



HAL
open science

Reappraising double pendulum dynamics across multiple computational platforms

Sandy Herho, Faiz Fajary, Katarina Herho, Iwan Anwar, Rusmawan Suwarman,
Dasapta Erwin Irawan

► To cite this version:

Sandy Herho, Faiz Fajary, Katarina Herho, Iwan Anwar, Rusmawan Suwarman, et al.. Reappraising double pendulum dynamics across multiple computational platforms. 2024. <hal-04568479>

HAL Id: hal-04568479

<https://hal.science/hal-04568479v1>

Preprint submitted on 5 May 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons CC BY 4.0 - Attribution - International License

1 Reappraising double pendulum dynamics across
2 multiple computational platforms

3 Sandy H. S. Herho^{1*}, Faiz R. Fajary^{2, 3}, Katarina E. P. Herho⁴,
4 Iwan P. Anwar⁵, Rusmawan Suwarman³, Dasapta E. Irawan⁶

5 ^{1*}Department of Earth and Planetary Sciences, University of California,
6 900 University Ave., Riverside, 92521, CA, USA.

7 ²Coastal Hazards and Energy System Science (CHESS) Lab, Hiroshima
8 University, 1-5-1 Kagamiyama, Higashihiroshima, 739-8529, Hiroshima,
9 Japan.

10 ³Atmospheric Science Research Group, Bandung Institute of Technology
11 (ITB), Jalan Ganesha 10, Bandung, 40132, West Java, Indonesia.

12 ⁴Department of Geological Engineering, Trisakti University, Jalan
13 Letjen S. Parman 1, West Jakarta, 1440, DKI Jakarta, Indonesia.

14 ⁵Oceanography Research Group, Bandung Institute of Technology
15 (ITB), Jalan Ganesha 10, Bandung, 40132, West Java, Indonesia.

16 ⁶Applied Geology Research Group, Bandung Institute of Technology
17 (ITB), Jalan Ganesha 10, Bandung, 40132, West Java, Indonesia.

18 *Corresponding author(s). E-mail(s): sandy.herho@email.ucr.edu;

19 **Abstract**

20 This study presents the complexity and sensitivity of chaotic system dynamics
21 in the case of the double pendulum. It applied detailed numerical analyses of
22 the double pendulum in multiple computing platforms in order to demonstrate
23 the complexity in behavior of the system of double pendulums. The equations
24 of motion were derived from the Euler-Lagrange formalism, in order to capture
25 the system's dynamics, which is coupled nonlinearly. These were solved numeri-
26 cally using the efficient Runge-Kutta-Fehlberg method, implemented in Python,
27 R, GNU Octave, and Julia, while runtimes and memory usage were extensively
28 benchmarked across these environments. Time series analyses, including the cal-
29 culation of Shannon entropy and the Kolmogorov-Smirnov test, quantified the
30 system's unpredictability and sensitivity to infinitesimal perturbations of the ini-
31 tial conditions. Phase space diagrams illustrated the intricate trajectories and
32 strange attractors, as further confirmation of the chaotic nature of the double

33 pendulum. All the findings have a clear indication of the importance of accu-
34 rate measurements of the initial condition in a chaotic system, contributing to
35 an increased understanding of nonlinear dynamics. Future research directions are
36 faster simulations using Numba and GPU computing, stochastic effects, chaotic
37 synchronization, and applications in climate modeling. This work will be use-
38 ful for understanding chaos theory and efficient computational approaches in
39 complex systems of dynamical nature.

40 **Keywords:** Chaotic Dynamics, Computational Platforms, Nonlinear Dynamics,
41 Numerical Simulations, Sensitivity Analysis

42 1 Introduction

43 The double pendulum forms an interesting physics system consisting of two pendul-
44 ulum hanging in a rigidly connected line. It does not only rely on the observation of
45 immediate effects but also the study of the underlying structures that make things
46 change. Though at first glance this machine may look mundane, its behavior will prove
47 to be exciting and the underlying idea of balance between order and disorder that has
48 intrigued scientists and mathematicians for centuries will be inspirationally brought
49 to life. It is an example of the fascinating ways simple systems can exhibit complex
50 behavior. From people like Henri Poincaré, who first took a look at non-linear systems
51 and laid the foundation for chaos theory, to today's powerful computational models,
52 the double pendulum remains a symbol that the secret of nonlinear dynamics is a
53 profound one [1].

54 A double pendulum consists of two massless rods, each with a concentrated point
55 mass at its end, connected by a frictionless hinge. This seemingly simple setup
56 actually displays a remarkable sensitivity to initial conditions, a characteristic of
57 chaotic systems that was first discovered by British mathematician and physicist Mary
58 Cartwright in the early 20th century [1, 2]. Even tiny changes in the starting position
59 can cause significant divergences in the paths the system takes - a concept that chal-
60 lenges the traditional idea put forth by French scholar Pierre-Simon Laplace known
61 as Laplace's demon, which suggests that knowing all initial conditions of a system
62 guarantees the ability to predict its future evolution with total certainty [3].

63 The double pendulum is a prominently studied dynamical system in climate sci-
64 ence due to its exquisite sensitivity to initial conditions, making it a valuable model
65 for studying chaos theory and its applications in understanding Earth's complex cli-
66 mate systems [4]. Gaining insights into chaotic systems like the double pendulum
67 could prove vital for tackling one of our most pressing global issues - anthropogenic
68 climate change driven by human activities. Several studies have used simplified mod-
69 els like the double pendulum to gain insights into the non-linear dynamics underlying
70 atmospheric-oceanic flows and long-range climate predictions [e. g. 5–9].

71 The double pendulum exemplifies how deterministic systems can exhibit unpre-
72 dictable, chaotic behavior, bridging the gap between the simple, ordered world of
73 classical physics and the apparent randomness we observe in complex phenomena like

74 weather and climate patterns [10]. This chaotic complexity represents a major fron-
75 tier in our scientific understanding - we have robust theories for simple systems and
76 stochastic models for randomness, but lack a unified framework to explain the rich
77 dynamical behaviors that emerge in between the two extremes [11]. Unraveling the
78 chaos inherent in multi-scale systems like the double pendulum may unlock deeper
79 insights into the fundamental laws governing our climate and the universe at large.

80 In this paper, our main goal is to thoroughly investigate and analyze the complex
81 movements of a double pendulum system using detailed numerical simulations. We
82 utilized open-source computing platforms to develop engaging visuals that can be used
83 as effective educational resources. This graphical representation will help students,
84 especially those with limited abstract mathematics knowledge, better understand the
85 concepts of calculus of variations. The calculus of variations is a branch of mathe-
86 matical analysis concerned with optimizing functionals, which are mappings from a
87 space of functions to the real numbers. It addresses problems where the goal is to
88 find a function that extremizes a given functional, either minimizing or maximizing
89 it [12]. This field is particularly important in areas such as physics, engineering, and
90 economics, where one seeks to optimize quantities that depend on functions, such as
91 energy, action, or cost [13]. By presenting this information visually and in a practical
92 context, we hope to make it easier for students to connect theoretical concepts with
93 real-world applications.

94 Furthermore, allowing students to freely access the simulation code enables them to
95 explore the computational side of the project, enhancing their comprehension of how
96 theory translates into practice. This also gives them the opportunity to improve and
97 perfect the models, promoting a practical approach to learning physics and applied
98 mathematics through coding. To ensure optimal performance and efficiency, we thor-
99 oughly evaluated the free and open-source computing environments utilized in this
100 study. This study will offer helpful information for teachers, scholars, and profession-
101 als, helping them choose the best tools for their individual computing requirements
102 and improving resource management and promoting excellence in scientific computing.

103 2 Methods

104 In order to predict the positional paths of two point masses in a double pendulum
105 system, we need a classical mechanics framework represented by the Euler-Lagrange
106 equation. This is because the Euler-Lagrange equation provides a systematic approach
107 to deriving the equations of motion for complex systems like the double pendulum,
108 taking into account the constraints and forces involved. On the other hand, using New-
109 tonian mechanics alone for the double pendulum can be challenging due to the system's
110 nonlinear nature and the presence of constraints such as the lengths of the pendu-
111 lum arms. While it's possible to analyze simpler pendulum systems using Newton's
112 laws directly, the double pendulum's motion involves coupled, nonlinear differential
113 equations that are more conveniently handled using the Euler-Lagrange formalism [14].

114 The Euler-Lagrange equation is essential in classical mechanics and field theory,
115 describing the motion of particles or fields by minimizing a functional known as the
116 action. To derive this equation from scratch, we started with the action S , defined as

117 the integral of the Lagrangian (\mathcal{L}) over time:

$$S = \int_{t_1}^{t_2} \mathcal{L}(q, \dot{q}, t) dt \quad (1)$$

118 Here, q represents the generalized coordinates, \dot{q} is the derivative of q with respect to
119 time, and t is time. The Lagrangian (\mathcal{L}) has a simple and concise definition:

$$\mathcal{L} \equiv T - V \quad (2)$$

120 The kinetic energy (T) and potential energy (V) together make up the classical
121 Lagrangian, which is just the difference between these energies in the system. This
122 applies to classical mechanics with conservative systems, where the total energy is the
123 sum of kinetic and potential energies. Our next step was to find the path $q(t)$ that
124 keeps the action constant, even if the path changes a little. This basic idea is called
125 the principle of least action.

