People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

MOHAMED KHIDER UNIVERSITY of BISKRA

FACULTY of EXACT SCIENCES and SIENCES of NATURE and LIFE

DEPARTMENT OF MATHEMATICS

Thesis Submitted in Partial Execution of the Requirements of the Degree of

Master In Applied Mathematics

Option: **ANALYSIS**

Presented by:

Ouafa GHEDJEMIS

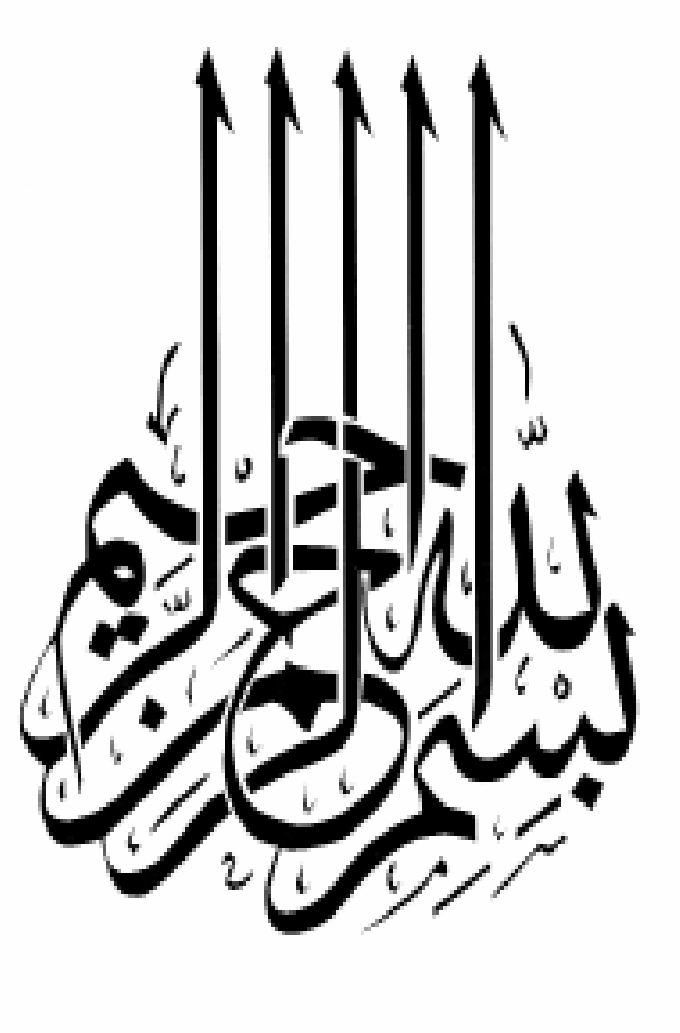
Titled:

Numerical Solutions of Differential Equations System

Examination Committee Members:

Dr.	**Fatma KACI**	MKUB	Chairperson
Dr.	**Fatima OUAAR**	MKUB	Supervisor
Dr.	**Abedelkader LAIADI**	MKUB	Examiner

26^{th} June 2022


بسم الله الرحمن الرحيم

بسم الله الرحمن الرحيم

الحمد لله الذي هدانا لهذا و ما كنا لنهتدي لولا أن هدانا الله

سورة الأعراف الآية 43

Dedication

I dedicate this humble work to

My dear mother.

My father may "God" have mercy on him.

My dear grandmother.

My dear sisters and their children.

My uncles and all my familly members.

To all my friends and coworkers.

To everyone that helped me to finish this thesis.

Ouafa GHEDJEMIS

Acknowledgements

This work is an end of a long journey, despite facing so many hardships in work, in studies and even in personal life I finished it.

In the first place, I'd like to thank "God" the Almighty for helping me succeed and for giving me courage and patience to carry out this work.

The accomplishment of this work would not have been possible without the support and collaboration of many people whom I would like to sincerely thank:

First of all, I would like to thank Mrs. **Fatima OUAAR,** the one who was the best honorable and the best advisor to me and did not skimp on us with her support, guidance, opinions and advice, she has the utmost thanks and gratitude from me.

I want to thank the jury members that accepted the discussion of this thesis, starting with Mrs. **Fatma KACI**, chairperson then Mr. **Abedelkader AIADI**, examiner. I express my deep gratitude to them.

