# RUNGE-KUTTA Schemes for Joint Identification and Integration of Stiff Systems

**Thomas Aussaguès** (student), **Said Ouala** (teacher)
IMT Atlantique; Lab-STICC, 29200 Brest, France
{thomas.aussagues,said.ouala}@imt-atlantique.net

***Abstract*—In this work, we focus on the joint identification and integration of dynamical systems using a residual neural network. We propose a framework that learns simultaneously the RUNGE-KUTTA coefficients of the integration scheme and the parameters of the dynamical operator. We test the proposed architecture on several examples including stiff systems.**

***Index Terms*—Dynamical systems, Residual neural network, Integration scheme, Stiff systems, Joint identification and integration**

## I. Introduction

The modeling of the temporal variability of the geophysical characteristics of the oceans such as the temperature or the sea level is a critical challenge. These phenomenon are governed by complex differential equations. These equations are useful for forecasting application in several areas such as meteorology or navigation.

Beyond classical imperfections present in instrumentation tools, observation of ocean related variables are sampled with a predefined sampling rate. Unfortunately, the characteristic time scale of the system is often inferior to the sampling time of the sensors [Dickes, 1991] making the observations miss fast evolving variability that is crucial when trying to understand the underlying phenomenons, as well as from a modeling and an operational perspectives.

When considering the modeling of these variables, the complexity of the governing equations is linked to several considerations regarding the underlying physics. Among other aspects, the non-linear nature of the models makes their understanding and exploitation non trivial and matching such models to real observations requires increasing their dimensionality which tends to make them even more non-linear [van10].

From a operational point of view, the non-linear nature of the models makes them impossible to solve analytically and most of the time, numerical integration techniques are necessary in order to simulate an equation given some auxiliary conditions.

In practice, solving a non-linear differential equation corresponds to transforming the continuous equation to a discrete one through a numerical integration scheme [PC89a]. When considering Ordinary Differential Equations (ODEs), a single (independent) variable (which is time in the case of dynamical systems) is discretized using a given integration time-step. It must be chosen according to the the characteristic time scale of the problem. However, computational limits can not allow us to correctly choose the integration time-step. Specifically, if the integration time-step is too small, round-off errors can have a higher impact then expected leading to poor integration of the system. Nonetheless, if it is too high, we will miss the fast time variations of the system because the time varying scale of the system is lower than the integration time-step. In extreme cases, the prediction could diverge.

When considering identification applications, the dynamical is unknown and must be learnt during the identification part preceding the integration one.

In this work, we focus on integration and identification of dynamical systems, and especially stiff ones. To judge the quality of the prediction, we concentrate on the asymptotic behaviour of the prediction.

## II. Problem formulation

Let us consider a dynamical system which is described by the following ODE (1):

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = \mathcal{F}(t, X) \tag{1}$$

$$X(t) = \int_{t_0}^{t_1} \mathcal{F}(t, X)\mathrm{d}t \tag{2}$$

where $\mathcal{F}$ is the dynamical operator, $t_0$ and $t_1$ the time limits of the integration. In order to solve the ODE (1), we are discretizing the previous formula (2) using a integration time-step $h$. Let us consider $N$ points and an integration time-step given by

$$h = \frac{t_1 - t_0}{N} \tag{3}$$

Then, we introduce the integration scheme modeled with the function $\Phi$ such that the approximated solution at each grid point is :

$$X_{t_{k+1}} = X_{t_k+h} = X_{t_k} + h\Phi(t_k, X_k, h) \tag{4}$$

with $\Phi(t_k, X_k, h)$ a given integration formula.

In this work we focus on explicit/implicit RUNGE-KUTTA schemes as they allow an efficient trade-off between the computational complexity of the numerical integration and the precision of the solution. Furthermore, these schemes can be easily implemented in differentiable frameworks [OPF19] which promotes their application for the integration and identification of non-linear differential equations.

### A. RUNGE-KUTTA explicit schemes

A $s$-stage RUNGE-KUTTA explicit integration scheme can be defined by the following update relation :

$$X_{t_{k+1}} = X_{t_k} + h\sum_{i=1}^{s} b_i k_i \tag{5}$$

where $k_i$ is given by :

$$\forall 1 \leq i \leq s, k_i = \mathcal{F}\left(t_k + c_i h, X_{t_k} + h\sum_{j=1}^{i-1} a_{ij} k_j\right) \tag{6}$$

The RUNGE-KUTTA coefficients are subject to the following constraints :

$$\begin{cases} \sum_{i=1}^{s} b_i = 1 \text{ (consistency condition)} \\ \forall i \in [1, s], c_i \in ]0, 1[ \\ \forall i \in [0, s], \sum_{j=1}^{i-1} a_{ij} = c_i \end{cases} \tag{7}$$

For example, the well-known RK4 method is given by :

$$X_{t_{k+1}} = X_{t_k} + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

where

$$\begin{aligned} k_1 &= \mathcal{F}(t_k, X_{t_k}) \\ k_2 &= \mathcal{F}(t_k + \frac{h}{2}, X_{t_k} + \frac{h}{2}k_1) \\ k_3 &= \mathcal{F}(t_k + \frac{h}{2}, X_{t_k} + \frac{h}{2}k_2) \\ k_4 &= \mathcal{F}(t_k + h, X_{t_k} + hk_3) \end{aligned}$$

### B. Stability analysis

The most used stability analysis of integration schemes consists on studying a numerical integration of the following ODE :

$$\begin{cases} \frac{\mathrm{d}X(t)}{\mathrm{d}t} = \lambda X(t), \lambda \in \mathbb{C}, \mathrm{Re}(\lambda) \leq 0 \\ X(t=0) = X_0 \end{cases} \tag{8}$$

Then, for the problem (8), we can rewrite the update formula (5) by using the stability function of the integration scheme $\mathcal{R}(\lambda h)$ :

$$X_{t_{k+1}} = \mathcal{R}(\lambda h) X_{t_k} \tag{9}$$

When considering a $s$-stages RUNGE-KUTTA integration scheme, the stability function is a polynomial of order less or equal to $s$.

