqrt{|u|}\right), \quad t = 0$$

If we focus at the neighborhood of $u = 0$, we find out that function f is no longer globally Lipschitz (which is due to the non-Lipschitz characteristic of function $\sqrt{\cdot}$).

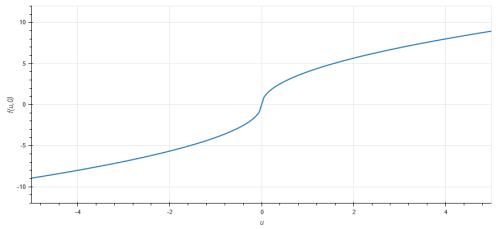


Figure 1 – $f(u,0)$

However, the function f is still continuous for $\forall(u,t) \in \mathbb{R}\times\mathbb{R}$: When $t \neq 0$, or $t \equiv 0$, we can easily prove that f is continuous thanks to the continuity of $\sqrt{\cdot}$ and $\cos(\cdot)$. When $t \to 0$, suppose there $u, v \geq 0$, we have :

$$\begin{aligned}|f(v,t) - f(u,0)| &= |4(\sqrt{v} - \sqrt{u}) + 4\max(0, t - \frac{v}{t})\cos(\frac{\pi\log(t)}{\log(2)})| \\ &\leq |4(\sqrt{v} - \sqrt{u})| + |4t\cos(\frac{\pi\log(t)}{\log(2)})| \xrightarrow{t\to 0, v\to u} 0\end{aligned}$$

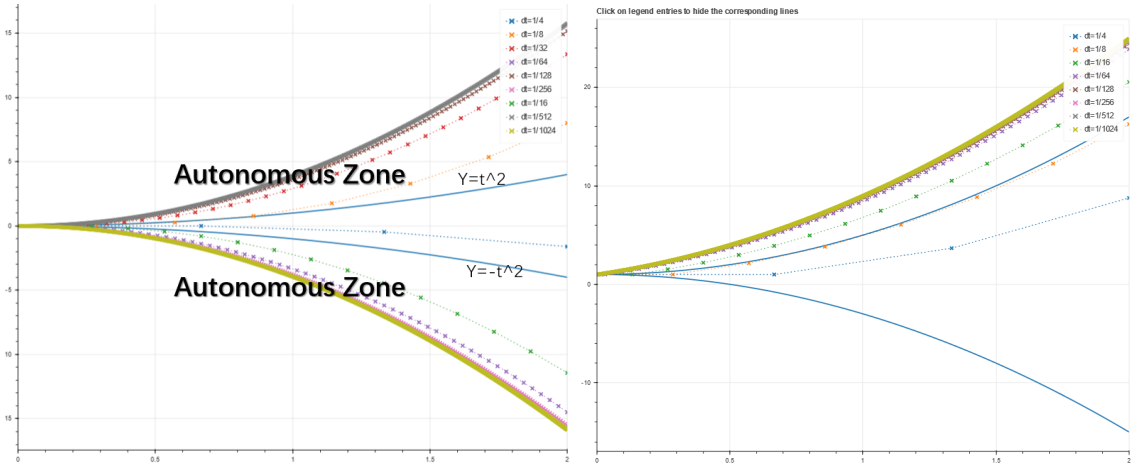
Therefore, according to the *Peano theorem*, there exist at least one solution for the dynamical system. (**2.1.1**)

We assume that $\Delta t = 2^{-i}$, a constant step for time discretization. We evaluate two values :

$$f(\Delta t, 0) = 4\Delta t\cos(\frac{\pi\log(2)\times(-i)}{\log 2}) = 4\times(-1)^{-i}\Delta t$$

And when $|y| \geq t^2$:(**2.1.2**)

$$f(t,y) = 4\,\mathrm{sign}(y)\sqrt{|y|} \qquad \forall t \in \mathbb{R}$$



$u(0) = 0$ $\qquad$ $u(0) = 1$

Figure 2 – Numerical solution with different initial conditions

Consider the case where $u(0) \neq 0$, on the neighborhood of $u(0)$, the function $f(0,\cdot)$ is Lipschitz, which represents the premise for Cauchy-Lipschitz theorem.