I would also like to thank all the Mathematics department members especially the head of department Mr. **Imad Eddine LAKHDARI** and all the teachers at the university of Mohamed khider, for their help and support.

Not forgetting a special thanx to my sister and teacher **Fatiha GHEDJEMIS** who has been my right arm since the very start of my school life.

Finally, I must express my very profound gratitude to my parents, sisters, friends and coworkers for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of writing this thesis.This accomplishment would not have been possible without them.

Thank you all.

GHEDJEMIS.

Contents

Dedication i

Acknowledgements ii

Table of Contents iii

List of Figures vi

General Introduction 1

Differential Equations System 2

1 Differential Equations System 3

1.1 Ordinary Differential Equations 3

1.2 First Order Differential Equations 5

1.2.1 First Order Linear Differential Equations 5

1.2.2 Separable Differential Equations 5

1.3 Cauchy Problem 6

1.4 Homogeneous and Non-Homogeneous Differential Equations: 7

1.5 Differential System 8

1.5.1 System of Equations 8

1.5.2 System of Ordinary Differential Equations 10

1.5.3 Notions about Stability 11

1.6 Numerical Solution of Differential Equations 12

1.6.1 Euler's Method . . . 12
1.6.2 The Quadratic Taylor Method . . . 14
1.6.3 Euler's Midpoint Method . . . 15
1.6.4 Runge-Kutta methods . . . 16

The Lotka-Volterra System **17**

2 The Lotka-Volterra System **18**

2.1 Lotka-Voltera Model . . . 18
2.2 Problem of Existence . . . 19
2.2.1 Local Existence . . . 19
2.2.2 Existence and Positivity of Solution . . . 20
2.2.3 Periodicity . . . 22
2.2.4 Equilibrium Points . . . 22

MATLAB Resolution of Lotka-Volterra Model **24**

3 Matlab Resolution of Lotka-Volterra Model **25**

3.1 Example of Lotka-Volterra Model . . . 25
3.2 Solving the System Using Euler's Method . . . 26
3.2.1 Euler's Code for the step size h=0.05 . . . 26
3.2.2 Euler's Code for the step size h=0.001 . . . 27
3.2.3 The Phase-Plane Trajectories using Euler Method . . . 28
3.3 Solving the System using Runge-Kutta Fourth-Order Method . . . 28
3.3.1 Solving the Predater-Prey Model using ode45 . . . 28
3.3.2 The Phase-Plane Trajectories using RK4 Method . . . 29
3.4 Concluding Remarks . . . 30

General Conclusion **31**

Bibliography **32**

Appendix A: MATLAB **34**

Appendix B: Abreviations and Notations **36**

Appendix C : MATLAB's Code Used **37**

3.5 Euler's Code 38

3.5.1 Euler's Code for the step size h=0.05 38

3.5.2 Euler's Code for the step size h=0.001 39

3.5.3 The Phase-Plane Trajectories 39

3.6 RK4's Code 40

3.6.1 Solving the Predater-Prey Model using ode45 40

3.6.2 The Phase-Plane Trajectories: 41

List of Figures

2.1 Example of Lotka-Volterra model. 18

3.1 Prey-Predator example 25

3.2 Solutions of prey-predator system using Euler's method with step size h=0.05 . . 27

3.3 Solutions of prey-predator system using Euler's method with step size h=0.001 . 27

3.4 Solutions' trajectories with Euler's method 28

3.5 Solutions of prey-predator system using RK-4 method (ode45) 29

3.6 Solutions' trajectories with RK4's method 30

3.7 MATLAB icon 35

General Introduction

Almost any natural, biological and physical phenomenon is described by differential equations or system of differential equations. And since we can't get exact solutions we often must solve them with numerical methods, such as Euler's and Runge Kutta's methods. One of the most popular modelled biological phenomenon, is the predator-prey model that is interpreted by Lotka-Volterra system of equations. But what's it's history? and why we thought about their solutions?

Just after the First World War (a period where fishing had been clearly reduced), an Italian mathematician and physicist Volterra had been consulted by the responsible of Italian fishing in Trieste who had noticed that, the proportion of sharks and other predators which are unsuitable for consumption, that were caught among the edible fish was clearly higher than it was before the war.