126 To find this stationary path, we used the calculus of variations. Let $\delta q(t)$ be a
127 small variation in the path $q(t)$, such that $q(t)$ becomes $q(t) + \delta q(t)$. The variation in
128 the action is then:

$$\delta S = \int_{t_1}^{t_2} [\mathcal{L}(q + \delta q, \dot{q} + \delta \dot{q}, t) - \mathcal{L}(q, \dot{q}, t)] dt \quad (3)$$

129 We expanded $\mathcal{L}(q + \delta q, \dot{q} + \delta \dot{q}, t)$ in a Taylor series around q and \dot{q} :

$$\mathcal{L}(q + \delta q, \dot{q} + \delta \dot{q}, t) = \mathcal{L}(q, \dot{q}, t) + \frac{\partial \mathcal{L}}{\partial q} \delta q + \frac{\partial \mathcal{L}}{\partial \dot{q}} \delta \dot{q} + \mathcal{O}(\delta q^2, \delta \dot{q}^2) \quad (4)$$

130 Substituting the expression back into equation 3 we would consider terms up to second
131 order or higher in δq and $\delta \dot{q}$ in the variation of the action S , $\mathcal{O}(\delta q^2, \delta \dot{q}^2)$:

$$\delta S = \int_{t_1}^{t_2} \left[\frac{\partial \mathcal{L}}{\partial q} \delta q + \frac{\partial \mathcal{L}}{\partial \dot{q}} \delta \dot{q} \right] dt + \mathcal{O}(\delta q^2, \delta \dot{q}^2) \quad (5)$$

132 Integrating the first term by parts with respect to t and assuming that variations
133 $\delta q(t_1)$ and $\delta q(t_2)$ vanish (boundary conditions), we got:

$$\delta S = \int_{t_1}^{t_2} \left[\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}} \right) - \frac{\partial \mathcal{L}}{\partial q} \right] \delta q dt \quad (6)$$

134 For the action to be stationary, δS must be zero for all possible variations δq . This
135 leads to the Euler-Lagrange equation:

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}} \right) - \frac{\partial \mathcal{L}}{\partial q} = 0 \quad (7)$$

136 This equation governs the dynamics of the system and provides the equations of motion
137 for the generalized coordinates $q(t)$.

138 To apply the Euler-Lagrange equation to the double pendulum system, we must
 139 first establish its Lagrangian, which encapsulates both the system's kinetic and poten-
 140 tial energies. This process started by delineating the geometric relationships governing
 141 the vertical and horizontal positions within the double pendulum system, as illus-
 142 trated in Fig. 1. These relationships were formalized through the following equations,
 143 denoted as equations 8 and 9:

$$x_1 = L_1 \sin(\theta_1) \tag{8}$$

$$x_2 = x_1 + L_2 \sin(\theta_2)$$

$$y_1 = L_1 \cos(\theta_1) \tag{9}$$

$$y_2 = y_1 + L_2 \cos(\theta_2)$$

144 Here, x_1 and x_2 represent the horizontal positions, while y_1 and y_2 represent the
 145 vertical positions. These positions are determined by the lengths of the pendulum
 146 arms (L_1 and L_2) and the angles (θ_1 and θ_2).

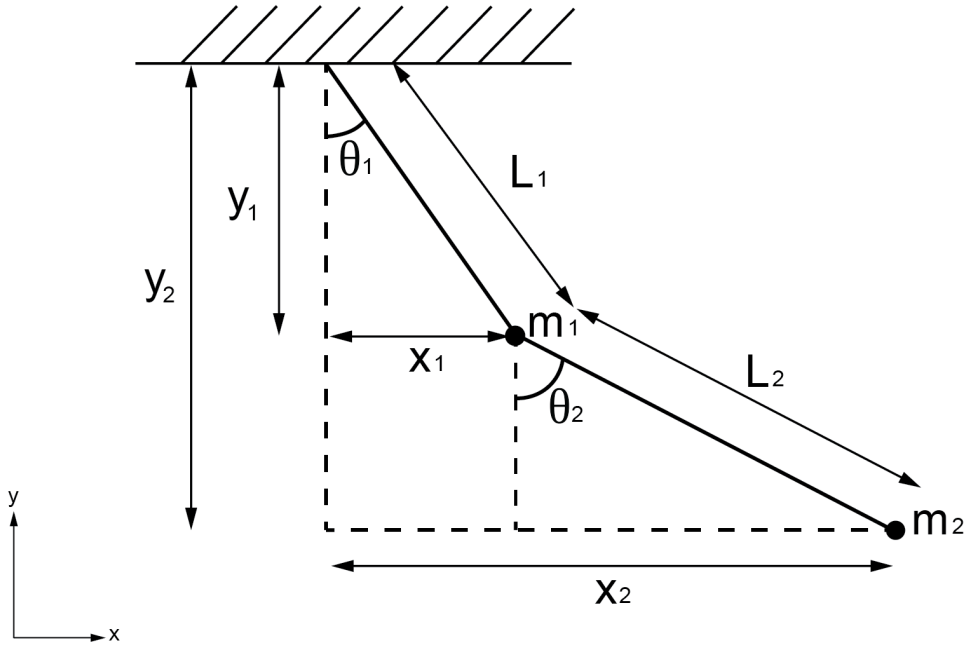


Fig. 1: Free-body diagram of a double pendulum. Point masses (m_1, m_2) connected by massless rods (L_1, L_2). Angles θ_1 and θ_2 represent deviation from the vertical axis.

147 Since the positions x and y are functions of the angles θ_1 and θ_2 , respectively,
 148 obtaining their first derivatives requires applying the chain rule. This application
 149 results in the following expressions:

$$\begin{aligned} \dot{x}_1 &= L_1 \dot{\theta}_1 \cos(\theta_1) \\ \dot{x}_2 &= L_1 \dot{\theta}_1 \cos(\theta_1) + L_2 \dot{\theta}_2 \cos(\theta_2) \end{aligned} \quad (10)$$

150

$$\begin{aligned} \dot{y}_1 &= -L_1 \dot{\theta}_1 \sin(\theta_1) \\ \dot{y}_2 &= -L_1 \dot{\theta}_1 \sin(\theta_1) - L_2 \dot{\theta}_2 \cos(\theta_2) \end{aligned} \quad (11)$$

151 These derivatives $\dot{x}_1, \dot{x}_2, \dot{y}_1$, and \dot{y}_2 represent the rates of change of the horizontal and
 152 vertical positions with respect to time, taking into account the angular velocities $\dot{\theta}_1$
 153 and $\dot{\theta}_2$.

154 Starting from the general formula for kinetic energy $T = \frac{1}{2}mv^2$ (where m is the
 155 total of two-point masses of $m_1 + m_2$ and v is velocity), we substituted the velocities
 156 $\dot{x}_1, \dot{x}_2, \dot{y}_1$, and \dot{y}_2 from the systems of equations 10 and 11 into equation 12. This
 157 substitution yields the expanded form:

$$\begin{aligned} T &= \frac{1}{2} (m_1(\dot{x}_1^2 + \dot{y}_1^2) + m_2(\dot{x}_2^2 + \dot{y}_2^2)) \\ &= \frac{1}{2} \left(m_1 \left(L_1 \dot{\theta}_1 \cos(\theta_1) \right)^2 + m_1 \left(-L_1 \dot{\theta}_1 \sin(\theta_1) \right)^2 \right. \\ &\quad \left. + m_2 \left(L_1 \dot{\theta}_1 \cos(\theta_1) + L_2 \dot{\theta}_2 \cos(\theta_2) \right)^2 + m_2 \left(-L_1 \dot{\theta}_1 \sin(\theta_1) - L_2 \dot{\theta}_2 \cos(\theta_2) \right)^2 \right) \end{aligned} \quad (12)$$

158 Expanding and simplifying each term step by step, we obtained expressions for the
 159 kinetic energy components. These components involve terms related to the masses m_1
 160 and m_2 , the lengths L_1 and L_2 of the pendulum arms, and the angular velocities $\dot{\theta}_1$
 161 and $\dot{\theta}_2$. After combining and simplifying the terms, we arrived at the final form of the
 162 kinetic energy:

$$T = \frac{1}{2} (m_1 + m_2) L_1^2 \dot{\theta}_1^2 + \frac{1}{2} m_2 L_2^2 \dot{\theta}_2^2 + m_2 L_1 L_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2) \quad (13)$$

163 This expression captures the kinetic energy T of the double pendulum system compre-
 164 hensively, incorporating the masses m_1 and m_2 , the lengths L_1 and L_2 of the pendulum
 165 arms, and the angular velocities $\dot{\theta}_1$ and $\dot{\theta}_2$ in a way that reflects the system's dynamics
 166 and interactions.

167 Beginning with equation 14, which defines potential energy as a function of the
 168 vertical positions y_1 and y_2 , and the gravitational constant g , we can express it as:

$$V = m_1 g y_1 + m_2 g y_2 \quad (14)$$

169 Substituting the expressions for y_1 and y_2 from the system of equations 9 into the
 170 above equation yields:

$$V = m_1 g(L_1 \cos(\theta_1)) + m_2 g(L_1 \cos(\theta_1) + L_2 \cos(\theta_2)) \quad (15)$$

171 We then simplified this expression to:

$$V = g((m_1 + m_2)L_1 \cos(\theta_1) + m_2 L_2 \cos(\theta_2)) \quad (16)$$

172 Finally, to express potential energy V solely in terms of the angles θ_1 and θ_2 , we
 173 derived:

$$V = -g((m_1 + m_2)L_1 \cos(\theta_1) + m_2 L_2 \cos(\theta_2)) \quad (17)$$

174 Here, g represents the gravitational acceleration, and the potential energy V is derived
 175 from the vertical positions of the pendulum components and their respective masses,
 176 articulated in terms of the angles θ_1 and θ_2 .

177 After deriving the expressions for kinetic energy T and potential energy V , we can
 178 now formulate the Lagrangian \mathcal{L} for the double pendulum system. The Lagrangian is
 179 defined in the definition 2. Substituting the expressions we derived for T and V into
 180 this equation, we obtained:

$$\begin{aligned} \mathcal{L} &= \frac{1}{2}(m_1 + m_2)L_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2L_2^2\dot{\theta}_2^2 + m_2L_1L_2\dot{\theta}_1\dot{\theta}_2 \cos(\theta_1 - \theta_2) \\ &\quad - (-g((m_1 + m_2)L_1 \cos(\theta_1) + m_2 L_2 \cos(\theta_2))) \\ &= \frac{1}{2}(m_1 + m_2)L_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2L_2^2\dot{\theta}_2^2 + m_2L_1L_2\dot{\theta}_1\dot{\theta}_2 \cos(\theta_1 - \theta_2) \\ &\quad + g((m_1 + m_2)L_1 \cos(\theta_1) + m_2 L_2 \cos(\theta_2)) \end{aligned} \quad (18)$$

181 This expression for the Lagrangian \mathcal{L} encapsulates the dynamic behavior of the
 182 double pendulum system. It incorporates the masses m_1 and m_2 , the lengths L_1 and
 183 L_2 of the pendulum arms, the angular velocities $\dot{\theta}_1$ and $\dot{\theta}_2$, and the gravitational
 184 constant g , as well as the angles θ_1 and θ_2 that describe the positions of the pendulum
 185 components. The Lagrangian \mathcal{L} serves as a fundamental quantity in the analysis of
 186 the system's motion and dynamics, providing a comprehensive representation of its
 187 energy and interactions.