We focus on the half-plan where $t > 0$, we suppose firstly, $u(0) > 0$.we can see that $f(0,u) > 0$, there exists an interval $[0, t^+(u_0)[$ where $u(t) > 0$ and $u > t$. Thus $f(t,u) = 4\,\mathrm{sign}(u)\sqrt{|u|}$ and $u \geq u(0) > 0$. $f(t,u)$ posses Lipschitz property for all $t \in [0, t^+[$. If $t > t^+(u_0)$, f is still Lipschitz with a bigger Lipschitz Coefficient than that of $[0, t^+[$. **However, t^+ depends only on the u_0, then for all $t > 0$, f is a Lipschitz function.** (The discussion for u(0) <0 should be similar).(**2.1.3**)

1.2 Numerical integration

From Fig 3, when time step Δt is constructed with even i, then the curve goes downward. While i is odd, the curve goes upwards. That is due to the non-Lipschitz, thus non-uniqueness of the solution that passes $u(0) = 0$. **To reach the case of unique solution, a necessary condition is that $u(0) \neq 0$. The Peano theorem is still adequate for numerical simulations, since we can always find solutions numerically.** (**2.1.3-2.1.4**)

2 Conservative System and Euler integration methods

If we look at the earth-moon system in Galilean frame of reference :

$$\epsilon \, \mathrm{d}_t^2 Y = \frac{\epsilon(1-\mu)}{||A-Y||^2} \frac{A-Y}{||A-Y||} + \frac{\epsilon\mu}{||B-Y||^2} \frac{B-Y}{||B-Y||}.$$

Where μ, $1-\mu$ represent respectively the mass of moon and the earth. ϵ is the satellite mass that can be ignored compared with earth and the moon.

With substitution of variable $y = e^{-it}Y = y_1 + y_2$, another representation of equations is showed below :

$$\begin{aligned}
\mathrm{d}_t y_1 &= y_3, \\
\mathrm{d}_t y_2 &= y_4, \\
\mathrm{d}_t y_3 &= y_1 + 2\, y_4 - (1-\mu)(y_1+\mu)/r_1^3 - \mu(y_1 - 1 + \mu)/r_2^3, \\
\mathrm{d}_t y_4 &= y_2 - 2\, y_3 - (1-\mu) y_2/r_1^3 - \mu y_2/r_2^3,
\end{aligned}$$

Finally, given with initial values, we suppose :

$$y_1(0) = 0.994, \quad y_2(0) = 0, \quad y_3(0) = 0, \quad y_4(0) = -2.00158510637908252240537862224.$$

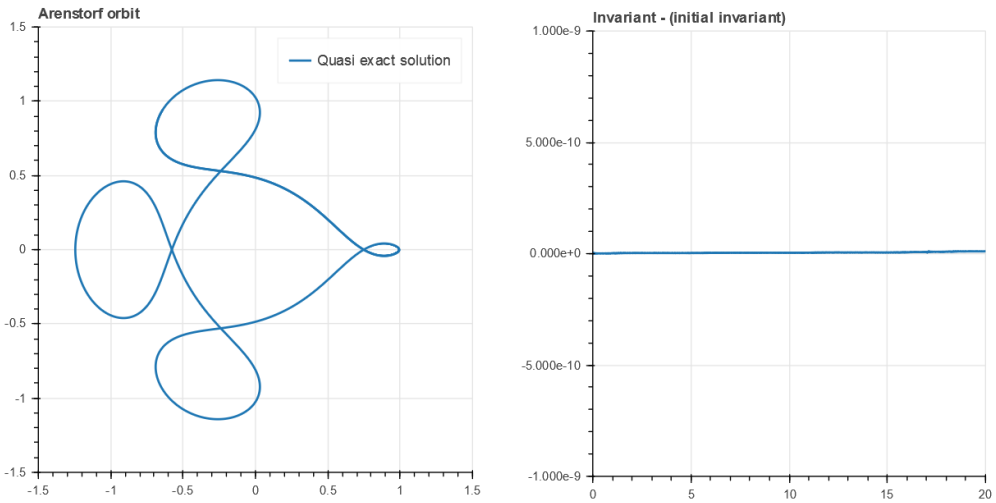
If we consider the quantity H :

$$H = \underbrace{\frac{1}{2}(y_3^2 + y_4^2)}_{\text{kinetic energy}} - \underbrace{\frac{1}{2}(y_1^2 + y_2^2)}_{\text{entrainment energy}} \underbrace{-\frac{1-\mu}{r_1} - \frac{\mu}{r_2}}_{\text{gravitation energy}}$$

