Open Access Library Journal
2021, Volume 8, e7618
ISSN Online: 2333-9721
ISSN Print: 2333-9705

Numerical and Chaotic Analysis of Proposed SIR Model

Maysoon M. Aziz[1], Arqam S. Mahmood[2]

[1]Department of Mathematics, College of Computer Sciences and Mathematics, University of Mosul, Mosul, Iraq
[2]Directorate General of Education Nineveh Ministry of Education, Iraq
Email: aziz_maysoon@yahoo.com, asmd.2013mosl@gmail.com

How to cite this paper: Aziz, M.M. and Mahmood, A.S. (2021) Numerical and Chaotic Analysis of Proposed SIR Model. *Open Access Library Journal*, **8**: e7618.
https://doi.org/10.4236/oalib.1107618

Received: June 7, 2021
Accepted: August 3, 2021
Published: August 6, 2021

Abstract

In this paper, we use the classic mathematical model SIR with the three differential equations as a non-linear system and combine it with the Runge-Kutta numerical method of the fourth order and the sixth and seventh order of the same method to generate simulated data in each of the mentioned ranks (for susceptible people, Infected and recovered from the disease) for the epidemic disease COVID-19 by giving the initial values (initial conditions) for the population in a certain country of the world, and we chose this country that is Iraq. Through this work the difference between the results for the three methods (4^{th}, 6^{th}, and 7^{th} order) was observed in terms of the error value, the time taken for each step and the total time to implement the solution in each rank, and this has been clarified in a table showing the comparison between the results for each rank for the numerical method. The binary test (0-1) was also used to study the chaotic behavior of the disease. The simulation data for the number of infected to solve in each rank was used to show the chaos of the dynamic system, and all methods of solution led to the results that the behavior of the disease is chaotic, the value of ($Kc \cong 1$) and we explained that With a table showing the Kc values for the disease in each rank, also we used the Matlab system to write the important programs to obtain all the results and graphics required in this work.

Subject Areas

Dynamical Systems

Keywords

SIR Model, (R-K) Numerical Method, Chaotic Analyzing

1. Introduction

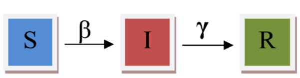
The continuous dynamical system is very rumor in several of applied sciences and engineering also computational mathematics [1]. in this paper, we will build a continuous dynamic system consisting of the classic three-dimensional SIR model, which is a model for studying and analyzing epidemic diseases [2] [3] [4] [5] and the known numerical method (Runge-Kutta) [5] [6], with the fourth order [6], sixth order [7] [8] and the seventh order [9], where the model SIR is used with the 4th order, 6th order of the method (Runge-Kutta) Once and with the 7th rank of the same method again to generate simulated data (hypothetical) by taking initial values (initial conditions) from the real data of the daily statistics of the epidemic disease (COVID-19) for a specific country of the world and using those resulting data for each rank in the chaos test of the outbreak disease By using the binary test method (0-1), which is one of the chaotic testing methods for dynamic systems, which has the advantage that it depends mainly on observations of time series data, and the test result is either close to zero (which is the regularity state) or close to one (which is the chaotic state of the system) [10]-[16], The disease dynamic lamentation has been shown to be chaotic (Kcorr ≅ 1). All figures and drawings showing the behavior of the chaotic system have been included. Also, programs have been built in Matlab system to apply to all the aforementioned operations (from finding simulation data, testing chaos, etc.). A program is also build to indicate which of the three ranks is better to study the dynamic system in terms of finding the step size, the maximum error, the time limit for each step, and the total time it takes to solve (in seconds). The results of this mathematical process have been shown in a table. At the end, a summary is listed containing all work results.

2. SIR Model

It's an epidemiologic mathematical model that does computing the theoretic numbers of individuals infected with a contagious disease in a closed population over times. This classic model his name is derived from the fact that they involve coupled equations relating the number of (susceptible (S), infected (I) and recovered (R)). SIR model is one of the simplest models that developed by (Kermack-Mckendrick) in (1927).

Description of SIR model:

$$\left.\begin{aligned} \dot{S} &= \frac{\mathrm{d}S}{\mathrm{d}t} = -\beta SI \\ \dot{I} &= \frac{\mathrm{d}I}{\mathrm{d}t} = \beta SI - \gamma I \\ \dot{R} &= \frac{\mathrm{d}R}{\mathrm{d}t} = \gamma I \end{aligned}\right\} \text{ non-linear System} \quad (1)$$



$$N = S + I + R, \quad \dot{S} + \dot{I} + \dot{R} = 0$$

where:

N: The total number of population.

$S(t)$: The number of Susceptible people at time (t).

$I(t)$: The number of Infected people at time (t).

$R(t)$: The number of Recovered people at time (t).

$$\left.\begin{array}{l}(\beta = 0.0845): \text{is a transmission rate} \\ (\gamma = 0.07): \text{is recovered rate}\end{array}\right\} \text{are constant parameters of the model in Iraq}$$

where: $\beta = \mu + \gamma$, $\gamma = 1/D$, (μ = 0.0145): is mortality rate in day, D = 14: is duration of disease time.