188 Starting with the Lagrangian \mathcal{L} derived earlier (equation 18), we applied the Euler-
 189 Lagrange equation (equation 7) to derive the equations of motion for a system with
 190 two point masses (the double pendulum in this case). The Euler-Lagrange equation
 191 for a variable q_i is given by:

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i} \right) - \frac{\partial \mathcal{L}}{\partial q_i} = 0 \quad (19)$$

192 Applying this equation to the variables θ_1 and θ_2 separately, we obtained two set of
 193 equations:

$$\begin{aligned}\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_1} \right) - \frac{\partial \mathcal{L}}{\partial \theta_1} &= 0 \\ \frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2} \right) - \frac{\partial \mathcal{L}}{\partial \theta_2} &= 0\end{aligned}\quad (20)$$

194 For θ_1 , we first calculated the partial derivative of \mathcal{L} with respect to $\dot{\theta}_1$:

$$\frac{\partial \mathcal{L}}{\partial \dot{\theta}_1} = (m_1 + m_2)L_1^2\dot{\theta}_1 + m_2L_1L_2\dot{\theta}_2 \cos(\theta_1 - \theta_2) \quad (21)$$

195 Then, we took the derivative of this with respect to time t :

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_1} \right) = (m_1 + m_2)L_1^2\ddot{\theta}_1 + m_2L_1L_2\ddot{\theta}_2 \cos(\theta_1 - \theta_2) \quad (22)$$

196 Next, we calculated the partial derivative of \mathcal{L} with respect to θ_1 :

$$\frac{\partial \mathcal{L}}{\partial \theta_1} = -m_2L_1L_2\dot{\theta}_1\dot{\theta}_2 \sin(\theta_1 - \theta_2) - g(m_1 + m_2)L_1 \sin(\theta_1) \quad (23)$$

197 Finally, substituting these derivatives into the Euler-Lagrange equation (equation 20)
 198 for θ_1 :

$$(m_1 + m_2)L_1^2\ddot{\theta}_1 + m_2L_1L_2\ddot{\theta}_2 \cos(\theta_1 - \theta_2) = 0 \quad (24)$$

199 To derive the equation of motion for θ_2 using the Euler-Lagrange equation
 200 (equation 20), we started by calculating the partial derivative of the Lagrangian \mathcal{L}
 201 with respect to the derivative of θ_2 , denoted as $\dot{\theta}_2$. This yields:

$$\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2} = m_2L_2^2\dot{\theta}_2 + m_2L_1L_2\dot{\theta}_1 \cos(\theta_1 - \theta_2) \quad (25)$$

202 Taking the time derivative of this expression gave us:

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2} \right) = m_2L_2^2\ddot{\theta}_2 + m_2L_1L_2\ddot{\theta}_1 \cos(\theta_1 - \theta_2) - m_2L_1L_2\dot{\theta}_1\dot{\theta}_2 \sin(\theta_1 - \theta_2) \quad (26)$$

203 Next, we calculated the partial derivative of \mathcal{L} with respect to θ_2 , denoted as $\frac{\partial \mathcal{L}}{\partial \theta_2}$,
 204 which was given by:

$$\frac{\partial \mathcal{L}}{\partial \theta_2} = -m_2L_1L_2\dot{\theta}_1\dot{\theta}_2 \sin(\theta_1 - \theta_2) - m_2L_2g \sin(\theta_2) \quad (27)$$

205 Substituting the expressions for $\frac{\partial \mathcal{L}}{\partial \dot{\theta}_2}$ and $\frac{\partial \mathcal{L}}{\partial \theta_2}$ into the Euler-Lagrange equation and
 206 simplifying this equation further results in the equation of motion for θ_2 , given as:

$$m_2L_2\ddot{\theta}_2 + m_2L_1\ddot{\theta}_1 \cos(\theta_1 - \theta_2) - m_2L_1\dot{\theta}_1^2 \sin(\theta_1 - \theta_2) + m_2g \sin(\theta_2) = 0 \quad (28)$$

207 Combining the final forms of the Euler-Lagrange equations for θ_1 (equation 24)
 208 and θ_2 (equation 28) together form the complete system of equations of motion for
 209 the double pendulum system. They are coupled second-order ordinary differential
 210 equations (ODEs) because the acceleration terms $\ddot{\theta}_1$ and $\ddot{\theta}_2$ are dependent on each
 211 other due to the cosine and sine terms involving θ_1 and θ_2 . This coupling reflects
 212 the interdependence of the pendulum's motions and positions, making the system
 213 dynamically rich and challenging to analyze without numerical or advanced analytical
 214 techniques.

215 For the numerical analysis, we substituted the following expressions ($\dot{\theta}_1 =$
 216 ω_1 , $\dot{\theta}_2 = \omega_2$, $\Delta\theta = \theta_1 - \theta_2$) to simplify the system of equations for computa-
 217 tional purposes. Substituting these into the given equations (equations 24 and 28), we
 218 obtained the following system:

$$\begin{aligned} (m_1 + m_2)L_1\dot{\omega}_1 + m_2L_2\dot{\omega}_2 \cos(\Delta\theta) + m_2L_2\omega_2^2 \sin(\Delta\theta) + (m_1 + m_2)g \sin \theta_1 &= 0 \\ m_2L_2\dot{\omega}_2 + m_2L_1\dot{\omega}_1 \cos(\Delta\theta) - m_2L_1\omega_1^2 \sin(\Delta\theta) + m_2g \sin(\theta_2) &= 0 \end{aligned} \quad (29)$$

219 This transformed system allows us to numerically solve for the angular frequencies ω_1
 220 and ω_2 given initial conditions and system parameters.

221 To simplify the numerical analysis, we first rewrote the original system of equations
 222 in terms of the defined variables:

$$\begin{cases} \alpha = (m_1 + m_2)L_1 \\ \beta = m_2L_2 \cos(\Delta\theta) \\ \gamma = m_2L_1 \cos(\Delta\theta) \\ \delta = m_2L_2 \\ \varepsilon = -m_2L_2\omega_2^2 \sin(\Delta\theta) - m_2g \sin(\theta_2) \\ \zeta = m_2L_2\omega_1^2 \sin(\Delta\theta) - m_2g \sin(\theta_2) \end{cases} \quad (30)$$

223 We then rewrote the system of equations in matrix form:

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \begin{pmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \end{pmatrix} = \begin{pmatrix} \varepsilon \\ \zeta \end{pmatrix} \quad (31)$$

224 Finally, solving for $\dot{\omega}_1$ and $\dot{\omega}_2$, we obtained:

$$\begin{aligned} \dot{\omega}_1 &= \frac{\varepsilon\delta - \beta\zeta}{\alpha\delta - \beta\gamma} \\ \dot{\omega}_2 &= \frac{\alpha\zeta - \gamma\varepsilon}{\alpha\delta - \beta\gamma} \end{aligned} \quad (32)$$

225 These derived equations were expressed in terms of the defined variables, providing a
226 more organized and manageable representation for the numerical experiments.

227 For the numerical simulation needs in this study, we used parameters consistent
228 with the physical properties of the system. These include an acceleration due to gravity
229 (g) of 9.8 m/s^2 , the lengths of the pendulum arms L_1 and L_2 set to 2 meters and 1 meter
230 respectively, and the masses of the pendulums (m_1 and m_2) set at 1 kilogram and 2
231 kilograms respectively. The simulation time (t) is chosen to be 10 seconds with 10,000
232 linearly time spacing, providing sufficient duration to observe the system's behavior.

233 With these parameters established, we set the initial conditions for the simulation.
234 The initial angles (θ_1 and θ_2) are set to 3.14 radians (which is equivalent to 180°)
235 for θ_1 and 1.57 radians (equivalent to 90°) for θ_2 . Additionally, the initial angular
236 velocities (ω_1 and ω_2) are initialized to 0 radians per second, representing a starting
237 point where the pendulums are at rest. For the subsequent simulation, we changed
238 the both angular velocities to 0.001 radians per seconds fo the sensitivity test with
239 the initial conditions.

240 We used the Runge-Kutta-Fehlberg (RKF) method for approximating the solution
241 of our problem (system of equations 32). RKF is a powerful numerical integration
242 technique used to solve systems of ODEs with high accuracy and computational effi-
243 ciency [13]. Its adaptability makes it particularly suitable for dynamic systems like the
244 double pendulum, where the motion can be complex and highly nonlinear.

245 We started with the initial conditions and set the initial step size h . Then, we
246 proceeded to define the predictor step by using the 4th order Runge-Kutta formulas to
247 predict the solution at the next time step. For a general ODE of the form $\frac{dy}{dt} = f(t, y)$,
248 the 4th order Runge-Kutta formulas are:

$$\begin{aligned}k_1 &= hf(t_n, y_n) \\k_2 &= hf(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \\k_3 &= hf(t_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \\k_4 &= hf(t_n + h, y_n + k_3)\end{aligned}\tag{33}$$

249 The predicted solution at t_{n+1} is then given by:

$$y_{n+1}^{(4)} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)\tag{34}$$

250 We used the 5th order Runge-Kutta formulas to compute a more accurate estimate of
251 the solution at the next time step. The 5th order Runge-Kutta formulas are similar

252 to the 4th order ones but include an additional evaluation point:

$$\begin{aligned}
k_1 &= hf(t_n, y_n) \\
k_2 &= hf(t_n + \frac{h}{4}, y_n + \frac{k_1}{4}) \\
k_3 &= hf(t_n + \frac{3h}{8}, y_n + \frac{3k_1}{32} + \frac{9k_2}{32}) \\
k_4 &= hf(t_n + \frac{12h}{13}, y_n + \frac{1932k_1}{2197} - \frac{7200k_2}{2197} + \frac{7296k_3}{2197}) \\
k_5 &= hf(t_n + h, y_n + \frac{439k_1}{216} - 8k_2 + \frac{3680k_3}{513} - \frac{845k_4}{4104}) \\
k_6 &= hf(t_n + \frac{h}{2}, y_n - \frac{8k_1}{27} + 2k_2 - \frac{3544k_3}{2565} + \frac{1859k_4}{4104} - \frac{11k_5}{40})
\end{aligned} \tag{35}$$

253 The corrected solution at t_{n+1} is then given by:

$$y_{n+1}^{(5)} = y_n + \frac{1}{90}(7k_1 + 32k_3 + 12k_4 + 32k_5 + 7k_6) \tag{36}$$

254 Then, we calculated the local truncation error by comparing the 4th and 5th order
255 solutions. The error estimate ϵ_n is given by:

$$\epsilon_n = |y_{n+1}^{(5)} - y_{n+1}^{(4)}| \tag{37}$$

256 We compared the error estimate ϵ_n to a predefined tolerance. If ϵ_n is within the
257 tolerance, accept the step and update the solution. If ϵ_n exceeds the tolerance, reduce
258 the step size h and we repeated the process until the error is acceptable. Finally, we
259 continued integrating until reaching the desired end time or number of steps.