We can see that :

$$\begin{aligned}
\mathrm{d}_t H &= \mathrm{d}_t y_1 y_1 + \mathrm{d}_t y_2 y_2 + y_3(y_1 + 2\, y_4 - (1-\mu)(y_1+\mu)/r_1^3 - \mu(y_1 - 1 + \mu)/r_2^3) + \cdots \\
&\quad \cdots + y_4(y_2 - 2\, y_3 - (1-\mu) y_2/r_1^3 - \mu y_2/r_2^3) + \frac{1-\mu}{r_1^3}\Big((y_1+\mu)y_3 + y_2 y_4\Big) + \cdots \\
&\quad \cdots + \frac{\mu}{r_2^3}\Big((y_1 - 1 + \mu)y_3 + y_2 y_4\Big) \\
&= (y_1 + 2y_4 - \mathrm{d}_t y_3)y_3 + (y_2 - 2y_3 - \mathrm{d}_t y_4)y_4 + \mathrm{d}_t y_3 y_3 + \mathrm{d}_t y_4 y_4 + y_3 y_1 + y_4 y_2 = 0
\end{aligned}$$

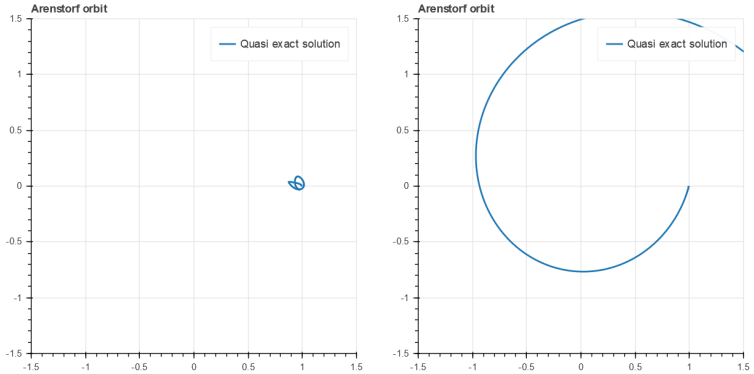
We conclude therefore H is a conservative invariant through time. (**3.1**)



phase plane (y_1, y_2) variation of invariant H

Figure 3 – Quasi exact solution with explicit RK45

The dynamics of the system seems like periodic in time. The orbit around earth (right) has a much smaller radius than that of the moon. That is because the bigger mass of earth is characterized with a bigger gravitation force. In the meanwhile, we could see that the initial velocity (energy) is still constrained within the earth-moon system (in other words, initial kinetic energy is smaller than the potential energies), else wise, the satellite will escape from two planets.



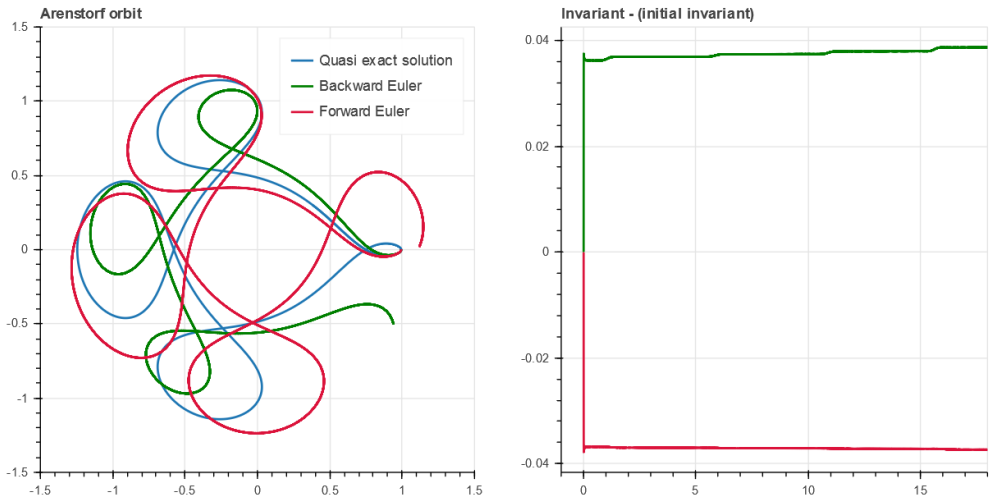
$y_4(0) = 1.935$ $y_4(0) = -3$

Figure 4 – Solution with different initial conditions

We could observe that the system is essentially sensible to initial datum. In the Fig 4, we have two different cases : 1 : the satellite can not escape from the earth. 2 : Initial energy is big enough for the satellite to escape from the two-planet system.