For example, the EULER explicit scheme (figure 2) ($s = 1$) can be written as (equation 9) using :

$$X_{t_{k+1}} = X_{t_k} + \lambda h X_{t_k} \iff \mathcal{R}(\lambda h) = \frac{X_{t_{k+1}}}{X_{t_k}} = 1 + \lambda h$$

with $\mathcal{R}(\lambda h)$ a polynomial of degree one (which is equal (here) to the number of stages $s = 1$).

Therefore, we can define the stability regions of the integration. This region corresponds to set of values $\lambda h$ for which the integration scheme is stable. This condition can be written as :

$$|\mathcal{R}(\lambda h)| < 1 \tag{10}$$

This condition makes sure that the discrete system (9) have the same behavior as system (8) with a fixed point at the origin.

For the EULER explicit scheme illustrated by figure 4, we obtain the following stability region (figure 2):

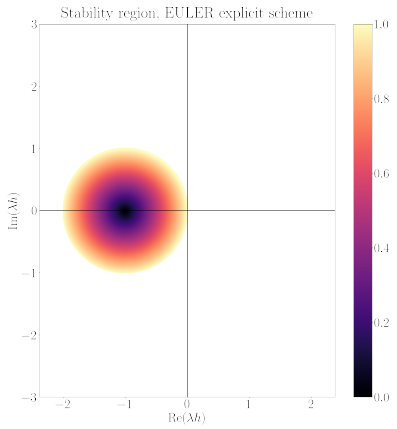


Fig. 2. Stability region of the EULER explicit scheme

When considering RUNGE-KUTTA integration schemes with an arbitrary number of $s$ stages, the formula (11) is used in order to derive the corresponding stability polynomial [HW96] :

$$\mathcal{R}(z) = 1 + zb^T (I - zA)^{-1} \mathbb{1} = \frac{\det\left(I - zA + z\mathbb{1}b^T\right)}{\det\left(I - zA\right)} \tag{11}$$

where $z = \lambda h$, $\mathbb{1} = [1, \ldots, 1]^T$, $A = (a_{ij})$ and $b = [b_1, \ldots b_s]^T$.

### C. Stiff ODEs

Some dynamical systems may have different time varying scales for each components. They are called stiff system. Then, we can quantify this characteristics with the stiffness $\xi$ defined by :

$$\xi(t) = \frac{\lambda_{max}}{\lambda_{min}}(t) \tag{12}$$

where $\lambda_{max}$ and $\lambda_{min}$ are respectively the modules of the largest and smallest eigenvalues of the dynamical operator $\mathcal{F}$. We can notice that for non-linear dynamical systems, the stiffness ratio depends on the time $t$. If the system is linear,

the dynamical operator is reduced to a matrix such as $\mathcal{F} = A$ and, therefore, the stiffness ratio is constant.

For example, the following linear system can be considered as a stiff one [PC89b] :

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = \begin{bmatrix} \lambda_f & 0 \\ 0 & \lambda_s \end{bmatrix} X(t) \tag{13}$$

Specifically, if we take $\lambda_s = -10^{-1}$ and $\lambda_f = -10^3$, we obtain a stiffness ratio (equation 12) of $\xi = 10^4$. Stiff systems are very tricky to integrate because of the different time varying scales involved. If we applied the EULER explicit integration scheme to the system (13), we must choose $h$ in the intersection of the two following subsets : $\left\{|1 + 10^{-1}h| < 1\right\}$ [1] and $\left\{|1 + 10^{4}h| < 1\right\}$ [2]. So, we can notice that for stiff systems, the integration step requires strong stability constraints and so a large stability region. Explicit RUNGE-KUTTA schemes are not suited for this type of system because of their small stability region. To solve this issue, we introduce implicit RUNGE-KUTTA integration schemes, which allows us using extremely efficient time-steps that that are suitable for the integration of stiff systems.

### *D. RUNGE-KUTTA implicit schemes*

For implicit schemes, the coefficient $k_i$, $1 \leq i \leq s$ involves all the coefficients $k_j$ with $1 \leq j \leq s$. Therefore, a RUNGE-KUTTA implicit scheme is defined with the same formula as before (equation (5)) with the following modification for computing the $k_i$ coefficients :

$$\forall 1 \leq i \leq s, k_i = \mathcal{F}\left(t_k + c_i h, X_k + h\sum_{j=1}^{s} a_{ij}k_j\right) \tag{14}$$

The previous equations ((5) and (14)) can be rewritten as a system of $s \times n$ algebric equations [CB83] using the concatenation operator $\bigoplus$ and the product of KRONECKER $\bigotimes$ :

$$\begin{aligned} Y_i &= x_k + h\sum_{i=1}^{s} a_{ij}\mathcal{F}(Y_j) \\ X_{t_{k+1}} &= X_{t_k} + h\sum_{i=1}^{s} b_i\mathcal{F}\left(t_k + c_i h, t_k, Y_i\right) \end{aligned} \tag{15}$$

We can then introduce the quantities $Y$, $X$ and $\mathcal{F}(Y)$ defined by :

$$Y = \bigoplus_{i=1}^{s} Y_i,\ X = \bigoplus_{k=1}^{s} X_k,\ \mathcal{F}(Y) = \bigoplus_{i=1}^{s} \mathcal{F}(Y_i) \tag{16}$$