In 1926 a reply in Italian was published by Volterra [9] which took the form of the famous predator-prey model. He then presented his results in English and, in more detailed and complete form, in French [[10],[11],[12]] in the first chapter of his book ''Leçons sur la théorie mathématique de la lutte pour la vie '' [12]. An American mathematician, Alfred James Lotka, published independently in 1924 (or 25 dates vary by source), an equivalent model. It is referred to today as name of "Lotka-Volterra model" so as not to make anyone jealous... It has been extended to several domains of the dynamics of populations and is widely used today in biology, ecology and even chemistry.

The model is written as follows:

$$\begin{cases} x'(t) = ax - bxy \\ y'(t) = cxy - dy \end{cases}$$

- x represents the number of prey.

- y represents the number of predators.
- $a > 0$ is the prey's birth rate.
- $d > 0$ is the predator's death rate.
- $b > 0$ and $c > 0$ are interactions coefficients between the two populations.

For obvious reasons, we are only interested in this system for positive values of x and y. Many researchers have been interrested by Lotka-Volterra systems we site here some of their works: In 1989, S.C. Bhargava generalized Lotka-Volterra equations and desribed the mechanism of technological substitution [1]

In 2009, C. Zhu and G. Yin discussed a hybrid competitive Lotka–Volterra ecosystems [13]. And recently in september 2022, Lei Li, Wan-Tong Li and Mingxin Wang studied a classical Lotka-Volterra predator-prey model with nonlocal diffusions with free boundary [4].

This work is organized as follows:

Chapter 1: In the first chapter we recall, without proof, some definitions and fundamental properties about the study of ordinary differential equations, which will be used directly in the rest of this work, let us quote for example: Ordinary differential equations, First order differential equations, Homogeneous and non-homogeneous differential equations, Differential system (System of ordinary differential equations), Numerical solution of differential equations.

Chapter 2: The second chapter deals with the application of the concepts introduced in the first chapter in the qualitative study of the solutions of the Lotka-Volterra system (existence, positivity and periodicity of the solutions and behavior in the vicinity of its equilibrium points).

Chapter 3: This chapter describes numerical results that we obtained while solving Lotka-Volterra system using two popular numerical methods, Euler and Runge-Kutta methods, implemented on MATLAB. In addition, a discussion and a comparaison between the results obtained by this two methods are presented.

Chapter 1

Differential Equations System

Differential equations are found everywhere to the point where its systems governs our real life begining by the first order differential equations system, they can also describe informations about population of one species, give details about prey and predators' problems.

1.1 Ordinary Differential Equations

A differential equation is an equation that contains one or more functions with its derivatives. The derivatives of function define the function's rate-of-change at a point. If all the derivatives are taken related to a single variable, we speak of an ordinary differential equation (ODE).

Definition 1.1.1 *An ordinary differential equation, abbreviated by ODE, of order n is a relation between the real variable t, an unknown function $t \longmapsto y(t)$ and its derivatives $y' = \frac{dy}{dt}, y'', ..., y^{(n)}$ on point t defined by:*

$$F\left(t, y, y', y'', ..., y^{(n)}\right) = 0 \tag{1.1}$$

Where F is not independent of its last variable $y^{(n)}$, we take t in an interval $I \subset \mathbb{R}$. The solution y generally will be a value on $\mathbb{R}^n$, such as $n \in \mathbb{N}^*$. If F has values in $\mathbb{R}$, then the equation 1.1 is scalar. The order of a differential equation is the order of the highest derivative present in the equation.

Example 1.1.1 *The following equations are ordinary differential equations :*

$y' + y = 3t^2$.

$y^{'} + ty - 1 = 0.$

$y^{''} - 3y = 0.$

$y^{'} = \sin(t).$

Definition 1.1.2 *We call autonomous differential equation of order n any equation of the form:*

$$y^{(n)} = f\left(y, y^{'}, y^{''}, ..., y^{(n-1)}\right),$$

which means that f doesn't explicitly depend on t.

Remark 1.1.1 *Autonomous equations are very important when looking for stationary solutions and their stability.*

Linear and Non-linear Ordinary Differential Equations:

The ordinary differential equation 1.1 of order n is linear if it has the form:

$$a_n(t)y^{(n)} + a_{n-1}(t)y^{(n-1)} + ... + a_0(t)y = g(t).$$

Knowing that $a_0(t), ..., a_n(t)$ and $g(t)$ can be zero or non-zero, constant or non-constant, linear or non-linear functions, according to the variable t. We note that there are no products of the function, $y(t)$ and its derivatives. Also the function and its derivatives occurs only the first power.