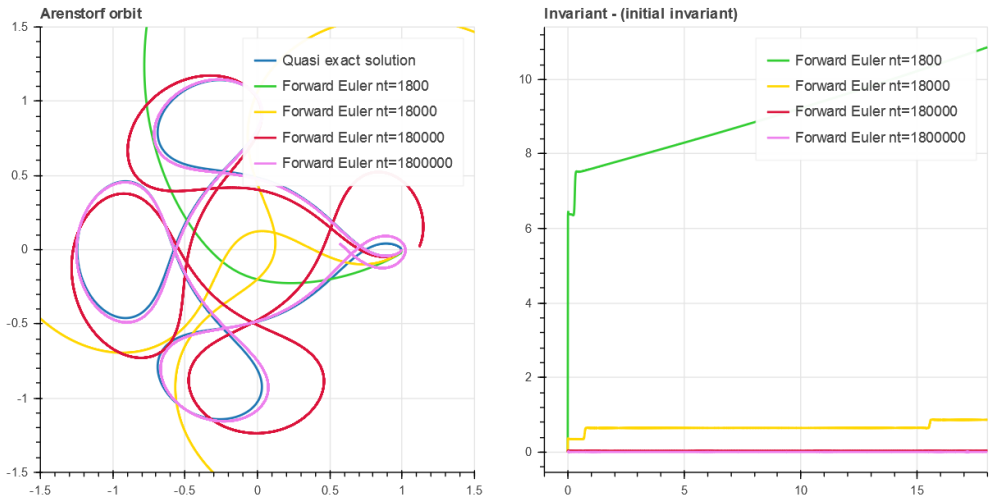
Due to the error term and the sensibility of the dynamical system, the trajectories are no longer periodic when we try to use Euler methods.

When the solver is stable and the system has unique solution, it comes as no surprise



Solutions with different methods Different "invariant"

Figure 5 – Solutions with different methods : Green : Backward Euler ; Red : Forward Euler



Solutions with different discretization Different "invariant"

Figure 6 – Solutions with different discretization using Forward Euler : Green : $n_t = 1800$; Gold : $n_t = 18000$; Red : $n_t = 180000$;Violet : $n_t = 1800000$ (**3.2**)

that we get better numerical solutions with finer discretizations.

We come back to the RK45 method, when we are trying to solve the system during long-time. **The deviation of the invariant can be also observed from the trajectory in the long term.** Here, we increased the precision (tolerance) to 10^{-14}, and we save numerically the solution only for a 4 periods problem.

Therefore, to solve a conservative system, the error is accumulating in time. If we want to observe the long-term behavior of the system, we need to set a very fine tolerance.

3 Stability, order and accuracy for non-stiff and stiff equations

We consider the following problem :

$$\begin{cases} \mathrm{d}_t u(t) = k\left(cos(t) - u(t)\right) & \text{avec } k > 1 \\ u(t_0) = u_0 \end{cases} \tag{1}$$

and in the following, we will assume $t_0 = 0$.

3.1 Stiffness ?

This is a first order ordinary differential equation. We could then solve it analytically. Homogeneous equation :

$$\begin{aligned} \mathrm{d}_t u(t) &= -ku(t) (\mathcal{H}) \\ u(t) &= Ae^{-kt} \end{aligned}$$

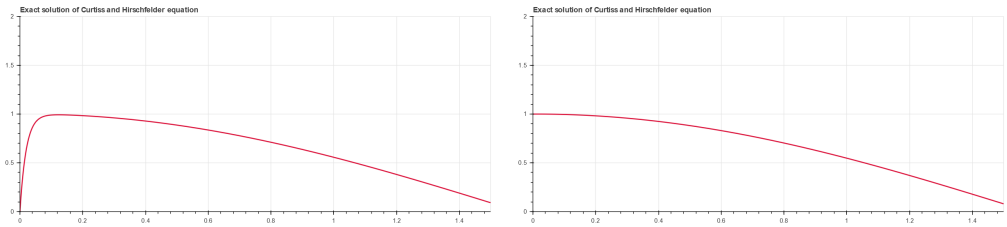
Particular equation :

$$\begin{aligned} \mathrm{d}_t A(t) e^{-kt} &= k\cos(t) (\mathcal{P}) \\ Re(\mathrm{d}_t A(t)) &= Re(ke^{(k+i)t}) \\ A(t) &= \frac{k}{k^2+1}\Big(kcos(t) + sin(t)\Big)\exp(kt) + B \end{aligned}$$

With the initial condition $u(0) = u_0$, we can solve for $c_0 = \Big(u_0 - \frac{k}{k^2+1}(k\cos(t_0) + \sin(t_0))\Big)e^{kt_0}$

$$u(t) = \frac{k}{k^2+1}\Big(kcos(t) + sin(t)\Big) + c_0 e^{-kt} \qquad \textbf{4.1.1} \tag{2}$$

We assume in this part that $t_0 = 0$, then $c_0 = u_0 - k^2/(k^2+1)$



$u(0) = 0,\ k = 105$ $\qquad$ $u(0) = k^2/(k^2+1),\ k = 105$

Figure 7 – Exact solution

When $k >> 1$, for the first order of k, we could have an equivalence for $u(t) = \cos(t) + c_0 e^{-kt} + o(1/k)$. Then we have two region as $t << \frac{1}{k}$ and another.