260 The adaptive step-size control implemented in the RKF method ensures accurate
261 integration while minimizing computational costs. By dynamically adjusting the step
262 size based on error estimation, the RKF method provides accurate numerical approxi-
263 mations of the system's behavior over time, rendering it an effective tool for analyzing
264 complex dynamic systems such as the double pendulum. However, in the present study,
265 we did not employ the RKF method directly from the beginning. Instead, we uti-
266 lized several platform-specific tools, including SciPy in Python [15], deSolve in R [16],
267 DifferentialEquations.jl in Julia [17], and the built-in function ode45 in GNU Octave
268 [18].

269 The motivation behind using multiple programming languages and their respective
270 computing environments was twofold. First, it allowed us to leverage the strengths and
271 unique features of each language and environment, enabling a comprehensive explo-
272 ration of the double pendulum problem from diverse perspectives. Second, it facilitated
273 a comparative analysis of the performance and accuracy of different numerical solvers
274 across these platforms.

275 In Python, the SciPy library provided a robust and well-established collection of
276 scientific computing tools, including numerical solvers for ODEs. The flexibility and
277 ease of use of Python, combined with the power of SciPy, made it a suitable choice for

278 implementing and testing numerical methods for the double pendulum problem [19,
279 20]. However, the interpreted nature of Python may introduce performance overhead
280 compared to compiled languages.

281 The R programming language, with its deSolve package, offered a specialized envi-
282 ronment for solving ODEs and differential algebraic equations (DAEs) [21]. R’s strong
283 emphasis on statistical analysis and data visualization made it an attractive option
284 for exploring the double pendulum problem, enabling efficient data analysis and visual
285 representation of the results. Nonetheless, the high-level nature of R may lead to
286 performance limitations for computationally intensive simulations.

287 Julia, a relatively new language designed for scientific computing, provided a high-
288 performance computing environment through its DifferentialEquations.jl package. The
289 combination of Julia’s dynamic programming capabilities and its efficient just-in-time
290 (JIT) compilation made it a promising choice for solving the double pendulum problem
291 with potentially improved computational performance [22, 23]. However, the relatively
292 young ecosystem of Julia may present challenges in terms of package maturity and
293 community support.

294 Finally, GNU Octave, a high-level language primarily intended for numerical
295 computations, offered the built-in ode45 function, which implements a versatile Runge-
296 Kutta method for solving ODEs. GNU Octave’s compatibility with MATLAB® syntax
297 and its open-source nature made it an accessible option for researchers and students
298 alike [18]. However, its performance may be limited compared to lower-level languages
299 or specialized numerical libraries. By employing these various computing environ-
300 ments, we aimed to evaluate the trade-offs between performance, accuracy, and ease
301 of use for each approach.

302 When conducting scientific research involving computational simulations, it is crucial
303 to ensure reproducibility and transparency in the methods used. In this particular
304 study, we aim to benchmark the performance of four different computing environ-
305 ments: Python, R, Octave/MATLAB, and Julia, by running a script that simulates the
306 motion of a double pendulum. The script was executed 1,000 times in each computing
307 environment, and the runtime and memory usage for each run will be measured and
308 recorded. The benchmarking was performed on a Fedora Linux 39 (Budgie) x86_64
309 system with a 20LB0021US ThinkPad P52s laptop equipped with an Intel i7-8550U
310 (8) @4.000GHz CPU.

311 The rationale behind this benchmarking approach is multifaceted. Firstly, it pro-
312 motes reproducibility by providing the scripts and the benchmarking script, allowing
313 other researchers to easily replicate the computational experiments and verify the
314 results. Additionally, it facilitates a direct comparison of the runtime and memory
315 usage across different computing environments for the same task, providing valuable
316 insights into the relative performance of each environment. This information can guide
317 researchers in choosing the most suitable tool for their specific computational needs.

318 Furthermore, running the scripts 1,000 times in each environment helps to account
319 for potential variability and ensures that the performance measurements are consis-
320 tent and reliable. This is particularly important when dealing with computationally
321 intensive simulations like the double pendulum simulation, where minor fluctuations
322 in hardware or software configurations can impact the results. With a large number

323 of runs, it becomes possible to perform statistical analysis on the collected data, such
 324 as calculating confidence intervals, identifying outliers, and determining the statistical
 325 significance of any observed performance differences.

326 Measuring memory usage alongside runtime can provide insights into the scala-
 327 bility and resource requirements of each computing environment. This information is
 328 crucial when working with large-scale simulations or data-intensive applications, where
 329 efficient memory management is essential. By including the benchmarking method-
 330 ology and results in the scientific paper, researchers demonstrate transparency and
 331 allow others to critically evaluate the computational approaches used in the study
 332 [24]. This aligns with the principles of open science and facilitates future replication
 333 and extension of the research.

334 After conducting the benchmarking process and collecting the runtime and mem-
 335 ory usage data for each computing environment, we performed statistical analyses to
 336 determine if there were significant differences in performance among the platforms.
 337 Specifically, we employed the Kruskal-Wallis (K-W) test and Dunn’s test with Bon-
 338 ferroni adjustment, which are commonly used in various scientific fields for comparing
 339 multiple groups or treatments.

340 The K-W test is a non-parametric alternative to the one-way analysis of variance
 341 (ANOVA) and is used when the assumptions of normality and homogeneity of vari-
 342 ances are violated [25]. It is a rank-based test that evaluates whether the populations
 343 from which the samples were drawn have the same distribution. The test statistic for
 344 the K-W test was calculated as:

$$H = \frac{12}{N(N+1)} \sum_{i=1}^k \frac{R_i^2}{n_i} - 3(N+1) \quad (38)$$

345 , where N is the total number of observations across all groups, k is the number of
 346 groups, R_i is the sum of ranks for group i , and n_i is the number of observations in
 347 group i . The null hypothesis for the K-W test is that the populations have the same
 348 distribution, while the alternative hypothesis is that at least one population has a
 349 different distribution from the others.

350 If the K-W test indicates significant differences among the groups, post-hoc tests
 351 are typically performed to determine which specific groups differ from each other.
 352 In our case, we used Dunn’s test with Bonferroni adjustment, which is a multiple
 353 comparison procedure that adjusts the significance level to control the family-wise
 354 error rate (FWER) [26]. The Dunn’s test statistic for comparing groups i and j was
 355 calculated as:

$$Z = \frac{R_i - R_j}{\sqrt{\frac{N(N+1)}{12} \left(\frac{1}{n_i} + \frac{1}{n_j} \right)}} \quad (39)$$

356 , where R_i and R_j are the average ranks for groups i and j , respectively, N is the total
 357 number of observations across all groups, and n_i and n_j are the number of observations
 358 in groups i and j , respectively. The Bonferroni adjustment was applied by dividing the
 359 desired significance level (α) by the number of pairwise comparisons made, resulting
 360 in an adjusted significance level of $\alpha/\binom{k}{2}$, where k is the number of groups.

361 The K-W test and Dunn’s test with Bonferroni adjustment are widely used in
 362 various scientific fields [e. g. 27–29]. These tests are particularly useful when dealing
 363 with non-normal data or when the assumptions of parametric tests (such as ANOVA)
 364 are violated. They provide a robust and reliable way to compare multiple groups or
 365 treatments, ensuring that any observed differences are statistically significant and not
 366 due to chance alone. By applying these statistical tests to our benchmarking data,
 367 we aimed to determine if there were significant differences in performance among the
 368 four computing environments (Python, R, GNU Octave, and Julia) for the double
 369 pendulum simulation task. These non-parametric procedures were implemented in
 370 Python using the SciPy stats module [15] and the scikit-posthoc library [30].

371 After conducting the benchmarking process to measure the runtime and memory
 372 usage of the double pendulum simulation across different computing environments,
 373 we further analyzed the obtained time series data to gain insights into the underlying
 374 dynamics of the system. One of the analysis techniques we employed was the calcu-
 375 lation of Shannon entropy [31], which is a measure derived from information theory
 376 that quantifies the amount of information or uncertainty present in a random variable
 377 or time series. The Shannon entropy was calculated using the following equation:

$$H(X) = - \sum_{i=1}^n p(x_i) \log_2 p(x_i) \quad (40)$$

378 , where $H(X)$ is the Shannon entropy, n is the number of unique values in the time
 379 series, and $p(x_i)$ is the probability of the occurrence of the value x_i in the time series.
 380 A higher Shannon entropy value indicates a higher degree of uncertainty or unpre-
 381 dictability in the time series, while a lower entropy value suggests a more predictable
 382 or regular pattern. This measure can provide valuable insights into the complexity
 383 and dynamics of the double pendulum system.

384 In our analysis, we calculated the Shannon entropy for each variable (e.g., x_1 , y_1 ,
 385 θ_1 , θ_2 , ω_1 , ω_2) in the time series data obtained from the double pendulum simula-
 386 tion. To interpret the entropy scores, we established thresholds based on the following
 387 criteria: low entropy (predictable) for entropy scores ≤ 0.5 , medium entropy (some pre-
 388 dictability) for entropy scores between 0.5 and 1.0, and high entropy (unpredictable)
 389 for entropy scores > 1.0 . These thresholds are commonly used in various applications
 390 and provide a convenient way to interpret the degree of predictability or uncertainty
 391 present in the time series data. By calculating and analyzing the Shannon entropy of
 392 the time series data, we can gain insights into the predictability and complexity of the
 393 double pendulum system’s dynamics. A high entropy score for a particular variable
 394 suggests that the corresponding time series is highly unpredictable or complex, while
 395 a low entropy score indicates a more regular or predictable pattern.

396 The choice of Shannon entropy as an analysis technique was motivated by its strong
 397 theoretical foundation in information theory and its widespread use in various scientific
 398 fields for quantifying the complexity and uncertainty of dynamical systems. Further-
 399 more, the interpretation of entropy scores based on predefined thresholds provides a
 400 convenient and standardized way to categorize the time series data into different levels
 401 of predictability or complexity.