Furthermore, using the product of KRONECKER, we obtain the following update relation :

$$Y = X + h\left(A \bigotimes I_s\right)\mathcal{F}(Y) \tag{17}$$

In order to solve the non-linear system (17), we use NEWTON's generalized algorithm given by :

$$Y^{(m)} = Y^{(m-1)} + \Delta^{(m)} \tag{18}$$

[1] Stability region of the fast component
[2] Stability region of the slow component

where the update term $\Delta$ is given by :

$$\begin{aligned} \Delta^{(m)} = &\left(I - hA\bigotimes J\right)^{-1} \\ &\left(X - Y^{(m-1)} + h\left(A\bigotimes I_s\right)\mathcal{F}(Y^{(m-1)})\right) \end{aligned} \tag{19}$$

with $J$ the jacobian matrix of the dynamical operator $\mathcal{F}$.

The resolution of the previous system increases the computational complexity of the scheme. Nevertheless, this rise of complexity is offset by the expansion of the stability region. Therefore, implicit RUNGE-KUTTA schemes are a trade-off between complexity and stability.

For example, we can consider the EULER implicit integration scheme given by :

$$X_{t_k} = X_{t_{k+1}} + \lambda h X_{t_{k+1}} \Longleftrightarrow \mathcal{R}(\lambda h) = \frac{X_{t_{k+1}}}{X_{t_k}} = \frac{1}{1+\lambda h}$$

As done with the EULER explicit scheme (figure 1), we illustrate EULER implicit integration scheme in figure 3. The stability region of the EULER implicit scheme is also

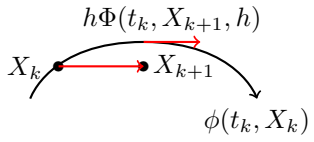


Fig. 3. Geometrical representation of EULER implicit algorithm

illustrated in figure 4. This stability region is far more wider than the one obtained for the explicit scheme (figure 2).

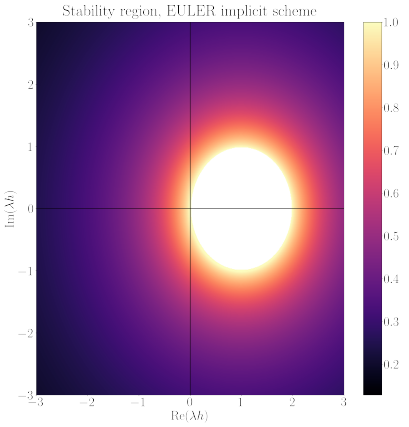


Fig. 4. Stability region of the EULER implicit scheme

## III. Proposed solution

In this paper, we are using the Residual Integration Neural Network proposed in [OPF19]. We aim to maximize the accuracy of the integration of the learned dynamical operator according to some observations. We can formulate this with the following optimization problem :

$$\min \sum_{k=1}^{n} \|X_{t_k}^T - \psi_{NN}\left(X_{t_{k-1}}^T, \theta_{NN}, a, b, c\right)\| \tag{20}$$

where $X_{t_k}^T$ is referring to the true states of the system, $\psi_{NN}$ is the output of a residual neural network, $\theta_{NN}$ the parameters of the learned dynamical operator (when considering identification applications) and $a, b, c$ the RUNGE-KUTTA coefficients that may make a $s$-stages RUNGE-KUTTA scheme trainable. All these parameters will be optimized by using the root mean squared error loose function.

### A. Extension to implicit schemes

The initial formulation of the RINN as proposed in [OPF19] was built on explicit formation. In this work, we upgraded this formulation to account for the learning of implicit differentiable schemes by including non linear solvers (II-D and equations (17), (18) and (19)).

### B. Batchs exclusion mechanism

During the training process, we noticed that for some batchs the mean squared error (MSE) was very low comparing to the other batchs, about $10^{-3}$, after 10 epochs. Since the MSE is almost null, there is interest to keep optimizing these batchs.

Therefore, we exclude batchs using the following algorithm in the training loop.

**Algorithm 1:** Batchs exclusion algorithm

```
for t in epochs do
    for b in batchs do
        training loop
        if loss< ϵ then
            batchs.del(b)
        end
    end
end
```

The following figure explains the exclusion policy that we are using. If the batch MSE is superior to a loss criterion $\epsilon$,

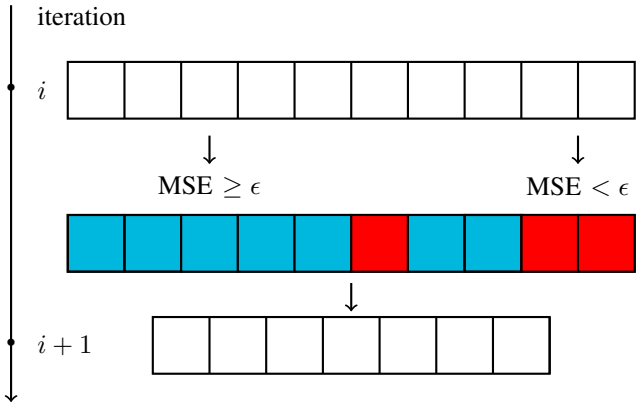


Fig. 5. Batchs exclusion policy

we continue optimizing this batch (blue case). Otherwise, the MSE is inferior to $\epsilon$ and we stop optimizing the batch.