If $u_0 \neq k^2/(k^2+1)$ and given that $k >> 1$, $\mathrm{d}_t u(t)$ will be very big which leads to a violent variation of $u(t)$. However, as we expected, when $t >> 1/k$, the exact solution behaves as $\cos(t)$ that is relatively smooth since the principal term $k^2/(k^2+1)|\mathrm{d}_t \cos(t)| < 1 \quad \forall t \in \mathbb{R}$. As a result, the equation yield to a "stiffness" when k is large.(**4.1.2**)

If initial data taken as $u_0 = k^2/(k^2+1)$, even if k is large, the stiffness disappears because of the disappearance of first stiff region. From this exercise, **we saw that the 'stiffness' propriety lies on several reasons : (4.1.3)**

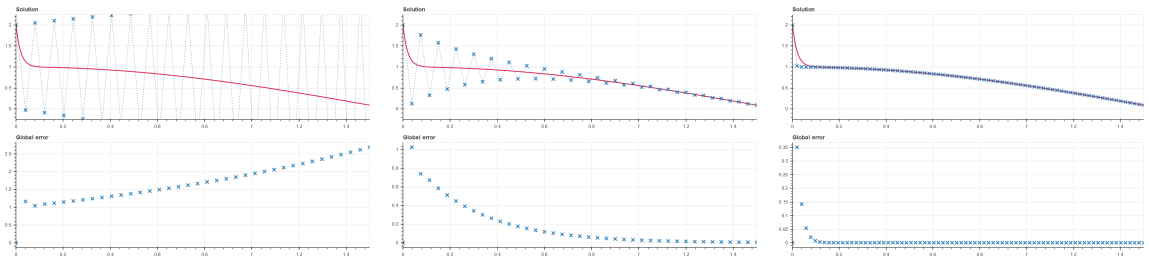
1. initial condition, initial datum ;
2. external parameters
3. $f(\cdot)$ function's propriety

3.2 Explicit Euler

The explicit Euler method to solve $\mathrm{d}_t u(t) = f(t,u)$ can be written :

$$\begin{cases} U^0 = u_0 \\ U^{n+1} = U^n + \Delta t\ f(t^n, U^n) \quad \text{where} \quad \Delta t = t^{n+1} - t^n \end{cases}$$

From Fig 9, we obtained three cases, where, firstly the numerical solution diverge completely ; and then the solution converge with an oscillation ; finally the it converge from the beginning. Respectively, the first regime corresponds with time step inferior to $\Delta t = 0.0375$ ($n_t = 40$). **And the limit for regime 2 and regime 3 is around** $\Delta t = 0.024$ **where** $n_t = 63$**.** (**4.2.1**)