3. Basic Reproduction Number (R_0)

Definition(R_0): Represent to the average number of new infections generated by each infected person, the high value of (R_0) means easy to transmit the disease, and the low value of (R_0) means difficult to transmit the disease. (R_0) is called threshold of disease, (the value of (R_0) assumes that no pre-existing immunity, *i.e.* it mean everyone is susceptible), where Ro = β/γ.

Lemma: If $R_0 > 1$ then $I(t)$ is increasing and the disease is epidemic, and if $R_0 < 1$ then $I(t)$ is decreasing and the disease is endemic, It is assumed in the absence of a vaccine, the entire population will be susceptible to infection, meaning that $S \cong N$, so we divide β by N.

Proof: From the SIR model we have: $\mathrm{d}I/\mathrm{d}t = (\beta/N) * SI - \gamma I \rightarrow$
$\mathrm{d}I/\mathrm{d}t = (\beta - \gamma)I \rightarrow \mathrm{d}I/I = (\beta - \gamma)\mathrm{d}t$ by integral of two hand sides:
$\ln I = (\beta I - \gamma I) + C \rightarrow I(t) = \mathrm{e}^{\beta T(t) - \gamma I(t)} \cdot \mathrm{e}^C$, when $t = 0 \rightarrow I(0) = \mathrm{e}^C$
then $I(t) = \mathrm{e}^{\beta T(t) - \gamma I(t)} \cdot I(0) > 0$ when $\mathrm{d}I/\mathrm{d}t > 0$ then $\beta I - \gamma I > 0$ $\beta I > \gamma I$
$\rightarrow \beta/\gamma > 1 \rightarrow R_0 > 1$.

4. The Numerical Method "Runge-Kutta"

To solve the differential equations, we will use the following Relationships:

$$\mathrm{d}x_1/\mathrm{d}t = g_1(t, x_1, x_2, \cdots, x_n)$$
$$\mathrm{d}x_2/\mathrm{d}t = g_2(t, x_1, x_2, \cdots, x_n)$$
$$\vdots$$
$$\mathrm{d}x_n/\mathrm{d}t = g_n(t, x_1, x_2, \cdots, x_n)$$

For more simply we write: $\mathrm{d}y/\mathrm{d}t = g(t, x)$

Where (x) is a vector for n-dimensions, this is not an independent equation. If we replace the right-hand side by $g(x)$ we will get on an independent equation.

If the function (f) in the R.H. side is nonlinear, we will need numerical methods. The Runge-Kutta methods have the same precisions in solution as the Taylor expansions in any order, but there is no need for derivatives. We compute (x) at $X_{ti+1} = X_{ti} + h$.

4.1. Runge-Kutta Method with 4th Order

The 4th order Runge-Kutta numerical method is one of the classic numerical methods used to solve the ordinary differential equations of nonlinear and time-related continuous dynamic systems with a number of iterations to obtain the best approximate value.

Description Method:

$$y_{n+1} = y_n + h * (k1 + 2k2 + 2k3 + k4)/6.$$

where: $\Delta t = h = t_{n+1} - t_n$ and:

$k1 = f(t_n, y_n)$;

$k2 = f(t_n + h/2, y_n + h * k1/2)$;

$k3 = f(t_n + h/2, y_n + h * k2/2)$;

$k4 = f(t_n + h, y_n + h * k3)$.

From system (1) we have: dS/dt = f1, dI/dt = f2, dR/dt = f3, (So, Io, Ro) = (3×10^6, 100, 0) initial condition, to = 1,

K1 = h * f1(to, So, Io, Ro),

L1 = h * f2(to, So, Io, Ro),

M1 = h * f3(to, So, Io, Ro);

K2 = h * f1(to + h/2, So + K1/2, Io + L1/2, Ro + M1/2),

L2 = h * f2 (to + h/2, So + K1/2, Io + L1/2, Ro + M1/2),

M2 = h * f3(to + h/2, So + K1/2, Io + L1/2, Ro + M1/2);

K3 = h * f1(to + h/2, So + K2/2, Io + L2/2, Ro + M2/2),

L3 = h * f2(to + h/2, So + K2/2, Io + L2/2, Ro + M2/2),

M3 = h * f3(to + h/2, So + K2/2, Io + L2/2, Ro + M2/2);

K4 = h * f1(to + h, So + K3, Io + L3, Ro + M3),

L4 = h * f2(to + h, So + K3, Io + L3, Ro + M3),

M4 = h * f3(to + h, So + K3, Io + L3, Ro + M3);

$S_1 = So + h/6 \times (K1 + 2K2 + 2K3 + K4)$,

$I_1 = Io + h/6 \times (L1 + 2L2 + 2L3 + L4)$,

$R_1 = Ro + h/6 \times (M1 + 2M2 + 2M3 + M4)$.

The process is repeated with (n) iterations by using Matlab program.