### C. Adaptive learning rate

We also implemented an adaptive learning rate to avoid the plateau of MSE obtain during the training of the system (24). This adaptive learning $\rho$ is periodic and characterised by a period $T$ according to the formula :

$$\rho_{k+1} = \rho_k \left(1 + \cos\frac{k}{T}\right) \tag{21}$$

## IV. Results

During the experiments, we applied both explicit and implicit RUNGE-KUTTA integration schemes on two experimental setups :

- the identification of the RUNGE-KUTTA scheme to integrate a known dynamical model from sparse date,
- the joint identification, from spare data, of the RUNGE-KUTTA scheme and the ODE representation of the dynamical system.

### A. Explicit RUNGE-KUTTA integration schemes

Beyond the identification of RUNGE-KUTTA schemes for non-linear systems, we focus in this subsection on the joint identification of the integration scheme and the ODE.

*1) LORENZ-63:* We consider the LORENZ-63 system which a well-known non-linear chaotic system [Lor63] governed by the following equations :

$$\begin{cases} \dfrac{\mathrm{d}x(t)}{\mathrm{d}t} = \sigma\left(y(t) - x(t)\right) \\ \dfrac{\mathrm{d}y(t)}{\mathrm{d}t} = x(t)\left(\rho - z(t)\right) - y(t) \\ \dfrac{\mathrm{d}z(t)}{\mathrm{d}t} = x(t)y(t) - \beta z(t) \end{cases} \tag{22}$$

where $\sigma = 10$, $\beta = 8/3$ and $\rho = 28$. We simulate the true states of system 22 using [ALR04]. Then, we sub-sampled the true states using a sampling rate of 0.2 s to simulate sparse data. The figure (6) shows the generated attractors with their characteristic butterfly shape. The left trajectory corresponds to the true states and the right one to sub-sampled ones.

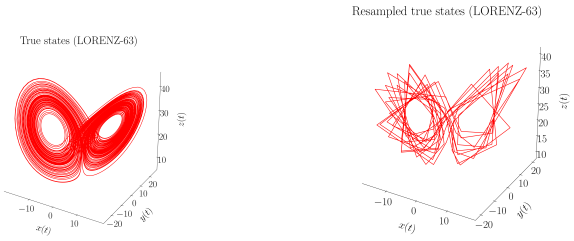


Fig. 6. True states and resampled true states (LORENZ-63)

To jointly learn the integration scheme and the dynamical model, we use a linear quadratic model (of dimension 3) and a 11 stages RUNGE-KUTTA explicit scheme. Regarding the training process, we consider the following parameters :

- number of epochs : 2000,

- learning rate $\rho = 0.001$,
- integration step-time : $h = 0.2$ s.

**•Analysis of the model prediction capacity**

The figures 7 and 17 presents the results obtained after 2000 epochs (the final loss was about $10^{-3}$) .

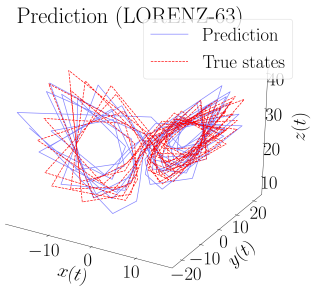


Fig. 7. Prediction of LORENZ-63

One can notice that, even if there are some imperfections, the prediction (in blue) is pretty accurate compared to the resampled true states (in red). The figure 7 shows that the predicted trajectory has a similar characteristic butterfly shape as the true one. The learnt coefficients can be found on figure 22.

**•Analysis of the learnt integration scheme stability**

The learnt scheme can be dissociated from the learnt ODE representation of the dynamical system. We can analyze the stability of the learnt scheme for different step-integration time values. Using equation (11), we can compute the stability region of the integration scheme. The computed stability

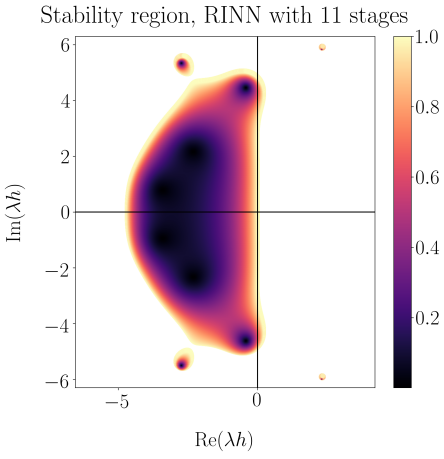


Fig. 8. Stability region of RINN 11

region is wide and it includes a large range of stable integration step-times that we did not use. Then, we can try to apply the learnt integration scheme to another dynamical system with one of these integration step-times.

Let us consider the LORENZ-96 dynamical system [Mat09] modeled with the following governing equations :

$$\forall i \in [0, J-1]\,,$$
$$\frac{\mathrm{d}X_i(t)}{\mathrm{d}t} = X_{i-1}(t)\left(X_{i+1}(t) - X_{i-2}(t)\right) - X_i(t) + F \quad (23)$$

with $F$ a fixed positive constant and $[0, J-1]$ a circular array. It means that $X_{-1}$ refers to $X_{J-1}$ and $X_J$ to $X_0$ and so on. Using the previous learnt RUNGE-KUTTA explicit scheme, we obtained the following results (figure 9) :

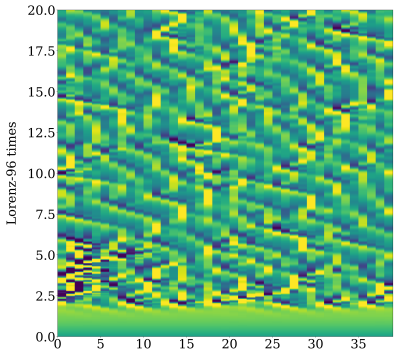


Fig. 9. Prediction of LORENZ-96 ($J = 40$ and $F = 8$)

Compare to the EULER explicit scheme stability region (figure 2), the obtained stability (figure 8 and figure 18) region is wider which relaxes the stability constraints. Nevertheless, the subset of the complex plan $\{z \in \mathbb{C}, \mathrm{Re}(z) < 0\}$ is not included in the stability region which implies that the integration scheme is not $A$-stable (and so $L$-stable). Moreover, the stability region is munch smaller than the one obtained with the implicit EULER scheme. Therefore, the constraints over $h$ (with a fixed value of $\lambda$) remain strong. We can not choose a high value of $h$ to speed up the integration process, the computed states will diverge. Therefore, the RINN 11 may not be suited for the integration of stiff systems.