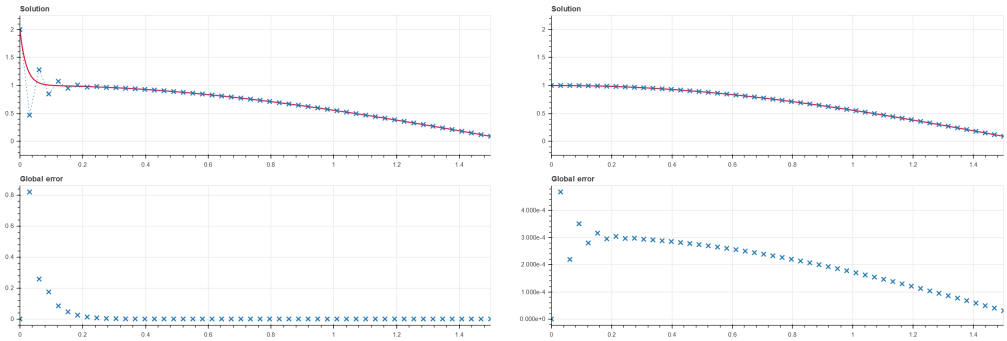
$n_t = 39$ $n_t = 41$ $n_t = 78$

Figure 8 – Numerical resolution with different initial datum, Forward Euler

$$U^{n+1} = (1 - k\Delta t)U^n + k\cos(t_0 + n\Delta t)\Delta t$$

As we know, the explicit Euler method is not stable and may introduce a strong instability to the solution, which is obviously observed when Δt is large. From the equation above, if $-1 < (1-k\Delta t) < 1$, we can guarantee the convergence of the homogeneous part. Instinctually, the critical $n_t = (t_{end} - t_{ini})/(2/k) = 37.5$. However, the term $k\cos(t_0 + n\Delta t)\Delta t$ prevent us from an obvious explicit expression of U^{n+1} out of U^n. And actually, the solution exploded already when $n_t = 38$.

This instability leads the resolution to nowhere but a total error. Without stability, we can no more talk about the convergence. (So as $n_t = 39$) (**4.2.2**)



$u(0) = 2,\ k = 50$ $\qquad$ $u(0) = k^2/(k^2+1),\ k = 50$

Figure 9 – Numerical resolution with different initial datum, Forward Euler

As for forward Euler method, supposed that the function is $\mathcal{C}^2$, we look at the quantity below :

$$\begin{aligned} d_{t^n}u(t^n) - k(\cos(t^n) - u(t^n)) &\approx \frac{u^{n+1} - u^n}{\Delta t} - k(\cos(t^n) - u^n) \\ &= \frac{u^n + d_{t^n}u^n \times \Delta t + \frac{d^2_{t^n}u^n}{2}\Delta t^2 - u^n}{\Delta t} - k(\cos(t^n) - u(t^n)) + \mathcal{O}(\Delta t^2) \end{aligned}$$

If u^t is the real analytic solution, then these terms become :$\frac{d^2_{t^n}u^n}{2}\Delta t$, which shows the fact that the schema has a local error at first order.

Now, we concentrate on the interval $[0, 1.5]$, we solve numerically the equation with forward method with respectively time step : 0.01,0.001, 0.0001 and 0.00001. We show the error in the diagram below with error defined as :

$$err_{||.||} = \log ||f_n - f||$$

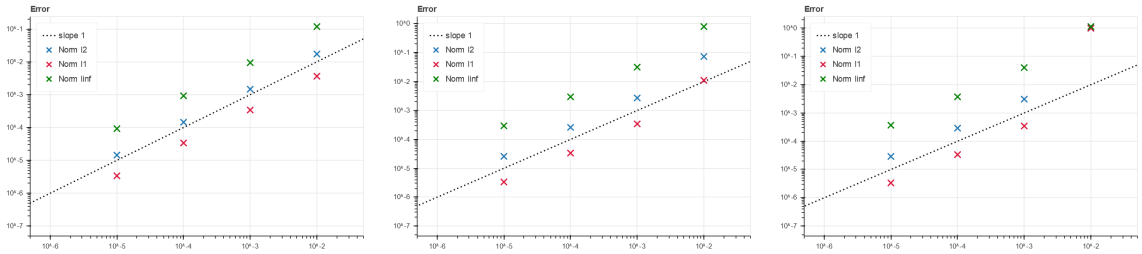
where f_n stands for the numerical resolution. We can use l^2, l^1, l^∞ as norms in the formula.

Graphically, we can see that the error converge at a first order velocity to the real solution. At we know, in finite dimension, all norms are equivalent. **So error are all at first order in terms of convergence velocity.** (**4.2.3**)

$$Err = \log(C^{te} \times \Delta t) = \log(C^{te}) + \log(\Delta t)$$

It comes as no surprise when k increase, the system becomes stiffer and thus, the error goes larger. **Although, the order of convergence doesn't change, but the constant increases.** Fig 10 (**4.2.4**)

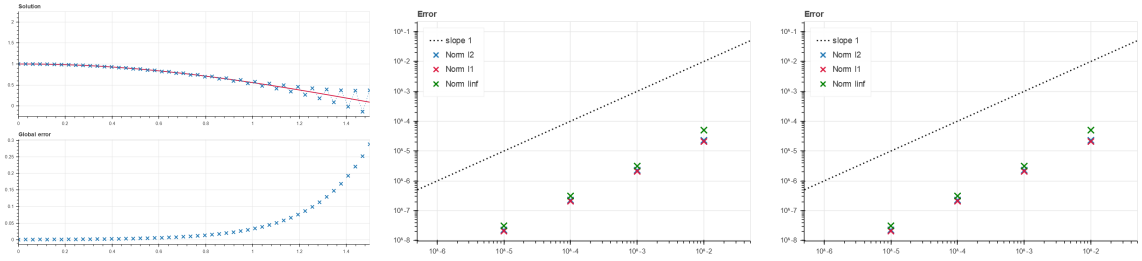
We now envision the configuration where $u_0 = \frac{k^2}{k^2+1}$. The numerical solution can still be unstable when k gets larger. As for errors, we can see that errors get much smaller in the stable zone when compared with the precedent results. The "stiffness" of the function seems