Only the unknown function ,$y(t)$, and it's derivatives are used in determining if a differential equation is linear. If a differential equation is not linear depending on the unknown function y and its derivatives, then it is called a nonlinear differential equation. Also we can say that the trigonometric functions are nonlinear.

Example 1.1.2 $y^{'} + 2y = 0$ *is linear.*

$y^{''} + y^{'} = 3$ *is linear.*

$y^{''} - sin(y) = 0$ *is non-linear.*

$y^{'} + y^2 = 0$ *is non-linear.*

$(1 - y)y^{'} + 3y = 0$ *is non-linear.*

1.2 First Order Differential Equations

Definition 1.2.1 *A first order differential equation is an equation of the form:*

$$F\left(t, y, \acute{y}\right) = 0, \tag{1.2}$$

where y is the unknown function of the variable t (or x) to value in $\mathbb{R}^n$, $y^{'}$ its derivative and F is a function of a part of $\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n$ towards $\mathbb{R}^n$.

Remark 1.2.1 *The ordinary y for the equation 1.2 depends on a single variable t. When there are several derivatives that depends on several variables we speak about partial differential equations (PDE).*

1.2.1 First Order Linear Differential Equations

Definition 1.2.2 *We say that the first order differential equation $F\left(t, y, y^{'}\right) = 0$ is a linear equation if it can be written in the form*

$$y^{'} = p\left(t\right) y + q\left(t\right),$$

where p and q are continuous functions on an interval $I \subset \mathbb{R}$.

Example 1.2.1 *The equation $ty^{'} + 2y = t^2$ is a first order linear differential equation.*

For a non-linear first order differential equation we can take the Ricatti equation as an example which is:

$$t^2 y^{'} + y + y^2 = 0.$$

1.2.2 Separable Differential Equations

Definition 1.2.3 *Generally, a separable differential function is any differential function that can be written in the following form:*

$$N(y)y^{'} = M(t),$$

where N and M are two functions defined respectively on I and K (I and K are intervals of

$\mathbb{R}$*). Note that in the separable differential equation all the* y *must be multiplied by the derivative and all the* t *must be on the other side of the equal sign.*

Example 1.2.2 *The following equations are separable differential equations:*

$y^{'} = 3y^2t.$

$y^{'} = \frac{2t^2+3t-3}{y-3}.$

$y^{'} = e^{-y}(3t+2).$

1.3 Cauchy Problem

A Cauchy problem is given by a differential equation and an initial condition (also called initial value problem). So it's a problem of the type:

$$\begin{cases} y^{'}(t) = f(t, y(t)), \\ \quad y(t_0) = y_0 \end{cases} \tag{1.3}$$

with $t_0 \in I$, $y_0 \in \Omega$ and $f : I \times \Omega \to \mathbb{R}^m$ a continuous function.

Lipschitz and Locally Lipschitz Function

Definition 1.3.1 *Let* $f : U \times \Omega \to \mathbb{R}^m$ *be a function* $(m \geq 1)$*. We say that* f *is lipschitz in* y *(uniformly with respect to* t*), and we will note* $f \in Lip(U)$*, if there exists* $k > 0$ *such that:*

$$\|f(t, y_2) - f(t, y_1)\|_{\mathbb{R}^m} \leq k \|y_2 - y_1\|_{\mathbb{R}^m},$$

for each $(t, y_1), (t, y_2) \in U$*. Notice that this notion doesn't lead to* f *being continious on* U *as the following example proves:*

$$U = \mathbb{R} \times \times \mathbb{R}, \ f(t, y) = \begin{cases} 1, \ \textit{if } t > 0 \\ 0, \ \textit{if } t \leq 0 \end{cases}.$$

However, if f *is Lipschitz function in the classical meaning, i.e if there exists* $k > 0$ *such that:*

$$\|f(t, y_2) - f(t, y_1)\|_{\mathbb{R}^m} \leq k(|t_2 - t_1| + \|y_2 - y_1\|_{\mathbb{R}^m});$$

for each $(t, y_1), (t, y_2) \in U$, *then* f *is especially Lipschitz in* y *(uniformly in* t*).*

Definition 1.3.2 *We say that* f *is locally Lipschitz in* U *if for each* $(t, y_1) \in U$, *there is a ball* $B = \left\{(t^{'}, y^{'}) \in U; \left|t^{'} - t_1\right| < \varepsilon \text{ and } \left\|y^{'} - y_1\right\|_{\mathbb{R}^m} < \varepsilon\right\}$ *and a constant* $k > 0$ *such as* f *is Lipschitz, on* B. *Then we note* $Lip_{loc}(U)$.