### *B. Implicit RUNGE-KUTTA integration schemes*

To solve the previous issue, we propose an implicit RUNGE-KUTTA scheme II-D which is a trade-off between complexity and stability. Indeed, these schemes involve more complex equations which both increase the complexity and the stability region [CB83]. Such schemes have wider stability region than their explicit counterparts which makes them valuable for the integration of stiff systems.

Let us consider the $2 \times 2$ stiff linear system described by the following equations :

$$\begin{cases} \dfrac{\mathrm{d}x(t)}{\mathrm{d}t} = y(t) \\ \dfrac{\mathrm{d}y(t)}{\mathrm{d}t} = -1000x(t) - 1001y(t) \end{cases} \quad (24)$$

One can notice the two main particularities of system (24) :

- this is a stiff system, the stiffness ratio (equation 12) of system 24 is $\xi = 1000/1 = 10^3 \gg 1$,
- it possess a unique stable equilibrium point at the origin (because all the eigenvalues of 24 are negative).

In this subsection, we will consider two experimental setups described at the beginning of IV. Firstly, we focus on the identification of an implicit RUNGE-KUTTA integration scheme for stiff systems.

As done before, we generate true states of system (24) with the LSODA ODE solver [CB83] with a integration step-time of $0.1$ s. We will learn an implicit RUNGE-KUTTA scheme on with $s = 4$ stages with the following parameters for the training process :

- number of epochs : about 1500 (we stop earlier if we reach a certain value of MSE),
- learning rate $\rho = 0.001$,
- integration step-time : $h = 0.1$ s.

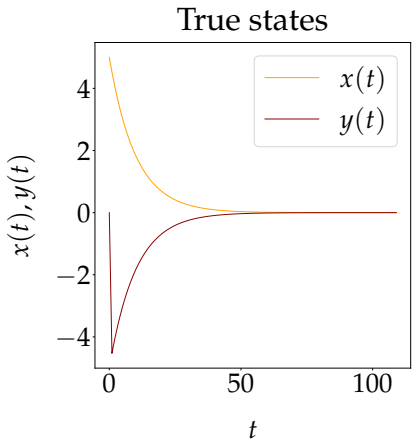


Fig. 10. True states of equation (24)

**•Analysis of the model prediction capacity**

After about 1500 epochs, we obtained the following prediction.

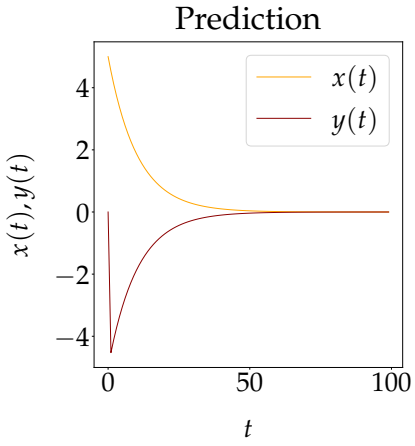


Fig. 11. Prediction of equation (24) (identification of the integration scheme only)

Firstly, the transitional regime (from 0 to 40 s) is well predicted as shown by figure 11 compared to figure 10. This result was expected since the RINN used in this experiment is implicit and can theoretically integrate stiff dynamics at a reasonable integration time steps. Using an explicit RINN formulation in such experiment with stiff dynamics will require the use of an excessively small integration time step. The figure 19 illustrates the MSE during the training. The learnt coefficients can be found on figure 23.

**•Analysis of the learnt integration scheme stability**

Finally, we can compute the stability region of the scheme. The stability region includes all the complex plane except two small disks. Nevertheless, the implicit RINN with $s = 4$ stages is not $A$-stable (which implies that the scheme is not $L$-stable) : the subset of the complex plan $\{z \in \mathbb{C}, \mathrm{Re}(z) < 0\}$ is not included in the stability region.Theses aspects are illustrated in figures 16 and 20). However, The integration scheme have a large stability region for $\lambda h < 0$ which allows us to use reasonable values of $h$ suitable for the integration of system (24). This improves the computational costs of the integration both in terms of temporal and spatial complexity. Moreover, the figure 21 shows the phase of the stability function.

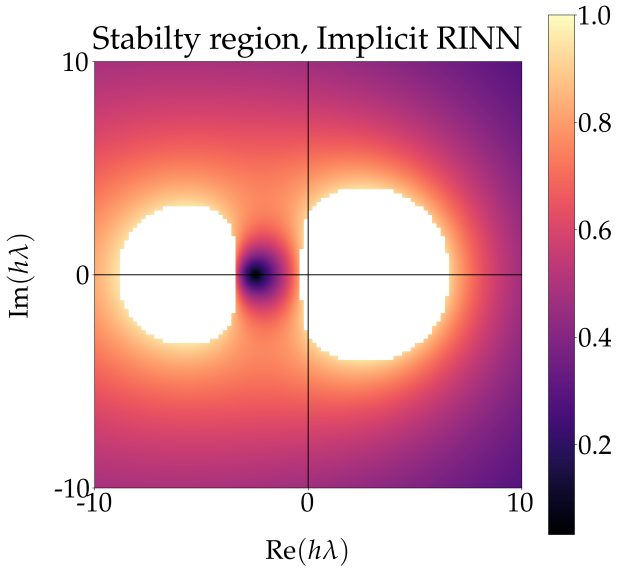


Fig. 12. Stability region of the learnt RUNGE-KUTTA scheme (identification of the integration scheme) for system (24)