$k = 50$ $k = 160$ $k = 200$

Figure 10 – Log error

not to influence on the error terms. In the meanwhile, the initial datum changes with the value of k, **so the error depends immensely on the initial condition.** (Fig 11) (**4.2.5**)



$k = 70\, n_t = 50$ $k = 160$ $k = 200$

Figure 11 – Question 4.2.5 with initial condition $u_0 = \frac{k^2}{k^2+1}$

The explicit Euler method is usually a time saving scheme since there is no need to inverse matrix. However, the main problem of this method is the stability. Especially, when the system becomes stiff, the non-convergence of the numerical solution could damage the further analysis. (**4.2.6**)

3.3 Implicit Euler

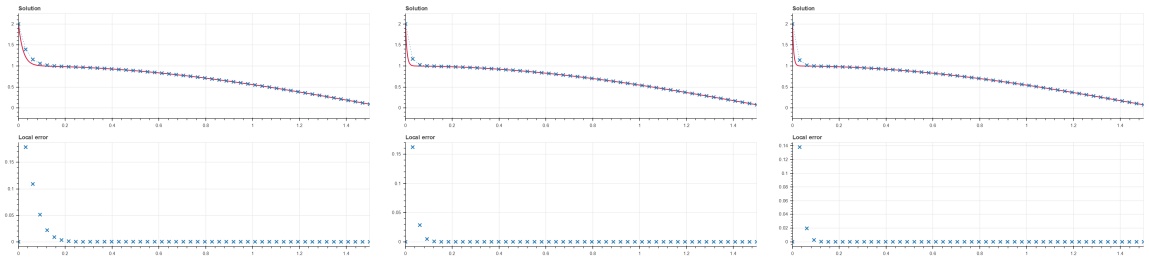
The backward Euler method to solve $\mathrm{d}_t u(t) = f(t, u)$ can be written :

$$\begin{cases} U^0 = u_0 \\ U^{n+1} = U^n + \Delta t\ f(t^{n+1}, U^{n+1}), \qquad \Delta t = t^{n+1} - t^n, \end{cases}$$

$$(1 + k\Delta t)U^{n+1} = U^n + k\cos(t_0 + (n+1)\Delta t)\Delta t$$

Actually cos function is bounded between -1 and 1. Given that $k\Delta t$ is a constant, so $k\cos(t_0 + (n+1)\Delta t)\Delta t$ is bounded as well. Then $k > 0$ and $\Delta t > 0$ is a sufficient condition for the convergence of the solution. In this case, the implicit Euler is stable.

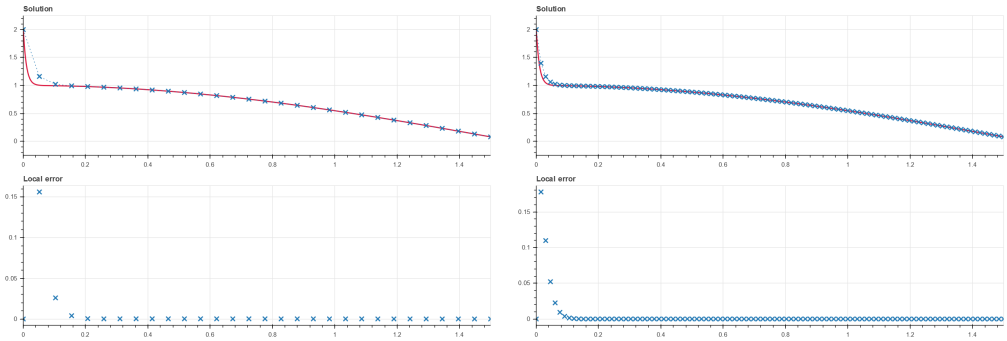
As we can see from Fig 12, whatever the stiffness is, the numerical solution converge always to the exact solution. **That is thanks to the stability of the backward Euler scheme.(4.3.1)**



$k = 50$ $k = 160$ $k = 200$

Figure 12 – Numerical resolution with different stiffness for a given $n_t = 50$, implicit Euler

Although the method is stable, which implies that the numerical solution always converges to the exact one, we could not be too optimistic in terms of the accuracy (cf. oscillation of explicit Euler). When the function gets stiff, the performance is no better than the explicit Euler method.