402 After conducting the entropy measurement, we employed the Kolmogorov-Smirnov
 403 (K-S) test to assess the sensitivity of the system to initial conditions, which is a
 404 characteristic feature of chaotic systems. The K-S test is a non-parametric statistical
 405 test that compares the cumulative distribution functions (CDFs) of two samples to
 406 determine if they are drawn from the same underlying distribution [32]. In the context
 407 of dynamical systems analysis, the K-S test can be used to compare the time series
 408 obtained from the original system with slightly perturbed initial conditions, allowing
 409 us to quantify the divergence between the trajectories. Let $F(x)$ and $G(x)$ be the
 410 empirical CDFs of the original time series and the perturbed time series, respectively.
 411 The K-S statistic is defined as the maximum absolute difference between these two
 412 CDFs:

$$D_{n,m} = \sup_x |F(x) - G(x)| \quad (41)$$

413 , where \sup_x represents the supremum (least upper bound) of the set of absolute
 414 differences between the CDFs over all possible values of x .

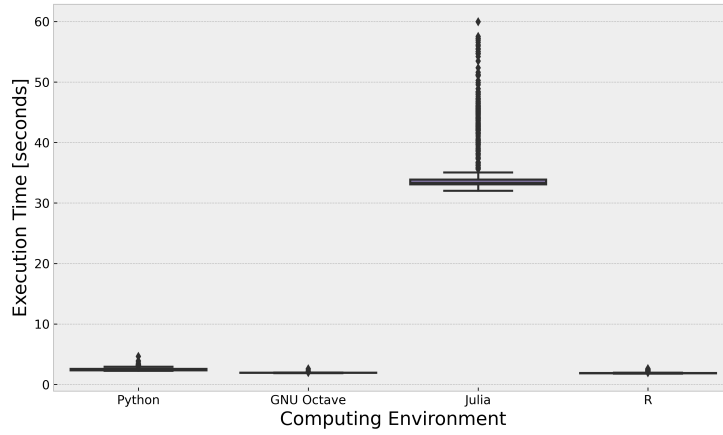
415 The null hypothesis for the K-S test is that the two samples are drawn from
 416 the same continuous distribution, while the alternative hypothesis is that they are
 417 drawn from different distributions. The null hypothesis is rejected if the K-S statis-
 418 tic $D_{n,m}$ exceeds a critical value that depends on the chosen significance level and
 419 the sample sizes n and m . In our analysis, we compared the original time series
 420 $(x_1, y_1, \theta_1, \theta_2, \omega_1, \omega_2)$ obtained from the double pendulum simulation with slightly per-
 421 turbed time series, where the initial conditions for ω_1 and ω_2 were perturbed by 0.001
 422 rad/s. By applying the K-S test to each pair of original and perturbed time series, we
 423 can assess whether the small perturbation in the initial conditions leads to a signifi-
 424 cant divergence in the trajectories over time. If the K-S test rejects the null hypothesis,
 425 indicating that the original and perturbed time series are drawn from different distri-
 426 butions, it suggests that the double pendulum system is sensitive to initial conditions,
 427 which is a hallmark of chaotic behavior. Conversely, if the test fails to reject the null
 428 hypothesis, it implies that the system is less sensitive to small perturbations in the
 429 initial conditions, suggesting a more predictable or regular dynamics.

430 The choice of the K-S test was motivated by its nonparametric nature, which means
 431 that it does not make assumptions about the underlying distribution of the data,
 432 making it suitable for analyzing complex dynamical systems where the distribution is
 433 often unknown or difficult to model parametrically. Additionally, the K-S test is widely
 434 used in various scientific fields for comparing distributions and detecting deviations
 435 from a hypothesized distribution [e. g. 33–35], further justifying its application in our
 436 analysis. We employed SciPy’s stats module [15] to conduct an automated K-S test.

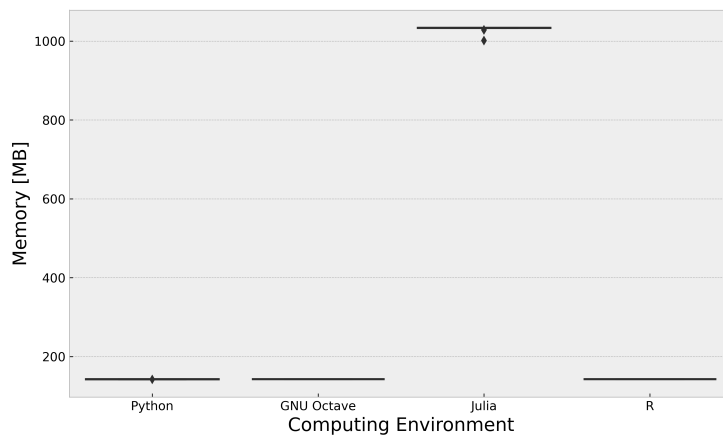
437 3 Results and Discussion

438 The K-W test, a non-parametric test for comparing multiple groups, yielded a test
 439 statistic of 3523.203 and a p-value of 0.000 for the runtime data, indicating that at
 440 least one group’s median runtime significantly differs from the others. Dunn’s post-
 441 hoc test, with Bonferroni adjustment for multiple comparisons, showed that all pairs
 442 of groups had p-values below 0.05, indicating significant differences in their median
 443 runtimes.

444 The results (Fig. 2a) indicate that R and GNU Octave exhibited the fastest run-
445 times, with mean values of 1.914 seconds and 1.944 seconds, respectively. The fast
446 performance of R can be attributed to the use of the deSolve package, which provides
447 efficient solvers for ODEs. GNU Octave, on the other hand, likely leverages opti-
448 mized numerical libraries or solvers for ODE systems. Python followed closely with
449 a mean runtime of 2.503 seconds, benefiting from the SciPy and NumPy libraries,
450 which provide efficient numerical computations and ODE solvers. Notably, Julia had
451 the slowest runtime performance, with a mean of 35.701 seconds, which is signifi-
452 cantly longer than the other environments. One potential factor contributing to Julia's
453 slower performance could be the overhead associated with its JIT compilation process.
454 JIT compilation can introduce additional runtime overhead, particularly for compu-
455 tationally intensive tasks like solving ODEs, which may have impacted Julia's overall
456 runtime performance in this specific implementation.



(a)



(b)

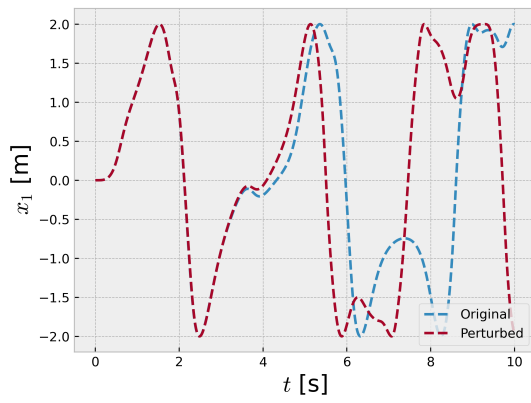
Fig. 2: Comparison of (a) execution time and (b) memory usage across different computing environments.

457 The K-W test for memory usage data yielded a test statistic of 3375.563 and a p-
 458 value of 0.000, indicating that at least one group's median memory usage significantly
 459 differs from the others. Dunn's post-hoc test showed that the pairs of groups with p-
 460 values below 0.05, indicating significant differences in their median memory usage, were
 461 GNU Octave-Julia, GNU Octave-Python, Julia-Python, and Julia-R. The results (Fig.
 462 2b) show that Python, R, and GNU Octave exhibited similar memory usage patterns,
 463 with a mean of approximately 142.84 MB. Julia, on the other hand, had significantly
 464 higher memory usage, with a mean of 1031.703 MB, which is about seven times higher
 465 than the other environments. The differences in memory usage across the computing
 466 environments can be attributed to various factors, such as the memory management

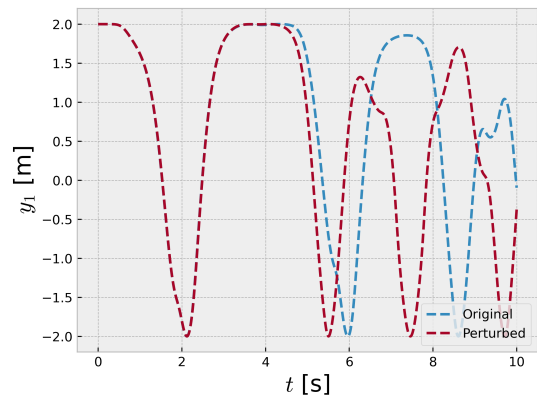
467 strategies of the respective programming languages and the specific implementation
468 of the double pendulum simulation in each environment. However, it is worth noting
469 that the pure Julia implementation using DifferentialEquations.jl may have different
470 memory requirements or optimization strategies compared to the packages or libraries
471 used in the other environments.

472 When selecting the appropriate computing environment for the double pendulum
473 simulation, the statistical significance of the runtime and memory usage differences
474 should be considered alongside other factors, such as existing codebase, familiarity
475 with the language, and potential optimizations. If runtime performance is the primary
476 concern, R and GNU Octave would be the recommended choices based on the pro-
477 vided results, with R leveraging the deSolve package and GNU Octave likely utilizing
478 optimized numerical libraries or solvers for ODE systems. Python, with the SciPy and
479 NumPy libraries, also exhibited a relatively good runtime performance and could be
480 a viable option, especially if existing Python code or familiarity with the language
481 is a consideration. If memory usage is a critical factor and the higher memory foot-
482 print of Julia is not a concern, Julia could be considered, potentially with further
483 optimization efforts or alternative implementations. It is important to note that the
484 pure Julia implementation using DifferentialEquations.jl may have different memory
485 requirements or optimization strategies compared to the packages or libraries used in
486 the other environments.

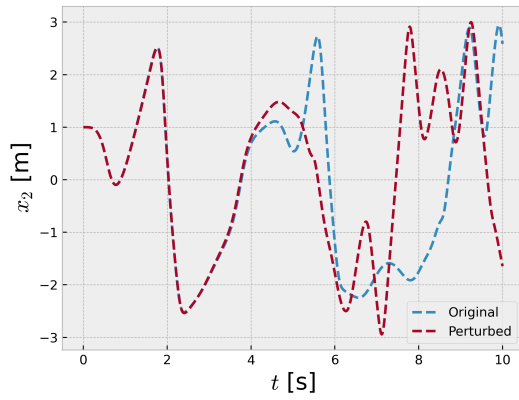
487 Figure 3 and Fig. 4 present time series plots of a double pendulum system with
488 slightly different initial angular velocities for the inner and outer pendulum. Each
489 subplot displays the time evolution of the angular positions of both pendulums over
490 10 seconds with 1000 time steps.