### *C. Joint identification and integration*

In this part, we seek to jointly derive a data driven ODE and its corresponding implicit RUNGE-KUTTA scheme given a sequence of training data. Specifically, in addition to the trainable RUNGE-KUTTA coefficients, the RINN must learn the parameters of the dynamical operator (represented by $\hat{\mathcal{F}}$) $\theta_{NN}$ (equation (20)). In this respect we utilize the improved implicit RINN with :

- adaptative learning rate (equation (21)),
- the batch exclusion mechanism (III-B).

We focus on the previous stiff linear system (equation (24)). For this dynamical model, the learnable parameters of the model, $\theta_{NN}$ correspond to a matrix $2 \times 2$ matrix of trainable coefficients.

$$\hat{\mathcal{F}}(\theta_{NN}) = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \tag{25}$$

with $\theta_{NN} = (m_{11}, m_{12}, m_{21}, m_{22})$.

- initial learning rate value $\rho = 0.01$,
- number of stages : $s = 4$,
- integration step-time : $0.1$ s.

During the joint identification, we encountered a plateau on the MSE. The MSE was stuck at about $0.2$ until the end of the training process. The adaptive learning rate as explained in equation (21) helps getting out of the plateau. On the the figure 13, one can notice the plateau from 1500 epochs until the end of the training. These oscillations are related to the poorly suited learning rate value.

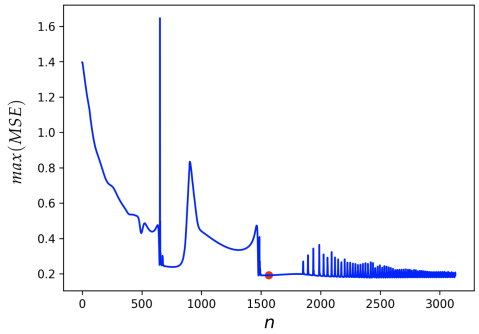


Fig. 13. MSE loss during the training without optimizations (joint scheme and model identification)

Therefore, we implemented the adaptive learning rate which suppressed the previous plateau and allows us to reach a (for 8000 epochs) loss value of $10^{-2}$ (figure 14).

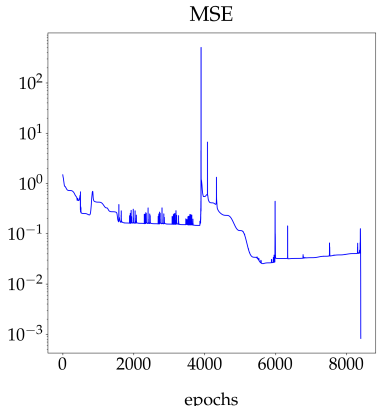


Fig. 14. MSE loss during the training (joint scheme and model identification)

The learnt coefficients can be found on figure 24.

**•Analysis of the model prediction capacity**

The predicted states using both identified integration scheme and model are satisfying. Although, there are some small differences in the transitional regime, we obtained a pretty accurate prediction as shown by the figure 15 where the true states are dashed and predicted ones are in are in continuous lines.

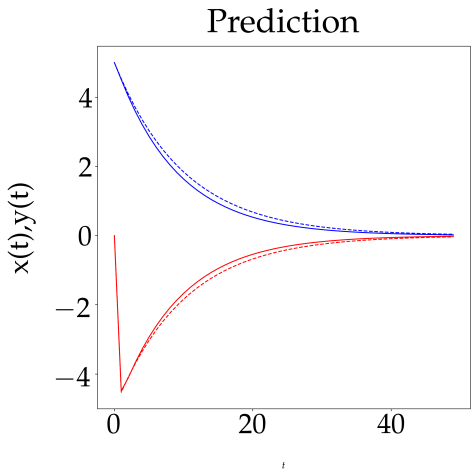


Fig. 15. Prediction of equation (24) (joint scheme and model identification)

**•Analysis of the learnt integration scheme stability**

Using equation (11), we can compute the stability region of the identified scheme. We obtained the following region (figure 16).

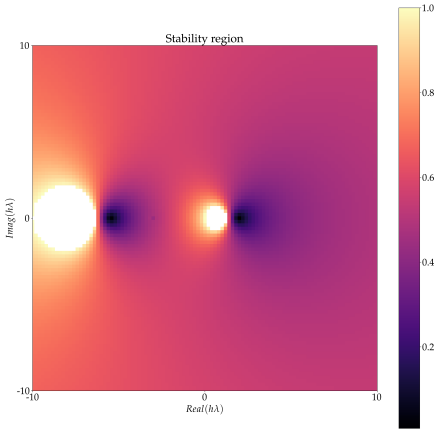


Fig. 16. Stability region of the learnt RUNGE-KUTTA integration scheme (joint scheme and model identification) for system 24

One can notice that the obtained region includes all the rectangle $[-10, 10] \times i[-10, 10]$ excepted two small disks. Therefore, there is a large panel of integration step-time which are reasonable for the integration of the studied stiff linear

system (24). Moreover, the implicit RINN is not $A$-stable (which implies that the scheme is not $L$-stable) : the subset of the complex plan $\{z \in \mathbb{C}, \mathrm{Re}(z) < 0\}$ is not included in the stability region. Nonetheless, the optimizations are a trade-off between precision and time efficiency. The obtained stability region may not be as accurate as the one obtained without any optimizations.