$k = 50,\ n_t = 30$ $k = 50,\ n_t = 100$

Figure 13 – Numerical resolution with discretization with given $k = 100$, implicit Euler

If we look at an extreme case where $k = 200$ and $n_t = 10$, numerically, the solution is no more acceptable. Yet, the solution is converging to the exact one.

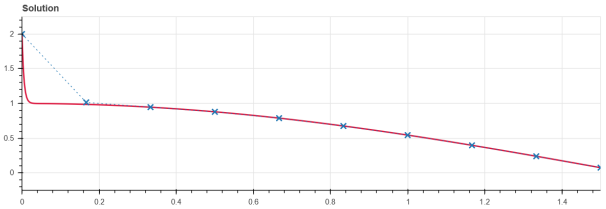
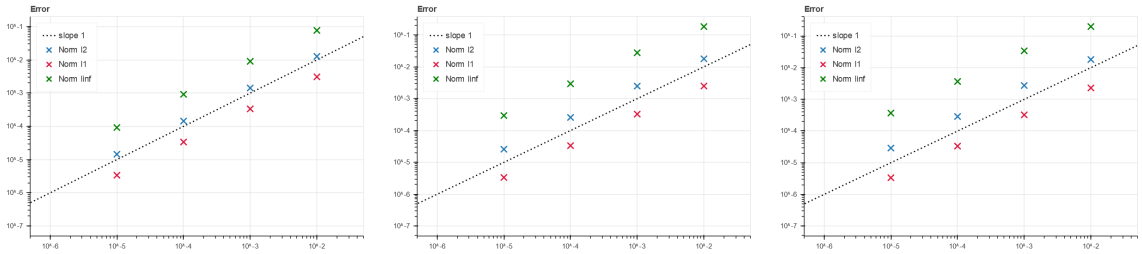


Figure 14 – An extreme case

Graphically (Fig 15), the stiffness will increases the constant error term. And the influence of stiffness towards error is smaller compared to the explicit case. If we do the same

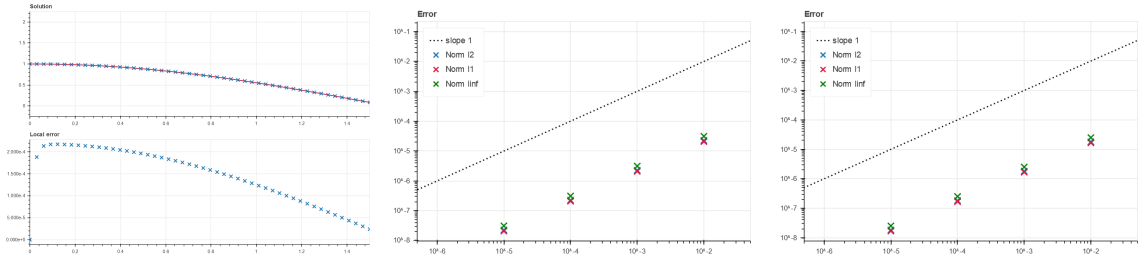
calculation in the question **4.2.3**, the implicit has a first order of time convergence propriety as well. And it is justified by the diagram.(**4.3.3**)



$k = 50$ $k = 160$ $k = 200$

Figure 15 – Log error

From Fig 16, we can tell that the implicit method is remarkably efficient and precise to solve the equation. Thanks to the stability of the scheme, the error doesn't deviate although the k gets larger. (**4.3.6**)



$k = 70\, n_t = 50$ $k = 160$ $k = 200$

Figure 16 – Question 4.3.5 with initial condition $u_0 = \frac{k^2}{k^2+1}$

3.4 Conclusion

From two sections above, we saw different proprieties of explicit and implicit Euler schema. In terms of time consumption, the explicit one prevails especially when the target function is defined in higher dimension (because there is no need to inverse matrixs). And we saw that, explicit Euler is conditionally convergent while implicit is always stable. The stability appears as an remarkable propriety, as we can just refine the time step to approach the exact solution. Imagine when we introduce another dimension x "space" as an variable. The interaction between x and t which was studied by **Courant, Friedrichs** and **Lewy** exerts unexpected impacts to the resolution of differential equation.

Coming back to the stiffness cases, the stiffness of function could lead to a bigger error term. Both methods suffer from the stiffness. Yet, it could be safer to use implicit method when function is stiff to avoid instability.