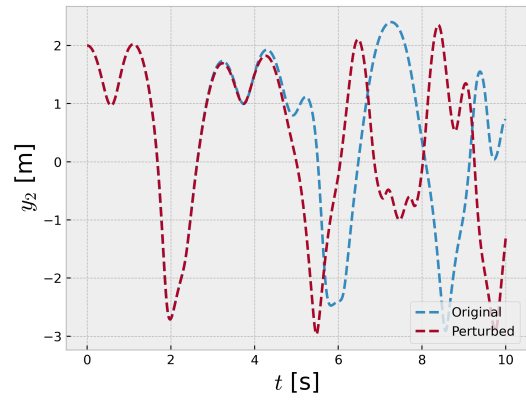
(a)



(b)



(c)



(d)

Fig. 3: Time series of a double pendulum positions with slightly different initial angular velocities for (a-b) inner and (c - d) outer pendulum.

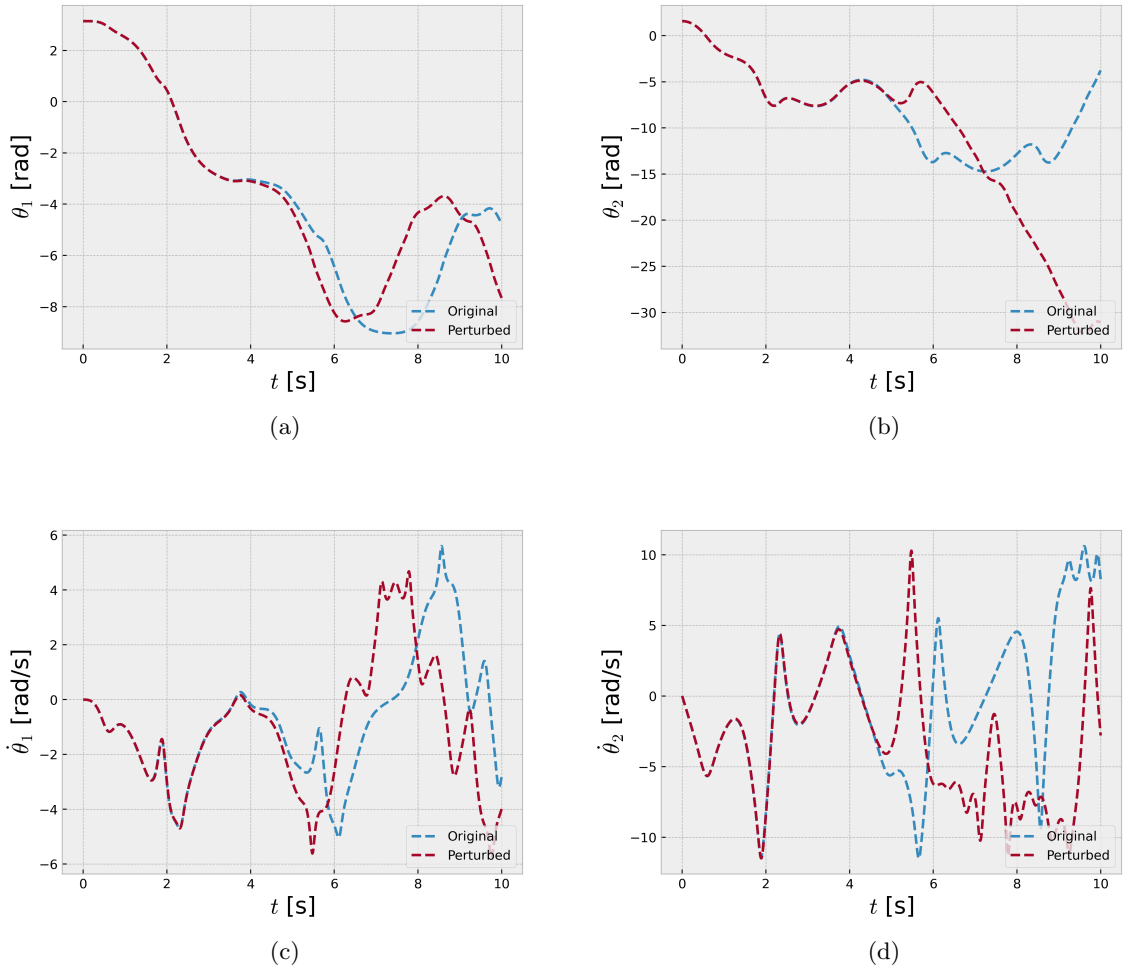


Fig. 4: Time series of a double pendulum (a - b) angles and (c - d) angular velocities with slightly different initial angular velocities for (a - c) inner and (b - d) outer pendulum.

491 The sensitivity to initial conditions is evident as the trajectories diverge signifi-
 492 cantly, despite only a small difference of 0.001 rad/s in the initial angular velocities
 493 (ω_1 and ω_2) of the inner and outer pendulums. This phenomenon is characteristic of
 494 chaotic systems, where minuscule changes in initial conditions can lead to vastly dif-
 495 ferent long-term behaviors, making precise predictions challenging [1]. The time series
 496 exhibit intricate patterns with recurring oscillations, indicative of the underlying non-
 497 linear dynamics. However, the trajectories quickly diverge, showcasing the system's

498 sensitivity to initial conditions, a hallmark of chaos theory [36]. The time series ini-
499 tially following similar paths but rapidly diverging due to the infinitesimal differences
500 in the initial angular velocities, further demonstrating the sensitive dependence on
501 initial conditions in chaotic systems.

502 These results highlight the importance of accurately measuring and accounting for
503 initial conditions in chaotic systems, as even minute uncertainties can amplify over
504 time, leading to substantial deviations in the system’s behavior. The double pendulum
505 serves as a compelling example of the intricate dynamics and unpredictability that
506 can arise in nonlinear systems due to their extreme sensitivity to initial conditions, a
507 fundamental concept in chaos theory [37].

508 To further quantify and statistically validate this divergence, we performed the
509 K-S test on the time series data for various variables, including positions (x, y, θ)
510 and angular velocities (ω_1, ω_2) , before and after the initial perturbation. The K-S test
511 results revealed that for each of these variables, the distributions of the time series
512 data before and after the perturbation were significantly different, with p-values less
513 than 0.05, leading to the rejection of the null hypothesis. This statistically confirms
514 that the slight change in the initial angular velocity has a profound impact on the
515 system’s dynamics, causing the distributions of the position and velocity variables to
516 diverge substantially.

517 This divergence can be attributed to the chaotic nature of the double pendulum
518 system, where the nonlinear equations governing its motion exhibit extreme sensitivity
519 to initial conditions. Even minuscule differences in the starting values can rapidly
520 amplify over time, leading to vastly different trajectories in the phase space [36]. The
521 K-S test results corroborate this fundamental characteristic of chaos, demonstrating
522 that the distributions of the system’s variables become statistically distinct due to
523 the exponential divergence of initially close trajectories, a phenomenon known as the
524 ”butterfly effect” [38].

525 Figure 5a and Fig. 5b show the trajectories of the inner and outer pendulums in
526 the $x - y$ plane, providing a visualization of their motion over time. However, Fig. 5c
527 and Fig. 5d represent the phase space diagrams of the coupled pendulum system. A
528 phase space diagram is a powerful tool for studying dynamical systems, as it provides
529 a geometric representation of the system’s state at any given time [13]. In the case
530 of the coupled pendulum system, the phase space diagrams plot the angular position
531 (θ) of each pendulum against its angular velocity (ω) , capturing the evolution of the
532 system’s state over time.

533 The complex patterns observed in Fig. 5c and Fig. 5d are indicative of the chaotic
534 nature of the coupled pendulum system. The phase space trajectories exhibit intri-
535 cate, aperiodic behavior, never repeating the same pattern or revisiting the same state.
536 The presence of chaos in the coupled pendulum system has significant consequences
537 for its predictability and controllability [36]. Even with precise knowledge of the ini-
538 tial conditions and governing equations, the system’s behavior becomes increasingly
539 difficult to predict over long time scales due to the exponential divergence of nearby
540 trajectories in the phase space.