Remark 1.3.1 - *A Lipschitzian application is continuous.*

- *A function of class* C^1 *is locally Lipschitzian.*

1.4 Homogeneous and Non-Homogeneous Differential Equations:

Definition 1.4.1 *A first order homogeneous linear differential equation is one of the form:*

$$y^{'} + p(t)y = 0$$

or equivalently $y^{'} = -p(t)y$.

Definition 1.4.2 *A first order non-homogeneous linear differential equation is one of the form:*

$$y^{'} + p(t)y = f(t)$$

Example 1.4.1 $ty^{'} + 2y = 0$ *is homogeneous.*

$2y^{'} + y\sin(t) = 0$ *is homogeneous.*

$y^{'} + 2\frac{y}{t} = t^3$ *is non-homogeneous.*

$y^{'} - y\exp(t) = 0$ *is homogeneous.*

Proposition 1.4.1 *The solutions of the homogeneous linear equation*

$$p(t)y^{'} + q(t)y = 0$$

On the domain I, *where* $y(t_0) = y_0$ *is defined by:*

$$y(t) = y_0 \exp(F(t))$$

For all $t \in I$, *With* $F(t) = \int_{t_0}^{t} -\frac{q(s)}{p(s)}ds$

Proposition 1.4.2 *The general solution of the non-homogeneous equation*

$$p(t)y^{'} + q(t)y = r(t),$$

On I where $y(t_0) = y_0$, equal to

$$y(t) = \exp\left(\int_{t_0}^{t} -\frac{q(s)}{p(s)}ds\right)\left(y_0 + \int_{t_0}^{t}\frac{r(s)}{p(s)}\exp\left(\int_{t_0}^{s}\frac{q(\sigma)}{p(\sigma)}d\sigma\right)ds\right).$$

Particular case:

Proposition 1.4.3 (Duhamel formula) *Proposed a continuous function on the interval I of $\mathbb{R}$, a a real constant and $t_0 \in I$ where $y(t_0) = y_0$. the general solution of the scalar equation*

$$y^{'} = ay + g(t)$$

is given by:

$$y(t) = y_0 e^{a(t-t_0)} + \int_{t_0}^{t} e^{a(t-s)}f(s)ds,$$

where a is a constant.

1.5 Differential System

1.5.1 System of Equations

Let's start with the following system of n equations with the n unknowns, $y_1, y_2, ..., y_n$.

$$\begin{cases} a_{11}y_1 + a_{12}y_2 + ... + a_{1n}y_n = b_1. \\ a_{21}y_1 + a_{22}y_2 + ... + a_{2n}y_n = b_2. \\ \qquad\qquad . \\ \qquad\qquad . \\ \qquad\qquad . \\ a_{n1}y_1 + a_{n2}y_2 + ... + a_{nn}y_n = b_n. \end{cases} \tag{1.4}$$

Note that in the subscripts on the coefficients in this system, a_{ij}, the i corresponds to the equation that the coefficient is in and the j corresponds to the unknown that is multiplied by

the coefficient.

To use linear algebra to solve this system we will first write down the augmented matrix for this system. An augmented matrix is really just all the coefficients of the system and the numbers for the right side of the system written in matrix form. Here is the augmented matrix for this system:

$$\begin{pmatrix} a_{11} & a_{12} & . & . & . & a_{1n} & b_1 \\ a_{21} & a_{22} & . & . & . & a_{2n} & b_2 \\ . & . & . & & & . & . \\ . & . & & . & & . & . \\ . & . & & & . & . & . \\ a_{n1} & a_{n2} & . & . & . & a_{nn} & b_n \end{pmatrix}$$

To solve this system we will use elementary row operations (which we'll define these in a bit) to rewrite the augmented matrix in triangular form. The matrix will be in triangular form if all the entries below the main diagonal (the diagonal containing a_{11}, a_{22}, ... , a_{nn}) are zeroes.