4.2. The Numerical Method "Runge-Kutta" of 6th Order

k1 = h * f(t(i); x(i), y(i), z(i)),

l1 = h * g(t(i), x(i), y(i), z(i)),

m1 = h * p(t(i), x(i), y(i), z(i));

k2 = h * f(x(i) + k1/3, y(i) + l1/3, z(i) + m1/3),

l2 = h * g(x(i) + k1/3, y(i) + l1/3, z(i) + m1/3),

m2 = h * p(x(i) + k1/3, y(i) + l1/3, z(i) + m1/3);

k3 = h * f(x(i) + 2 * k2/3, y(i) + 2 * l2/3, z(i) + 2 * m2/3),

l3 = h * g(x(i) + 2 * k2/3, y(i) + 2 * l2/3, z(i) + 2 * m2/3),

m3 = h * p(x(i) + 2 * k2/3, y(i) + 2 * l2/3, z(i) + 2 * m2/3);

k4 = h * f(x(i) + k1/12 + k2/3 − k3/12, y(i) + l1/12 + l2/3 − l3/12, z(i) + m1/12 + m2/3 − m3/12),

l4 = h * g(x(i) + k1/12 + k2/3 − k3/12, y(i) + l1/12 + l2/3 − l3/12, z(i) + m1/12 + m2/3 − m3/12),

m4 = h * p(x(i) + k1/12 + k2/3 − k3/12, y(i) + l1/12 + l2/3 − l3/12, z(i) +

m1/12 + m2/3 − m3/12) k5 = h * f(x(i) + 25 * k1/48 − 55 * k2/24 + 35 * k3/48 + 15 * k4/8, y(i) + 25 * l1/48 − 55 * l2/24 + 35 * l3/48 + 15 * l4/8, z(i) + 25 * m1/48 − 55 * m2/24 + 35 * m3/48 + 15 * m4/8);

l5 = h * g(x(i) + 25 * k1/48 − 55 * k2/24 + 35 * k3/48 + 15 * k4/8, y(i) + 25 * l1/48 − 55 * l2/24 + 35 * l3/48 + 15 * l4/8, z(i) + 25 * m1/48 − 55 * m2/24 + 35 * m3/48 + 15 * m4/8),

m5 = h * p(x(i) + 25 * k1/48 − 55 * k2/24 + 35 * k3/48 + 15 * k4/8, y(i) + 25 * l1/48 − 55 * l2/24 + 35 * l3/48 + 15 * l4/8, z(i) + 25 * m1/48 − 55 * m2/24 + 35 * m3/48 + 15 * m4/8);

k6 = h * f(x(i) + 3 * k1/20 − 11 * k2/20 − k3/8 + k4/2 + k5/10, y(i) + 3 * l1/20 − 11 * l2/20 − l3/8 + l4/2 + l5/10, z(i) + 3 * m1/20 − 11 * m2/20 − m3/8 + m4/2 + m5/10),

l6 = h * g(x(i) + 3 * k1/20 − 11 * k2/20 − k3/8 + k4/2 + k5/10, y(i) + 3 * l1/20 − 11 * l2/20 − l3/8 + l4/2 + l5/10, z(i) + 3 * m1/20 − 11 * m2/20 − m3/8 + m4/2 + m5/10),

m6 = h * p(x(i) + 3 * k1/20 − 11 * k2/20 − k3/8 + k4/2 + k5/10, y(i) + 3 * l1/20 − 11 * l2/20 − l3/8 + l4/2 + l5/10, z(i) + 3 * m1/20 − 11 * m2/20 − m3/8 + m4/2 + m5/10);

k7 = h * f(x(i) − 261 * k1/260 + 33 * k2/13 + 43 * k3/156 − 118 * k4/39 + 32 * k5/195 + 80 * k6/39, y(i) − 261 * l1/260 + 33 * l2/13 + 43 * l3/156 − 118 * l4/39 + 32 * l5/195 + 80 * l6/39, z(i) − 261 * m1/260 + 33 * m2/13 + 43 * m3/156 − 118 * m4/39 + 32 * m5/195 + 80 * m6/39),

l7 = h * g(x(i) − 261 * k1/260 + 33 * k2/13 + 43 * k3/156 − 118 * k4/39 + 32 * k5/195 + 80 * k6/39, y(i) − 261 * l1/260 + 33 * l2/13 + 43 * l3/156 − 118 * l4/39 + 32 * l5/195 + 80 * l6/39, z(i) − 261 * m1/260 + 33 * m2/13 + 43 * m3/156 − 118 * m4/39 + 32 * m5/195 + 80 * m6/39),

m7 = h * p(x(i) − 261 * k1/260 + 33 * k2/13 + 43 * k3/156 − 118 * k4/39 + 32 * k5/195 + 80 * k6/39, y(i) − 261 * l1/260 + 33 * l2/13 + 43 * l3/156 − 118 * l4/39 + 32 * l5/195 + 80 * l6/39, z(i) − 261 * m1/260 + 33 * m2/13 + 43 * m3/156 − 118 * m4/39 + 32 * m5/195 + 80 * m6/39);

x(i + 1) = x(i) + h * (13 * k1 + 55 * k3 + 55 * k4 + 32 * k5 + 32 * k6 + 13 * k7)/200,

y(i + 1) = y(i) + h * (13 * l1 + 55 * l3 + 55 * l4 + 32 * l5 + 32 * l6 + 13 * l7)/200,

z(i + 1) = z(i) + h * (13 * m1 + 55 * m3 + 55 * m4 + 32 * m5 + 32 * m6 + 13 * m7)/200.