## V. Conclusion

In this work, we address the joint identification of a dynamical model and the associated RUNGE-KUTTA scheme. We focus on the identification of dynamical systems including non-linear and stiff ones using both explicit and implicit RUNGE-KUTTA integration schemes. The results demonstrate the relevance of the proposed network with satisfying results on both prediction and stability of the RINN. We may point out that our results with implicit RUNGE-KUTTA scheme were obtained on a simple linear stiff system. Further work will extend the RINN to more complex systems such as the Oregonator [FN74] which is a tricky stiff non-linear system.

## References


[Lor63] Edward N Lorenz. "Deterministic nonperiodic flow". In: *Journal of atmospheric sciences* 20.2 (1963), pp. 130–141.

[FN74] Richard J. Field and Richard M. Noyes. "Oscillations in chemical systems. IV. Limit cycle behavior in a model of a real chemical reaction". In: *The Journal of Chemical Physics* 60.5 (1974), pp. 1877–1884. DOI: 10.1063/1.1681288. eprint: https://doi.org/10.1063/1.1681288. URL: https://doi.org/10.1063/1.1681288.

[CB83] G. J. COOPER and J. C. BUTCHER. "An Iteration Scheme for Implicit Runge—Kutta Methods". In: *IMA Journal of Numerical Analysis* 3.2 (Apr. 1983), pp. 127–140. ISSN: 0272-4979. DOI: 10.1093/imanum/3.2.127. eprint: https://academic.oup.com/imajna/article-pdf/3/2/127/1814216/3-2-127.pdf. URL: https://doi.org/10.1093/imanum/3.2.127.

[PC89a] Thomas S. Parker and Leon O. Chua. "Integration of Trajectories". In: *Practical Numerical Algorithms for Chaotic Systems*. New York, NY: Springer New York, 1989, pp. 83–114. ISBN: 978-1-4612-3486-9. DOI: 10.1007/978-1-4612-3486-9_4. URL: https://doi.org/10.1007/978-1-4612-3486-9_4.

[PC89b] Thomas S. Parker and Leon O. Chua. "Locating Limit Sets". In: *Practical Numerical Algorithms for Chaotic Systems*. New York, NY: Springer New York, 1989, pp. 115–138. ISBN: 978-1-4612-3486-9. DOI: 10.1007/978-1-4612-3486-9_5. URL: https://doi.org/10.1007/978-1-4612-3486-9_5.

[HW96] Ernst Hairer and Gerhard Wanner. "Construction of Implicit Runge-Kutta Methods". In: *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 71–90. ISBN: 978-3-642-05221-7. DOI: 10.1007/978-3-642-05221-7_5. URL: https://doi.org/10.1007/978-3-642-05221-7_5.

[ALR04] B. A. Allan, S. Lefantzi, and J. Ray. "ODEPACK++: refactoring the LSODE Fortran library for use in the CCA high performance component software architecture". In: *Ninth International Workshop on High-Level Parallel Programming Models and Supportive Environments, 2004. Proceedings.* Apr. 2004, pp. 109–119. DOI: 10.1109/HIPS.2004.1299196.

[Mat09] Adrian Matthews. "PREDICTABILITY OF WEATHER AND CLIMATE, edited by Tim Palmer and Renate Hagedorn. 2006. Cambridge University Press: Cambridge, UK. ISBN 9780521848824. 718 pp." In: *Meteorological Applications* 16.2 (2009), pp. 253–253. DOI: https://doi.org/10.1002/met.104. eprint:

https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/met.104. URL: https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/met.104.

[van10] van Leeuwen P. J. "Nonlinear data assimilation in geosciences: an extremely efficient particle filter". In: *Quarterly Journal of the Royal Meteorological Society* 136.653 (Dec. 2010), pp. 1991–1999. ISSN: 0035-9009. DOI: 10.1002/qj.699. URL: https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.699.

[OPF19] S. Ouala, A. Pascual, and R. Fablet. "Residual Integration Neural Network". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2019, pp. 3622–3626. DOI: 10.1109/ICASSP.2019.8683447.

## APPENDIX

### *A. Other results on the explicit learnt RUNGE-KUTTA schemes*

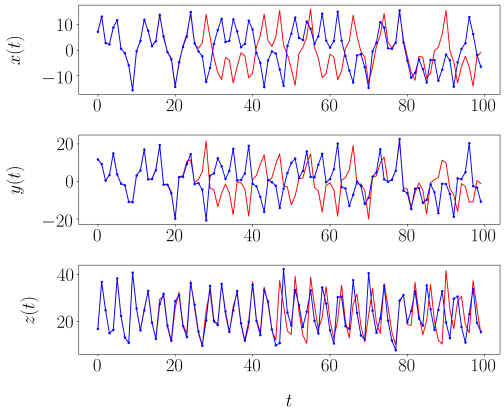


Fig. 17. Components of the predicted LORENZ-63 attractor in joint integration scheme and model identification

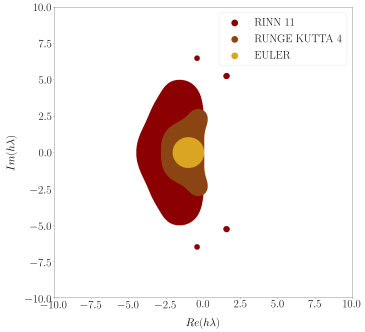


Fig. 18. Stability regions of RINN 11, EULER and RK4 explicit integration schemes