Remark 1.5.1 *Given a system of equations, 1.4 , we will have one of the three possibilities for the number of solutions:*

1. *No solution.*
2. *Exactly one solution.*
3. *Infinitely many solutions.*

The system of equations in 1.4 is called a nonhomogeneous system if at least one of the b_i is not zero. If however all of the b_i are zero we call the system homogeneous and it will be as follows:

$$\begin{cases} a_{11}y_1 + a_{12}y_2 + ... + a_{1n}y_n = 0. \\ a_{21}y_1 + a_{22}y_2 + ... + a_{2n}y_n = 0. \\ \quad . \\ \quad . \\ \quad . \\ a_{n1}y_1 + a_{n2}y_2 + ... + a_{nn}y_n = 0. \end{cases} \tag{1.5}$$

Notice that in the homogeneous case we are guaranteed to have the following solution

$$y_1 = y_2 = ... = y_n = 0.$$

This solution is often called the trivial solution. For homogeneous systems the remark above can be modified to the following.

Remark 1.5.2 *Given a homogeneous system of equations, 1.5, we will have one of the two possibilities for the number of solutions.*

1. *Exactly one solution, the trivial solution.*
2. *Infinitely many non-zero solutions in addition to the trivial solution.*

1.5.2 System of Ordinary Differential Equations

Definition 1.5.1 *A system of ordinary differential equations is a set of n equation involving the derivatives of n unknown functions with a single independent variable. We can write it on the form*

$$\vec{y}' = A\vec{y} + \vec{g}(t)$$

where $\vec{y}' = (y_1', y_2', ..., y_n')$, $\vec{y} = (y_1, y_2, ..., y_n)$ *and* $\vec{g}(t) = (b_1, b_2, ..., b_n)$. *It can be provided with initial or boundary value conditions.*

Example 1.5.1 *We can take the first-order differential equations system below as an example:*

$$\begin{cases} y_1' = 4y_1 + 7y_2. \\ y_2' = -2y_1 - 5y_2. \end{cases}$$

It can be written in the form:

$$\begin{pmatrix} y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} 4 & 7 \\ -2 & -5 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

Remark 1.5.3

- *We say that the system is homogeneous if* $\vec{g}(t) = \vec{0}$ *and we say the system is nonhomogeneous if* $\vec{g}(t) \neq \vec{0}$.
- *Equilibrium solutions are those solutions for which* $A\vec{y} = \vec{0}$.

1.5.3 Notions about Stability

Consider the finite dimension continuous system described by a first order non-linear autonomous differential equation:

$$y^{'}(t) = f(y(t)), \tag{1.6}$$

which means $f : \mathbb{R}^m \to \mathbb{R}^m$ $(m > 1)$ is a Lipschitz function, with $t \geq 0$.

The Equilibrium Point

Definition 1.5.2 *We say that a vector $E \in \mathbb{R}^m$ is an equilibrium point of f if $f(E) = 0$. The trajectory of this equilibrium is reduced to the point E.*

Remark 1.5.4 *The equilibrium point $y \equiv E$ is the unique solution of 1.6 with the initial condition $y(t_0) = E$ and $t_0 \geq 0$.*

Local Equilibrium Point Stability

Let $\varphi(t, y_0)$ the unique solution of the problem 1.6 with the initial condition $y(0) = y_0$, i.e $\varphi^{'}(t, y_0) = f(\varphi(t, y_0))$ and $\varphi(0, y_0) = y_0$.

Definition 1.5.3 *The equilibrium point $E = 0$ of the system 1.6 is:*

- ***Stable**: if, $\forall\ \varepsilon > 0, \exists\ \eta > 0 : \|y_0\| < \eta \Longrightarrow, \forall t > 0, \|\varphi(t, y_0)\| < \varepsilon$ and **unstable** if it's not stable.*
- ***Asymptotically stable:** if it's stable and if $\exists\ \eta > 0 : \|y_0\| < \eta \Longrightarrow, \lim_{t \to +\infty} \|\varphi(t, y_0)\| = 0$.*
- ***Marginally stable:** it it's stable without being asymptotically stable.*