4.3. The Numerical Method "Runge-Kutta" of 7th Order

k1 = h * f(x(i), y(i), z(i)),

l1 = h * g(x(i), y(i), z(i)),

m1 = h * p(x(i), y(i), z(i));

k2 = h * f(t(i) + h/18, x(i) + h/18 * k1, y(i) + h/18 * l1),

l2 = h * g(t(i) + h/18, x(i) + h/18 * k1, y(i) + h/18 * l1),

m2 = h * p(t(i) + h/18, x(i) + h/18 * k1, y(i) + h/18 * l1);

k3 = h * f(t(i) + (3 * h)/36, x(i) + 1/60 * (4 * k1 + k2), y(i) + 1/60 * (4 * l1 + l2)),

l3 = h * g(t(i) + (3 * h)/36, x(i) + 1/60 * (4 * k1 + k2), y(i) + 1/60 * (4 * l1 + l2)),

m3 = h * p(t(i) + (3 * h)/36, x(i) + 1/60 * (4 * k1 + k2), y(i) + 1/60 * (4 * l1 + l2));

k4 = h * f(t(i) + (4 * h)/36, x(i) + 1/180 * (−181 * k1 + 171 * k2 + 130 * k3), y(i) + 1/180 * (−181 * l1 + 171 * l2 + 130 * l3)),

l4 = h * g(t(i) + (4 * h)/36, x(i) + 1/180 * (−181 * k1 + 171 * k2 + 130 * k3), y(i) + 1/180 * (−181 * l1 + 171 * l2 + 130 * l3)),

m4 = h * p(t(i) + (4 * h)/36, x(i) + 1/180 * (−181 * k1 + 171 * k2 + 130 * k3), y(i) + 1/180 * (−181 * l1 + 171 * l2 + 130 * l3));

k5 = h * f(t(i) + (5 * h)/36, x(i) + 1/180 * (−902 * k1 + 293 * k2 − 2040 * k3 + 30 * k4), y(i) + 1/180 * (−902 * l1 + 293 * l2 − 2040 * l3 + 30 * l4)),

l5 = h * g(t(i) + (5 * h)/36, x(i) + 1/180 * (−902 * k1 + 293 * k2 − 2040 * k3 + 30 * k4), y(i) + 1/180 * (−902 * l1 + 293 * l2 − 2040 * l3 + 30 * l4)),

m5 = h * p(t(i) + (5 * h)/36, x(i) + 1/180 * (−902 * k1 + 293 * k2 − 2040 * k3 + 30 * k4), y(i) + 1/180 * (−902 * l1 + 293 * l2 − 2040 * l3 + 30 * l4));

k6 = h * f(t(i) + h/6, x(i) + 1/24 * (−15 * k1 + 48 * k2 + 31 * k3 + k4 + k5), y(i) + 1/24 * (−15 * l1 + 48 * l2 + 31 * l3 + l4 + l5)),

l6 = h * g(t(i) + h/6, x(i) + 1/24 * (−15 * k1 + 48 * k2 + 31 * k3 + k4 + k5), y(i) + 1/24 * (−15 * l1 + 48 * l2 + 31 * l3 + l4 + l5)),

m6 = h * p(t(i) + h/6, x(i) + 1/24 * (−15 * k1 + 48 * k2 + 31 * k3 + k4 + k5), y(i) + 1/24 * (−15 * l1 + 48 * l2 + 31 * l3 + l4 + l5));

k7 = h * f(t(i) + (2 * h)/6, x(i) + 1/30 * (17 * k1 − 48 * k2 + 31 * k3 − k4 − k5 + 12 * k6), y(i) + 1/30 * (17 * l1 − 48 * l2 + 31 * l3 − l4 − l5 + 12 * l6)),

l7 = h * g(t(i) + (2 * h)/6, x(i) + 1/30 * (17 * k1 − 48 * k2 + 31 * k3 − k4 − k5 + 12 * k6), y(i) + 1/30 * (17 * l1 − 48 * l2 + 31 * l3 − l4 − l5 + 12 * l6)),

m7 = h * p(t(i) + (2 * h)/6, x(i) + 1/30 * (17 * k1 − 48 * k2 + 31 * k3 − k4 − k5 + 12 * k6), y(i) + 1/30 * (17 * l1 − 48 * l2 + 31 * l3 − l4 − l5 + 12 * l6));

k8 = h * f(t(i) + (3 * h)/6, x(i) + 1/80 * (192 * k1 − 528 * k2 + 341 * k3 − 11 * k4 − 11 * k5 + 32 * k6 + 25 * k7), y(i) + 1/80 * (192 * l1 − 528 * l2 + 341 * l3 − 11 * l4 − 11 * l5 + 32 * l6 + 25 * l7)),

l8 = h * g(t(i) + (3 * h)/6, x(i) + 1/80 * (192 * k1 − 528 * k2 + 341 * k3 − 11 * k4 − 11 * k5 + 32 * k6 + 25 * k7), y(i) + 1/80 * (192 * l1 − 528 * l2 + 341 * l3 − 11 * l4 − 11 * l5 + 32 * l6 + 25 * l7)),

m8 = h * p(t(i) + (3 * h)/6, x(i) + 1/80 * (192 * k1 − 528 * k2 + 341 * k3 − 11 * k4 − 11 * k5 + 32 * k6 + 25 * k7), y(i) + 1/80 * (192 * l1 − 528 * l2 + 341 * l3 − 11 * l4 − 11 * l5 + 32 * l6 + 25 * l7));