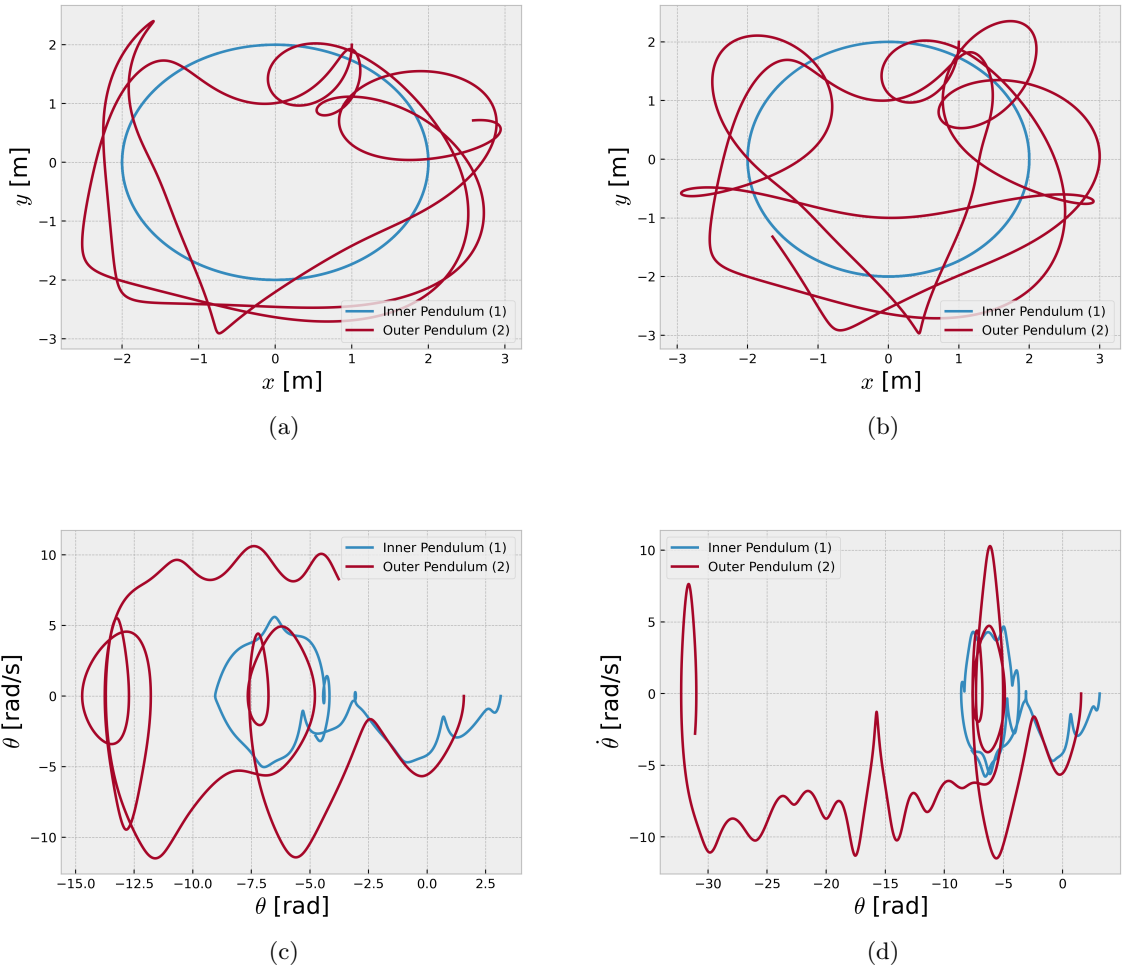


Fig. 5: Coupled inner-outer pendulum dynamics. (a - c) Original system. (b - d) System with 0.001 rad/s perturbation to initial angular velocities. (a - b) x - y trajectories. (c - d) θ vs. ω phase space diagrams for inner (blue) and outer (red) pendulums.

541 Furthermore, chaotic systems often exhibit strange attractors in the phase space,
 542 which are geometrically complex structures that the system's trajectories are con-
 543 fined. The presence of strange attractors in the phase space diagrams of the coupled
 544 pendulum system suggests that the system's dynamics are governed by an underly-
 545 ing deterministic process, despite the apparent randomness and unpredictability of its
 546 behavior [1].

547 The high Shannon entropy values (greater than 1.0) mentioned for the time series
548 data of the coupled pendulum system are consistent with the observed chaotic dynam-
549 ics. For chaotic systems, the Shannon entropy is expected to be high due to the inherent
550 unpredictability and lack of regularity in the system’s behavior [39].

551 4 Conclusion

552 The double pendulum system serves as a compelling example of the intricate dynamics
553 and unpredictability that can arise in nonlinear systems due to their extreme sensi-
554 tivity to initial conditions, a fundamental concept in chaos theory. Through numerical
555 simulations and extensive analysis, we have demonstrated the chaotic behavior of the
556 double pendulum, characterized by the exponential divergence of trajectories, the pres-
557 ence of strange attractors in the phase space, and high Shannon entropy values. These
558 findings underscore the importance of accurately measuring and accounting for ini-
559 tial conditions in chaotic systems, as even minute uncertainties can amplify over time,
560 leading to substantial deviations in the system’s behavior. Furthermore, our compar-
561 ative study of different computing environments (Python, R, GNU Octave, and Julia)
562 has revealed significant differences in their runtime performance and memory usage
563 for the double pendulum simulation task. This information can guide researchers in
564 selecting the most appropriate computing environment based on their specific needs
565 and resource constraints.

566 To further enhance the computational performance of the double pendulum simula-
567 tions, future work could explore the use of accelerated computing techniques. Numba,
568 a JIT compiler for Python [40], can be leveraged to optimize numerical computations
569 by compiling Python code to efficient machine instructions, potentially improving the
570 runtime performance of the Python-based simulations [e. g. 41–43]. Additionally, the
571 utilization of GPU-accelerated computing libraries like CuPy (CUDA for Python) [44]
572 could significantly accelerate the simulations by offloading computationally intensive
573 tasks to the highly parallel architecture of modern graphics processing units (GPUs)
574 [e. g. 45–47]. The massive parallelism provided by GPUs can lead to substantial
575 speedups, especially for large-scale simulations or ensemble runs. While the double
576 pendulum system studied in this work was modeled as a deterministic system, future
577 studies could explore the effects of incorporating stochastic elements. Real-world sys-
578 tems often exhibit random fluctuations or noise, which can significantly impact the
579 system’s dynamics and introduce additional complexity. By incorporating stochastic
580 components into the double pendulum model, researchers could investigate the inter-
581 play between deterministic chaos and random noise, potentially revealing new insights
582 into the behavior of complex dynamical systems.

583 Future investigations could also focus on studying chaotic synchronization, a phe-
584 nomenon where two or more chaotic systems can become synchronized, exhibiting
585 correlated behavior despite their inherent unpredictability. Exploring chaotic synchro-
586 nization in coupled double pendulum systems or the potential applications of such
587 synchronization in various fields, such as secure communication, signal processing,
588 and control systems, could yield valuable insights. As mentioned in the introduction,
589 the double pendulum system serves as a valuable model for studying chaos theory

590 and its applications in understanding Earth’s complex climate systems. Future work
591 could explore the potential use of the double pendulum as a simplified model for
592 investigating the nonlinear dynamics underlying atmospheric-oceanic flows and devel-
593 oping improved long-range climate predictions. By pursuing these future directions,
594 researchers can continue to deepen our understanding of chaotic systems, leverage
595 advanced computational techniques for efficient simulations, and explore the potential
596 applications of chaos theory in various scientific and engineering domains.

597 **Acknowledgments.** We extend our sincere gratitude to Ferio Brahmna (KAIST)
598 and Michael N. Evans (UMD) for their insightful discussions on nonlinear dynam-
599 ics several years ago. These earlier interactions undoubtedly played a significant role
600 in the development of this study. Furthermore, the authors also gratefully acknowl-
601 edge the financial support of the Dean’s Distinguished Fellowship at the University
602 of California, Riverside (UCR) awarded in 2023. We also recognize the support of the
603 ITB Research, Community Services, and Innovation Program (PPMI-ITB) in 2024.
604 The code used for the numerical simulations in this study is available on our GitHub
605 repository: <https://github.com/sandyherho/doublePendulum>.

606 References

- 607 [1] Letellier, C., Abraham, R., Shepelyansky, D.L., Rössler, O.E., Holmes, P., Lozi,
608 R., Glass, L., Pikovsky, A., Olsen, L.F., Tsuda, I., Grebogi, C., Parlitz, U.,
609 Gilmore, R., Pecora, L.M., Carroll, T.L.: Some elements for a history of the
610 dynamical systems theory. *Chaos: An Interdisciplinary Journal of Nonlinear
611 Science* **31**(5) (2021) <https://doi.org/10.1063/5.0047851>
- 612 [2] Atiyah, L.: Dame Mary Cartwright 1900–1997. *The Mathematical Gazette*
613 **82**(495), 494–496 (1998) <https://doi.org/10.1017/S002555720016262X>
- 614 [3] Crease, R.P.: Letting demons do the teaching. *Physics World* **35**(4), 31 (2022)
615 <https://doi.org/10.1088/2058-7058/35/04/20>
- 616 [4] Alexandrov, D.V., Bashkirtseva, I.A., Crucifix, M., Ryashko, L.B.: Nonlinear cli-
617 mate dynamics: From deterministic behaviour to stochastic excitability and chaos.
618 *Physics Reports* **902**, 1–60 (2021) <https://doi.org/10.1016/j.physrep.2020.11.002>
- 619 [5] Khatiwala, S., Shaw, B.E., Cane, M.A.: Enhanced sensitivity of persistent events
620 to weak forcing in dynamical and stochastic systems: Implications for climate
621 change. *Geophysical Research Letters* **28**(13), 2633–2636 (2001) [https://doi.org/
622 10.1029/2000GL012773](https://doi.org/10.1029/2000GL012773)
- 623 [6] Budyansky, M.V., Uleysky, M.Y., Prants, S.V.: Lagrangian coherent structures,
624 transport and chaotic mixing in simple kinematic ocean models. *Communications
625 in Nonlinear Science and Numerical Simulation* **12**(1), 31–44 (2007) [https://doi.
626 org/10.1016/j.cnsns.2006.01.008](https://doi.org/10.1016/j.cnsns.2006.01.008)

- 627 [7] Cropp, R., Moroz, I.M., Norbury, J.: Chaotic dynamics in a simple dynamical
628 green ocean plankton model. *Journal of Marine Systems* **139**, 483–495 (2014)
629 <https://doi.org/10.1016/j.jmarsys.2014.08.002>
- 630 [8] Tantet, A., Lucarini, V., Lunkeit, F., Dijkstra, H.A.: Crisis of the chaotic attractor
631 of a climate model: a transfer operator approach. *Nonlinearity* **31**(5), 2221 (2018)
632 <https://doi.org/10.1088/1361-6544/aaaf42>
- 633 [9] Shen, B.-W., Pielke Sr, R.A., Zeng, X., Baik, J.-J., Faghieh-Naini, S., Cui, J.,
634 Atlas, R.: Is weather chaotic?: Coexistence of chaos and order within a generalized
635 Lorenz model. *Bulletin of the American Meteorological Society* **102**(1), 148–158
636 (2021) <https://doi.org/10.1175/BAMS-D-19-0165.1>
- 637 [10] Ghil, M., Lucarini, V.: The physics of climate variability and climate change.
638 *Reviews of Modern Physics* **92**(3), 035002 (2020) [https://doi.org/10.1103/](https://doi.org/10.1103/RevModPhys.92.035002)
639 [RevModPhys.92.035002](https://doi.org/10.1103/RevModPhys.92.035002)
- 640 [11] Crutchfield, J.P.: Between order and chaos. *Nature Physics* **8**(1), 17–24 (2012)
641 <https://doi.org/10.1038/nphys2190>
- 642 [12] Rindler, F.: *Calculus of Variations* vol. 5. Springer, Berlin (2018)
- 643 [13] Higham, N.J., Dennis, M.R., Glendinning, P., Martin, P.A., Santosa, F., Tanner,
644 J.: *Princeton Companion to Applied Mathematics*. Princeton University Press,
645 Princeton (2015)
- 646 [14] Calvão, A.M., Penna, T.P.: The double pendulum: a numerical study. *European*
647 *Journal of Physics* **36**(4), 045018 (2015) [https://doi.org/10.1088/0143-0807/36/](https://doi.org/10.1088/0143-0807/36/4/045018)
648 [4/045018](https://doi.org/10.1088/0143-0807/36/4/045018)
- 649 [15] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau,
650 D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt,
651 S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones,
652 E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., Vander-
653 Plas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A.,
654 Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P.,
655 SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Com-
656 puting in Python. *Nature Methods* **17**, 261–272 (2020) [https://doi.org/10.1038/](https://doi.org/10.1038/s41592-019-0686-2)
657 [s41592-019-0686-2](https://doi.org/10.1038/s41592-019-0686-2)
- 658 [16] Soetaert, K., Petzoldt, T., Setzer, R.W.: Solving differential equations in R: pack-
659 age deSolve. *Journal of Statistical Software* **33**, 1–25 (2010) [https://doi.org/10.](https://doi.org/10.18637/jss.v033.i09)
660 [18637/jss.v033.i09](https://doi.org/10.18637/jss.v033.i09)
- 661 [17] Rackauckas, C., Nie, Q.: *Differentialequations.jl*—A Performant and Feature-rich
662 Ecosystem for Solving Differential Equations in Julia. *Journal of Open Research*
663 *Software* **5**(1), 15–15 (2017) <https://doi.org/10.5334/jors.151>