Linearization around Equilibrium Point

Suppose that E is an equilibrium point for the differential system

$$y^{'}(t) = f(y(t)), \tag{1.7}$$

and f is differentiable at E. To approach the function $f(y(t))$, we form its jacobian matrix in the neighborhood of E

$$Df(E) = \left(\frac{\partial f_i}{\partial x_j}\right)_{1\leq i,j\leq m}$$

where the partial derivatives are calculated at point E. The affine approximation of f is written in the form:

$$f(y) = f(E) + Df(E).(y - E) = Df(E).u,$$

where $u = y - E$, so in the neighborhood of E we have

$$u^{'} = y^{'} - E^{'} = f(y) = Df(E).u,$$

because $f(E) = 0$.

Definition 1.5.4 *We call a linearized system around the point E associated with the nonlinear system 1.7 the linear system*

$$u^{'} = Df(E).u.$$

Theorem 1.5.1 *If the system 1.7 is is linearizable around E, then:*

1. *The equilibrium point E is stable if and only if the origin is stable for the linearized system.*
2. *The equilibrium point E is asymptotically stable if and only if the origin is asymptotically stable for the linearized system.*

1.6 Numerical Solution of Differential Equations

1.6.1 Euler's Method

Supposing that the differential equation is

$$y^{'} = f(t, y), y(\alpha) = y_0,$$

and our aim is to compute a sequence of approximations $(t_k, y_k)_{k=0}^{n}$ to the solution, where $t_k = \alpha + kh$. The initial condition provides us with a point on the true solution, so (t_0, y_0) is also the natural starting point for the approximation. To get an approximation to the solution

at t_1, we compute the slope of the tangent at (t_0, y_0) as $y_0^{'} = f(t_0, y_0)$. This gives us the tangent $T_0(t) = y_0 + (t - t_0)y_0^{'}$ to the solution at t_0. As the approximation y_1 at t_1 we use the value of the tangent T_0 which is given by

$$y_1 = T_0(t) = y_0 + hy_0^{'} = y_0 + hf(t_0, y_0).$$

This gives us the next approximate solution point (t_1, y_1). To step to the next point (t_2, y_2), we move along the tangent to the exact solution that passes through (t_1, y_1). The derivative at this point is $y_1^{'} = f(t_1, y_1)$ and so the tangent is

$$T_1(t) = y_1 + (t - t_1)y_1^{'} = y_1 + (t - t_1)f(t_1, y_1).$$

Then the approximate solution at t_2 is

$$y_2 = y_1 + hf(t_1, y_1).$$

If we continue in the same way, we can compute an approximation y_3 to the solution at t_3, then an approximation y_4 at t_4, and so on. So we can say:

In Euler's method, an approximate solution (t_k, y_k) is advanced to (t_{k+1}, y_{k+1}) by the following tanget

$$T_k(t) = y_k + (t - t_k)y_k^{'} = y_k + (t - t_k)f(t_k, y_k),$$

at (t_k, y_k) from t_k to $t_{k+1} = t_k + h$. This result in the approximation $y_{k+1} = y_k + hf(t_k, y_k)$ to $y(t_{k+1})$.

Algorithm of Euler's Method

Let the differential equation $y^{'} = f(t, y)$ be given together with the initial condition $y(a) = y_0$, the solution interval $[a, b]$, and the number of steps n. If the following algorithm is performed

$h = (b - a)/n$;

$t_0 = a$;

for $k = 0, 1, ..., n - 1$

$y_{k+1} = y_k + hf(t_k, y_k)$;

$t_{k+1} = a + (k + 1)h$;

the value y_k will be an approximation to the solution $y(t_k)$ of the differential equation, for each $k = 0, 1, ..., n$.

1.6.2 The Quadratic Taylor Method

Approxiamating the solution by a Taylor polynomial of a suitable degree is called Taylor method. In Euler's method, which is the simplest Taylor method, we used the approximation

$$y(t+h) \approx y(t) + hy^{'}(t).$$

The quadratic Taylor method is based on the more precise approximation

$$y(t+h) \approx y(t) + hy^{'}(t) + \frac{h^2}{2}y^{''}(t). \tag{1.8}$$

The numerical solution can be advanced from a point (t_k, y_k) to a new point (t_{k+1}, y_{k+1}) with $t_{k+1} = t_k + h$.The main idea is to use 1.8 and compute y_{k+1} as

$$y_{k+1} = y_k + hy_k^{'} + \frac{h^2}{2}y_k^{''},$$

where $y_k, y_k^{'}$ and $y_k^{''}$ are approximations to the function value and derivatives of the solution at t.