k9 = h * f(t(i) + (4 * h)/6, x(i) + 1/66 * (54 * k1 − 144 * k2 + 93 * k3 − 3 * k4 − 3 * k5 + 32 * k6 − 17 * k7 + 32 * k8), y(i) + 1/66 * (54 * l1 − 144 * l2 + 93 * l3 − 3 * l4 − 3 * l5 + 32 * l6 − 17 * l7 + 32 * l8)),

l9 = h * g(t(i) + (4 * h)/6, x(i) + 1/66 * (54 * k1 − 144 * k2 + 93 * k3 − 3 * k4 − 3 * k5 + 32 * k6 − 17 * k7 + 32 * k8), y(i) + 1/66 * (54 * l1 − 144 * l2 + 93 * l3 − 3 * l4 − 3 * l5 + 32 * l6 − 17 * l7 + 32 * l8)),

m9 = h * p(t(i) + (4 * h)/6, x(i) + 1/66 * (54 * k1 − 144 * k2 + 93 * k3 − 3 * k4 − 3 * k5 + 32 * k6 − 17 * k7 + 32 * k8), y(i) + 1/66 * (54 * l1 − 144 * l2 + 93 * l3 − 3 * l4 − 3 * l5 + 32 * l6 − 17 * l7 + 32 * l8));

k10 = h * f(t(i) + (5 * h)/6, x(i) + 1/3960 * (−22876 * k1 + 64464 * k2 − 41633 * k3 + 1343 * k4 + 1343 * k5 − 656 * k6 − 460 * k7 − 40 * k8 + 1815 * k9), y(i) + 1/3960 * (−22876 * l1 + 64464 * l2 − 41633 * l3 + 1343 * l4 + 1343 * l5 − 656 * l6 − 460 * l7 − 40 * l8 + 1815 * l9)),

l10 = h * g(t(i) + (5 * h)/6, x(i) + 1/3960 * (−22876 * k1 + 64464 * k2 − 41633 * k3 + 1343 * k4 + 1343 * k5 − 656 * k6 − 460 * k7 − 40 * k8 + 1815 * k9), y(i) + 1/3960 * (−22876 * l1 + 64464 * l2 − 41633 * l3 + 1343 * l4 + 1343 * l5 − 656 * l6 − 460 * l7 − 40 * l8 + 1815 * l9)),

m10 = h * p(t(i) + (5 * h)/6, x(i) + 1/3960 * (−22876 * k1 + 64464 * k2 − 41633 * k3 + 1343 * k4 + 1343 * k5 − 656 * k6 − 460 * k7 − 40 * k8 + 1815 * k9), y(i) + 1/3960 * (−22876 * l1 + 64464 * l2 − 41633 * l3 + 1343 * l4 + 1343 * l5 − 656 * l6 − 460 * l7 − 40 * l8 + 1815 * l9));

k11 = h * f(t(i) + h, x(i) + 1/902 * (16139 * k1 − 45120 * k2 + 29140 * k3 − 940 * k4 − 940 * k5 + 1828 * k6 − 769 * k7 + 2752 * k8 − 1980 * k9 + 792 * k10), y(i) + 1/902 * (16139 * l1 − 45120 * l2 + 29140 * l3 − 940 * l4 − 940 * l5 + 1828 * l6 − 769 * l7 + 2752 * l8 − 1980 * l9 + 792 * l10)),

l11 = h * g(t(i) + h, x(i) + 1/902 * (16139 * k1 − 45120 * k2 + 29140 * k3 − 940 * k4 − 940 * k5 + 1828 * k6 − 769 * k7 + 2752 * k8 − 1980 * k9 + 792 * k10), y(i) + 1/902 * (16139 * l1 − 45120 * l2 + 29140 * l3 − 940 * l4 − 940 * l5 + 1828 * l6 − 769 * l7 + 2752 * l8 − 1980 * l9 + 792 * l10)),

m11 = h * p(t(i) + h, x(i) + 1/902 * (16139 * k1 − 45120 * k2 + 29140 * k3 − 940 * k4 − 940 * k5 + 1828 * k6 − 769 * k7 + 2752 * k8 − 1980 * k9 + 792 * k10), y(i) + 1/902 * (16139 * l1 − 45120 * l2 + 29140 * l3 − 940 * l4 − 940 * l5 + 1828 * l6 − 769 * l7 + 2752 * l8 − 1980 * l9 + 792 * l10));

x(i + 1) = x(i) + (41 * k1 + 216 * k6 + 27 * k7 + 272 * k8 + 27 * k9 + 216 * k10 + 41 * k11)/840,

y(i + 1) = y(i) + (41 * l1 + 216 * l6 + 27 * l7 + 272 * l8 + 27 * l9 + 216 * l10 + 41 * l11)/840,

z(i + 1) = z(i) + (41 * m1 + 216 * m6 + 27 * m7 + 272 * m8 + 27 * m9 + 216 * m10 + 41 * m11)/840.

where: xn + 1 = xn + [(41k1 + 216k6 + 27k7 + 272k8 + 27k9 + 216k10 + 41k11)/840]. The results of the three methods are shown in Table 1.