### *B. Other results on the implicit learnt RUNGE-KUTTA schemes and the improved RINN*

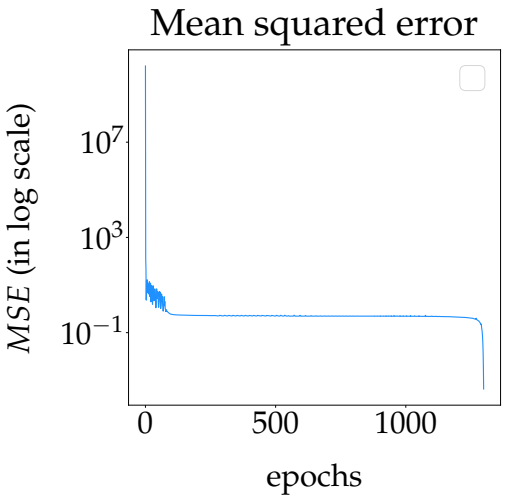


Fig. 19. MSE loss during the the identification of the RUNGE-KUTTA scheme for 24

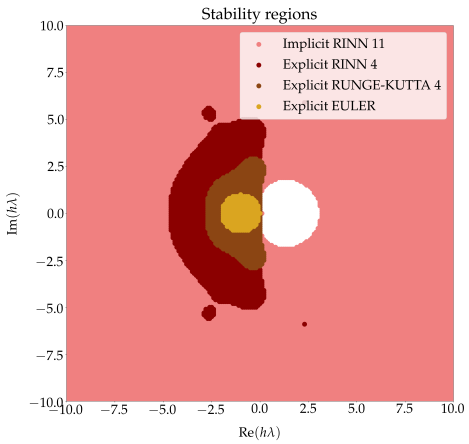


Fig. 20. Comparison of the obtained stability regions

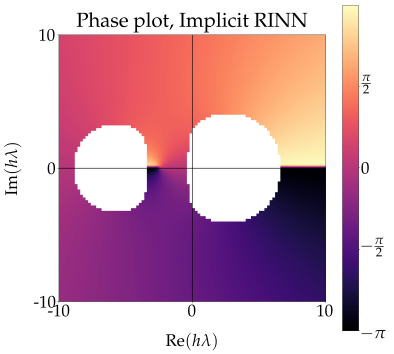


Fig. 21. Phase plot of the stability function

### C. Learnt RUNGE-KUTTA coefficients

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 0.00651 | 0.00651 | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | |
| 0.006520 | 1.83 | $-1.82$ | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 0.00689 | $-0.342$ | 0.641 | $-0.292$ | 0. | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 0.189 | $-0.929$ | 1.59 | $-0.364$ | $-0.109$ | 0. | 0. | 0. | 0. | 0. | 0. | 0. |
| 0.308 | 1.25 | $-1.13$ | $-0.0164$ | $-0.0515$ | 0.253 | 0. | 0. | 0. | 0. | 0. | 0. |
| 0.315 | 0.379 | $-0.323$ | $-0.0430$ | 0.0306 | 0.108 | 0.163 | 0. | 0. | 0. | 0. | 0. |
| 0.519 | 0.392 | $-0.192$ | 0.0307 | $-0.186$ | 0.104 | 0.0304 | 0.340 | 0. | 0. | 0. | 0. |
| 0.707 | 1.17 | $-0.353$ | 0.0926 | $-0.610$ | $-0.0148$ | $-0.0485$ | $-0.0785$ | 0.551 | 0. | 0. | 0. |
| 0.869 | 1.31 | $-0.447$ | $-0.740$ | $-0.0588$ | $-0.0261$ | 0.317 | $-0.0835$ | 0.306 | 0.296 | 0. | 0. |
| 1. | 0.152 | $-0.0752$ | $-0.320$ | 0.107 | 0.313 | 0.1867 | 0.317 | 0.0427 | 0.132 | 0.143 | 0. |
| | $6.08 \cdot 10^{-5}$ | $6.25 \cdot 10^{-5}$ | $6.02 \cdot 10^{-2}$ | $4.50 \cdot 10^{-5}$ | $8.50 \cdot 10^{-5}$ | $6.83 \cdot 10^{-5}$ | $5.16 \cdot 10^{-1}$ | $5.12 \cdot 10^{-5}$ | $2.75 \cdot 10^{-1}$ | $2.76 \cdot 10^{-5}$ | $1.48 \cdot 10^{-1}$ |

Fig. 22. Learnt RUNGE-KUTTA coefficients during the the identification of the RUNGE-KUTTA scheme for system (22)

| | | | | |
|---|---|---|---|---|
| 0.0 | 9.41 | $-3.40$ | $-3.15$ | $-2.85$ |
| 0.203 | $-0.193$ | $-0.082$ | 0.061 | 0.416 |
| 0.518 | 0.002 | $-0.078$ | 0.004 | 0.590 |
| 1. | $-0.655$ | 0.517 | 0.506 | 0.631 |
| | 0.999 | $5.85 \cdot 10^{-6}$ | 0.0002 | $5.30 \cdot 10^{-6}$ |

Fig. 23. Learnt RUNGE-KUTTA coefficients during the the identification of the RUNGE-KUTTA scheme for system (24)

| | | | | |
|---|---|---|---|---|
| $0.$ | $-15.5$ | $3.58$ | $4.84$ | $7.05$ |
| $0.755$ | $19.1, -6.10$ | $-6.85$ | $-5.35$ | |
| $0.996$ | $33.0$ | $-8.55$ | $-12.7$ | $-10.8$ |
| $1.$ | $51.3$ | $-16.5$ | $-16.1$ | $-17.7$ |
| | $0.913$ | $0.0330$ | $0.0501$ | $0.00360$ |

Fig. 24. Learnt RUNGE-KUTTA coefficients during the the joint identification of the RUNGE-KUTTA scheme associated to system (24)