- 664 [18] Eaton, J.W.: GNU Octave and reproducible research. *Journal of Process Control*
665 **22**(8), 1433–1438 (2012) <https://doi.org/10.1016/j.jprocont.2012.04.006>
- 666 [19] Herho, S.H.S.: Tutorial Pemrograman Python 2 Untuk Pemula. WCPL ITB,
667 Bandung (2017)
- 668 [20] Herho, S.H.S., Syahputra, M.R., Trilaksono, N.J.: Pengantar Metode Numerik
669 Terapan: Menggunakan Python. WCPL ITB, Bandung (2024)
- 670 [21] Kong, L.: Epidemic modeling using differential equations with implementation in
671 R. *International Journal of Mathematical Education in Science and Technology*
672 **55**(2), 480–491 (2024) <https://doi.org/10.1080/0020739X.2023.2249902>
- 673 [22] Bezanson, J., Chen, J., Chung, B., Karpinski, S., Shah, V.B., Vitek, J.,
674 Zoubitzky, L.: Julia: Dynamism and performance reconciled by design. *Proceedings of the ACM on Programming Languages* **2**(OOPSLA), 1–23 (2018)
675 <https://doi.org/10.1145/3276490>
- 677 [23] Gao, K., Mei, G., Piccialli, F., Cuomo, S., Tu, J., Huo, Z.: Julia language in
678 machine learning: Algorithms, applications, and open issues. *Computer Science*
679 *Review* **37**, 100254 (2020) <https://doi.org/10.1016/j.cosrev.2020.100254>
- 680 [24] Wang, D., Ueda, Y., Kula, R.G., Ishio, T., Matsumoto, K.: Can we benchmark
681 code review studies? a systematic mapping study of methodology, dataset, and
682 metric. *Journal of Systems and Software* **180**, 111009 (2021) <https://doi.org/10.1016/j.jss.2021.111009>
- 684 [25] Kruskal, W.H., Wallis, W.A.: Use of Ranks in One-Criterion Variance Analysis.
685 *Journal of the American Statistical Association* **47**(260), 583–621 (1952) <https://doi.org/10.1080/01621459.1952.10483441>
- 686
- 687 [26] Dunn, O.J.: Multiple comparisons using rank sums. *Technometrics* **6**(3), 241–252
688 (1964) <https://doi.org/10.1080/00401706.1964.10490181>
- 689 [27] Das, B., Murganekar, D., Navyashree, S., Kumar, P.: Novel combination artificial
690 neural network models could not outperform individual models for weather-based
691 cashew yield prediction. *International Journal of Biometeorology* **66**(8), 1627–
692 1638 (2022) <https://doi.org/10.1007/s00484-022-02306-1>
- 693 [28] Borko, Š., Premate, E., Zigmajster, M., Fišer, C.: Determinants of range sizes
694 pinpoint vulnerability of groundwater species to climate change: A case study on
695 subterranean amphipods from the Dinarides. *Aquatic Conservation: Marine and*
696 *Freshwater Ecosystems* **33**(6), 629–636 (2023) <https://doi.org/10.1002/aqc.3941>
- 697 [29] Herho, S.H.S., Anwar, I.P., Herho, K.E.P., Dharma, C.S., Irawan, D.E.: Numerical
698 simulation of the 2D trajectory of a non-buoyant fluid parcel under the influence
699 of inertial oscillation. *EarthArXiv* (2024) <https://doi.org/10.31223/X5GT35>

- 700 [30] Terpilowski, M.A.: scikit-posthocs: Pairwise multiple comparison tests in Python.
701 J. Open Source Softw. **4**(36), 1169 (2019) <https://doi.org/10.21105/joss.01169>
- 702 [31] Shannon, C.E.: A mathematical theory of communication. The Bell System Tech-
703 nical Journal **27**(3), 379–423 (1948) [https://doi.org/10.1002/j.1538-7305.1948.
704 tb01338.x](https://doi.org/10.1002/j.1538-7305.1948.tb01338.x)
- 705 [32] Massey Jr, F.J.: The Kolmogorov-Smirnov test for goodness of fit. Journal of the
706 American Statistical Association **46**(253), 68–78 (1951) [https://doi.org/10.1080/
707 01621459.1951.10500769](https://doi.org/10.1080/01621459.1951.10500769)
- 708 [33] Ghatak, G., Mohanty, H., Rahman, A.U.: Kolmogorov–Smirnov test-based
709 actively-adaptive Thompson sampling for non-stationary bandits. IEEE Transac-
710 tions on Artificial Intelligence **3**(1), 11–19 (2021) [https://doi.org/10.1109/TAI.
711 2021.3121653](https://doi.org/10.1109/TAI.2021.3121653)
- 712 [34] Kini, K.R., Harrou, F., Madakyaru, M., Sun, Y.: Enhanced Data-Driven Mon-
713 itoring of Wastewater Treatment Plants using the Kolmogorov-Smirnov Test.
714 Environmental Science: Water Research & Technology (2024) [https://doi.org/10.
715 1039/D3EW00829K](https://doi.org/10.1039/D3EW00829K)
- 716 [35] Sharma, V., Biswas, R.: Statistical analysis of seismic b-value using
717 non-parametric Kolmogorov–Smirnov test and probabilistic seismic hazard
718 parametrization for Nepal and its surrounding regions. Natural Hazards, 1–28
719 (2024) <https://doi.org/10.1007/s11069-024-06531-2>
- 720 [36] Tél, T., Gruiz, M.: Chaotic Dynamics: an Introduction Based on Classical
721 Mechanics. Cambridge University Press, Cambridge (2006)
- 722 [37] Hilborn, R.C.: Chaos and Nonlinear Dynamics: an Introduction for Scientists and
723 Engineers. Oxford University Press, Oxford (2000)
- 724 [38] Lorenz, E.N.: Designing Chaotic Models. Journal of the Atmospheric Sciences
725 **62**(5), 1574–1587 (2005) <https://doi.org/10.1175/JAS3430.1>
- 726 [39] Contreras-Reyes, J.E.: Mutual information matrix based on asymmetric Shannon
727 entropy for nonlinear interactions of time series. Nonlinear Dynamics **104**(4),
728 3913–3924 (2021) <https://doi.org/10.1007/s11071-021-06498-w>
- 729 [40] Lam, S.K., Pitrou, A., Seibert, S.: Numba: A LLVM-based Python JIT compiler.
730 In: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure
731 in HPC, pp. 1–6 (2015). <https://doi.org/10.1145/2833157.2833162>
- 732 [41] Bartman, P., Banaśkiewicz, J., Drenda, S., Manna, M., Olesik, M.A., Rozwoda, P.,
733 Sadowski, M., Arabas, S.: PyMPDATA v1: Numba-accelerated implementation
734 of MPDATA with examples in Python, Julia and Matlab. Journal of Open Source
735 Software **7**(77), 3896 (2022) <https://doi.org/10.21105/joss.03896>

- 736 [42] Clemente-López, D., Muñoz-Pacheco, J.M., Jesus, J.R.-M.: Experimental vali-
737 dation of IoT image encryption scheme based on a 5-D fractional hyperchaotic
738 system and Numba JIT compiler. *Internet of Things*, 101116 (2024) <https://doi.org/10.1016/j.iot.2024.101116>
739
- 740 [43] Herho, S.H.S., Kaban, S.N., Irawan, D.E., Kapid, R.: Efficient 1D Heat Equation
741 Solver: Leveraging Numba in Python. *EKSAKTA: Berkala Ilmiah Bidang MIPA*
742 **25**(02), 126–137 (2024) <https://doi.org/10.24036/eksakta/vol25-iss02/487>
- 743 [44] Okuta, R., Unno, Y., Nishino, D., Hido, S., Loomis, C.: CuPy: A NumPy-
744 Compatible Library for NVIDIA GPU Calculations. In: *Proceedings of Workshop*
745 *on Machine Learning Systems (LearningSys) in The 31st Annual Conference on*
746 *Neural Information Processing Systems (NIPS)* (2017). http://learningsys.org/nips17/assets/papers/paper_16.pdf
747
- 748 [45] Mathias, S., Coulier, A., Hellander, A.: CBMOS: a GPU-enabled Python frame-
749 work for the numerical study of center-based models. *BMC Bioinformatics* **23**(1),
750 55 (2022) <https://doi.org/10.1186/s12859-022-04575-4>
- 751 [46] Martin, D.S., Torres, C.: 2D Simplified Wildfire Spreading Model in Python: From
752 NumPy to CuPy. *CLEI electronic journal* **26**(1), 5–1 (2023) <https://doi.org/10.19153/cleiej.26.1.5>
753
- 754 [47] Askar, T., Yergaliyev, A., Shukirgaliyev, B., Abdikamalov, E.: Exploring Numba
755 and CuPy for GPU-Accelerated Monte Carlo Radiation Transport. *Computation*
756 **12**(3), 61 (2024) <https://doi.org/10.3390/computation12030061>