Table 1. Shown the comparison between results of 4^{th}, 6^{th}, and 7^{th} order of R-K numerical method and get more accuracy.

The Runge-Kutta method						
Step size	4th order		6th order		7th order	
	Error	Time in Second	Error	Time in Second	error	Time in Second
dt/1 = 0.01	Inf	1.4	Inf	6.2	Inf	3.9
Dt/2 = 0.005	2216.9968	1.4	Inf	6.0	Inf	4.2
Dt/4 = 0.0025	174.5643	1.4	320,083.898274	6.0	319,518.685882	3.8

Continued

Dt/8 = 0.00125	12.2611	1.4	2,539,526.94512	6.0	2,539,945.04633	3.8
dt/16 = 0.000625	0.8125	1.4	1,186,189.72471	6.0	1,189,148.66528	3.9
Dt/32 = 0.0003125	0.0523	1.4	273,214.706571	6.0	272,023.301318	3.7
Dt/64 = 0.00015625	0.0033	2.1	217,761.337783	6.0	216,946.587361	3.9
Dt/128 = 7.8125e−5	4.6614e−5	1.4	3204.148445892	6.0	3139.59110565	3.9
Dt/256 = 3.90625e−5	ERROR LIMIT SATISFIED		55.9530969299	6.0	55.7923759269	3.8
Dt/512 = 1.953125e5			2.735944731744	6.0	2.72836970305	3.8
Dt/1024 = 9.765625e−6			ERROR LIMIT SATISFIED		ERROR LIMIT SATISFIED	
Total Time to execution	12.0		66.2		42.5	

5. Chaotic Analysis

To explain the chaotic state we will take the (0-1) test to know if the system is regular or chaotic.

The Binary Test (0-1)

Consider scalar observable $\phi(k)$:

$$P_n = \sum_{k=1}^{n} \phi(k)\cos(kc), \quad q_n = \sum_{k=1}^{n} \phi(k)\sin(kc)$$

where $c \in (0, \pi)$, $n = 1, 2, 3, \cdots, L$ from behavior of P_n and q_n can be computing the Mean Square Displacement (MSD) = $M(n)$

$$M(n) = \lim_{L\to\infty}\left(\frac{1}{L}\right)\sum_{k=1}^{L}\left[\left(P(k+n) - P(k)\right)^2 + \left(q_n(k+n) - q_n(k)\right)^2\right]$$

where $n = 1, 2, \cdots, L/10$.

$$Vosc(n) = \left[E(\phi)\right]^2 \times \frac{1-\cos(nc)}{1-\cos(c)},$$

where $E(\phi) = \lim_{L\to\infty}\left(\frac{1}{L}\right)\sum_{k=1}^{n}\phi(k)$.

Then $D(n) = M(n) - Vosc(n)$,

$$Kcorr = Kc = \lim_{n\to\infty}\frac{\log Mc(n)}{\log(n)}.$$

Kc states:

Either the value of $K \cong 0$ it is signifying to regular dynamics.

Or the value of $K \cong 1$ it is signifying to chaotic dynamics, where $Kc \in [0,1]$.

Remark: if the motion is torus then the dynamic system is regular (non-chaotic), and if it behaves like a Brownian motion then the dynamic system is said it chaotic.

the Chaotic of simulated data of the 4^{th} order for Iraq by using Zero-one test shown in Figure 1, the chaotic of simulated data of 6^{th} order for Iraq by using Zero-one test shown in Figure 2 and in Figure 3 shows the chaotic of simulated data of 7^{th} order by Zero-one test for Iraq.

Then the behavior of disease is chaotic (in Iraq). The results of the three methods are shown in Table 2.

Table 2. Shown the results of kcorr of 4th, 6th and 7th order of R-Knumerical method for Iraq.

Order of Method	4^{th}	6^{th}	7^{th}
Kcorr	0.912	0.9212	0.9560

6. Graphical Analysis

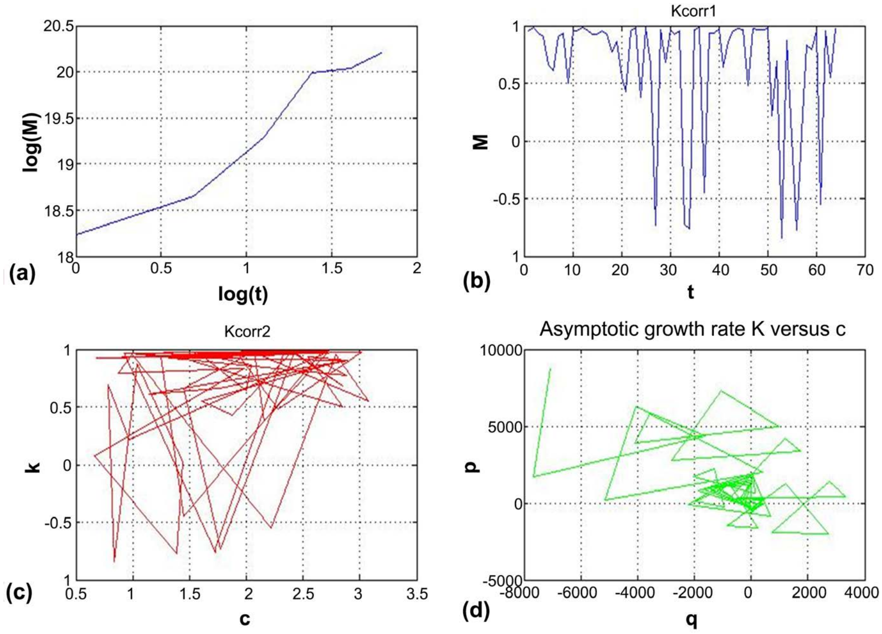


Figure 1. Show a chaotic of simulated data by numerical method of 4^{th} order for Iraq. (a) log(M) versus log(t); (b) (M) versus (t); (c) (k) versus (c); (d) (p) versus (q).

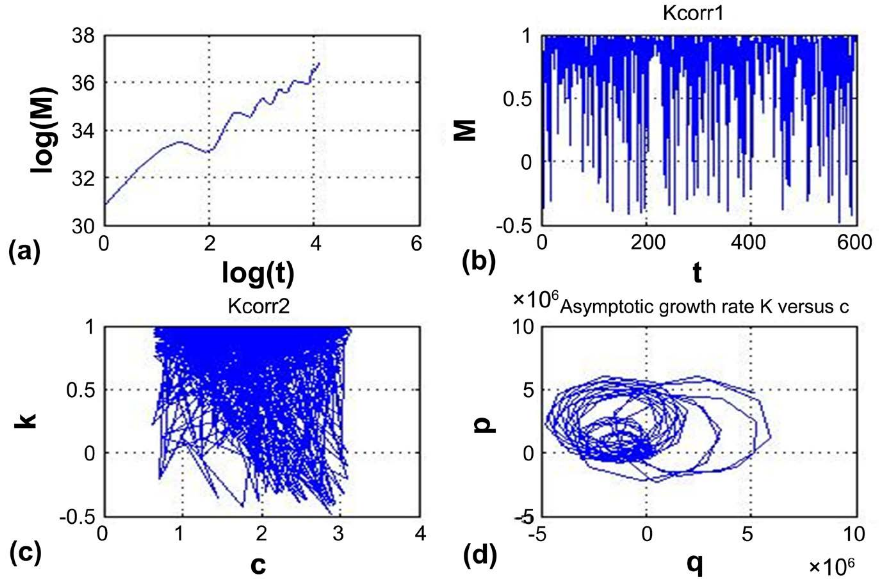


Figure 2. Shown chaotic system of R-K 6^{th} order method. (a) log(M) versus log(t); (b) (M) versus (t); (c) (k) versus (c); (d) (p) versus (q).

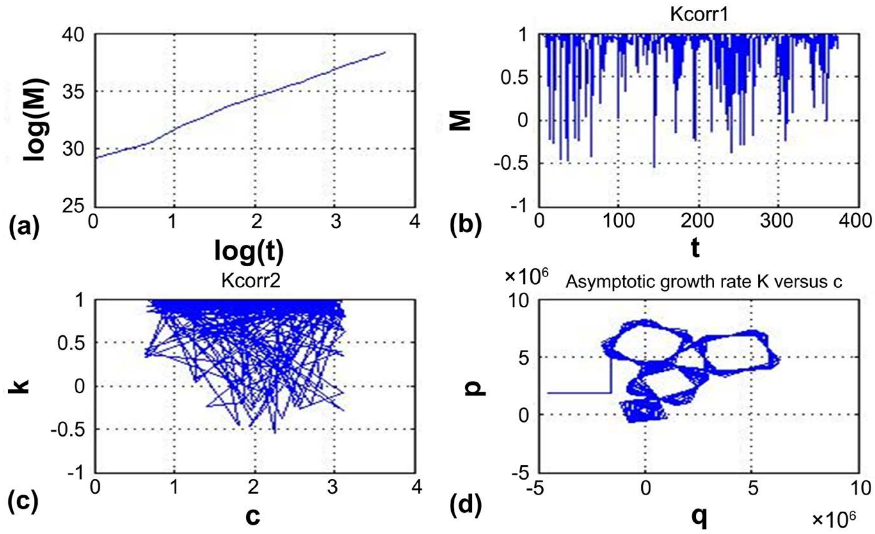


Figure 3. Shown chaotic system of R-K 7th order method for Iraq. (a) log(M) versus log(t); (b) M_n versur time (t); (c) K versus C; (d) P_n versus q_n.

7. Conclusions

We conducted a study on the error value when solving nonlinear problems and obtaining approximate values of the results as well as the time limit and the total execution time of the previously estimated error for the process of calculating a dynamical system consisting of ordinary differential equations and comparing the results which we obtained it.

In this paper, we dealt with the SIR mathematical model to study the characteristics of the epidemic disease (COVID-19), and We have used the numerical Runge-Kutta method of order 4th, 6th, and 7th to obtain a comparison between the results in terms of the estimated error value, the time limits for each step and the total time taken to implement the process in the program. Our choice to the orders of the numerical method is to obtain a more accurate solution with the least error and the shortest time, and we note the difference by building a table of the results obtained showing us that. Initial values were used for the application in resolving the system which was obtained from statistics and data on Covid-19 for a specific population from among the world's population, which is (Iraq). The elementary values were applied to the composite system from the nonlinear SIR equations with the three-order numerical method (R-K) and it gave great benefit in the information. The binary test is used for separate analysis of deterministic dynamical systems and is also used to test the chaos of the dynamic disease system. And we have applied the test on (1-0) on the model for each of the 4th, the 6th, and the 7th orders, and the result for all orders indicate that the behavior of the disease is chaotic (Kcorr of 4th ord. = 0.912 ≅ 1), (Kcorr of 6th ord. = 0.9212 ≅ 1) and (Kcorr of 7th ord. = 0.9560 ≅ 1). The result with the application of the 7th rank was better and more chaotic than the result of the application of the other ranks, Programs have been built in the Matlab system to perform all operations. Mathematical work through it and obtaining all the results and required values and figures illustrate our idea to provide organized scientific research that provides researchers with a good and useful idea.

Acknowledgements

The authors are very grateful to the University of Mosul/College of Computer Sciences and Mathematics for their provided facilities, which helped to improve the quality of this work.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Wiggins, S. (2003) Introduction to Applied Nonlinear Dynamical Systems and Chaos. Second Edition, Springer-Verlag, New York.

[2] Simon, C.M. (2020) The SIR Dynamic Model of Infectious Disease Transmission and Its Analogy with Chemical Kinetics. *PeerJ Physical Chemistry*, **2**, e14. https://doi.org/10.7717/peerj-pchem.14

[3] Rodrigues, H.S. (2016) Application of SIR Epidemiological Model: New Trends. *International Journal of Applied Mathematics and Informatics*, **10**, 92-97.

[4] Weiss, H.H. (2013) The SIR model and the Foundations of Public Health. *Materials Matemàtics*, **2013**, 17 p.

[5] Aziz, M.M. and Mahmood, A.S. (2021) Analysis of Dynamical Behavior for Epidemic Disease COVID-19 with Application. *Turkish Journal of Computer and Mathematics Education*, **12**, 568-577. https://doi.org/10.17762/turcomat.v12i4.538

[6] Hossain, T., Miah, M. and Hossain, B. (2017) Numerical Study of Kermack-Mckendrick SIR Model to Predict the Outbreak of Ebola Virus Diseases Using Euler and Fourth Order Runge-Kutta Method. *American Scientific Research Journal for Engineering, Technology, and Sciences*, **37**, 1-21.

[7] Al-Shimmary, A.F. (2017) Solving Initial Value Problem Using Runge-Kutta 6th Order Method. *ARPN Journal of Engineering and Applied Sciences*, **12**, 3953-3961.

[8] BaZezew, H. and Adamu, G. (2019) Numerical Solution of Second Order Initial Value Problems of Bratu-Type Equations Using Sixth Order Runge-Kutta Seven Stages Method. *International Journal of Computing Science and Applied Mathematics*, **5**, 10-14. https://doi.org/10.12962/j24775401.v5i1.3806

[9] Huťa, A. and Penjak, V. (1984) Contribution to Runge-Kutta formulas of the 7th Order with the Rational Coefficients for System of Differential Equations of the First Order. *Aplikace Matematiky*, **29**, 411-422. https://doi.org/10.21136/AM.1984.104115

[10] Aziz, M.M. and Merie, D.M. (2020) Stability and chaos with mathematical control of 4-d dynamical system. *Indonesian Journal of Electrical Engineering and Computer Science*, **20**, 1242-1251. https://doi.org/10.11591/ijeecs.v20.i3.pp1242-1251

[11] Gottwald, G.A. and Melbourne, I. (2009) On the Implementation of the 0-1 Test for Chaos. *SIAM Journal on Applied Dynamical Systems*, **8**, 129-145. https://doi.org/10.1137/080718851

[12] Aziz, M.M. and Hamid, M. (2019) The Possibility of Increasing The Predictability Indices After Control of 3D-Continuous-Time System. 2019 *International Confe-*

rence on Computing and Information Science and Technology and Their Applications (*ICCISTA*), Kirkuk, Iraq, 3-5 March 2019, 1-5. https://doi.org/10.1109/ICCISTA.2019.8830650

[13] Wontchui, T.T. and Effa, J.Y. (2017) Henri Paul Ekobena Fouda, Jean Sire Armand Eyebe Fouda, Dynamical Behavior of Peter-De-Jong Map Using the Modified (0-1) and 3ST Tests for Chaos. *Annual Review of Chaos Theory*, *Bifurcations and Dynamical Systems*, **7**, 1-21.

[14] Gopal, R., Venkatesan, A. and Lakshmanan, M. (2013) Applicability of 0-1 Test for Strange Non-Chaotic Attractors. *Chaos*: A*n Interdisciplinary Journal of Nonlinear Science*, **23**, Article No. 023123. https://doi.org/10.1063/1.4808254

[15] Aziz, M.M. and Faraj, M.N. (2015) Difficulty of Predicting Earthquakes in Mosul Dam. *International Journal of Technical Research and Applications*, **3**, 29-36.

[16] Aziz, M.M. and Faraj, M.N. (2012) Numerical and Chaotic Analysis of Chua's Circuit. *Journal of Emerging Trends in Computing and Information Sciences*, **3**, 783-791.