Remark 1.6.1 *The quadratic Taylor method advances the solution from a point (t_k, y_k) to a point (t_{k+1}, y_{k+1}) by evaluating the approximate Taylor polynomial*

$$y(t) \approx y_k + (t - t_k)y_k^{'} + \frac{(t-t_k)^2}{2}y_k^{''}$$

at $y = t_{k+1}$. In other word, the value y_{k+1} is given by

$$y_{k+1} = y_k + hy_k^{'} + \frac{h^2}{2}y_k^{''},$$

where $h = t_{k+1} - t_k$.

Algorithm of Taylor Method

Let the differential equation $y^{'} = f(t, y)$ be given together with the initial condition $y(a) = y_0$, the solution interval $[a, b]$ and the number of steps n, and let the function F_2 be such that $y^{''}(t) = F_2(t, y(t))$.

The quadratic Taylor method is given by the algorithm

$h = (b - a)/n;$

$t_0 = a;$

for $k = 0, 1, ..., n - 1$

$y_k^{'} = f(t_k, y_k);$

$y_k^{''} = F_2(t_k, y_k);$

$y_{k+1} = y_k + hy_k^{'} + h^2 y_k^{''}/2;$

$t_{k+1} = a + (k + 1)h;$

After these steps the value y_k will be an approximation to the solution $y(t_k)$ of the differential equation, for each $k = 0, 1, ..., n$.

Remark 1.6.2 *The advantage of the Taylor methods is that they can attain any approximation order, and their disadvantage is that they require symbolic differentiation of the differential equation (except for Euler's method).*

1.6.3 Euler's Midpoint Method

In Euler's midpoint method the solution is advanced from (t_k, y_k) to $(t_k + h, y_{k+1})$ in two steps: First an approximation to the solution is computed at the midpoint $t_k + h/2$ using Euler's method with step length $\frac{h}{2}$,

$$y_{k+1/2} = y_k + \frac{h}{2} f(t_k, y_k).$$

Then the solution is advanced to t_{k+1} by following the straight line from (t_k, y_k) with slope given by $f(t_k + h, y_{k+1/2})$,

$$y_{k+1} = y_k + hf(t_k + \frac{h}{2}, y_{k+1/2}).$$

Algorithm of Euler's midpoint method

Let the differential equation $y(t) = f(t, y)$ be given together with the initial condition $y(a) = y_0$, the solution interval $[a, b]$, and the number of steps n. Euler's midpoint method is given by

$h = (b - a)/n;$

$t_0 = a;$

for $k = 0, 1, ..., n - 1$

$y_{k+1/2} = y_k + hf(t_k, y_k);$

$y_{k+1} = y_k + hf(t_k + h/2, y_{k+1/2});$

$t_{k+1} = a + (k+1)h;$

After these steps the value $y_k$will be an approximation to the solution $y(t_k)$ of the differential equation at t_k, for each $k = 0, 1, ..., n$.

1.6.4 Runge-Kutta methods

Runge-Kutta methods are generalisations of the midpoint Euler method. The methods use several evaluations of f between each step in a clever way which leads to higher accuracy.

In the simplest Runge-Kutta methods, the new value y_{k+1} is computed from y_k with the formula

$$y_{k+1} = y_k + h(\lambda_1 f(t_k, y_k) + \lambda_2 f(t_k + r_1 h, y_k + r_2 h f(t_k, y_k)), \tag{1.9}$$

where $\lambda_1, \lambda_2, r_1$ and r_2 are constants to be determined. The idea is to choose the constants in such a way that the relation 1.9 mimics a Taylor method of the highest possible order. It turns out that the first three terms in the Taylor expansion can be matched. This leaves one parameter free (we choose this to be $\lambda = \lambda_2$ and set the other three in terms of λ,

$$\lambda_1 = 1 - \lambda \ \ , \ \lambda_2 = \ \lambda \ \ , \ \ r_1 = r_2 = \frac{1}{2\ \lambda}.$$

This determines a family of second order accurate methods.

Theorem 1.6.1 (Second order Runge-Kutta methods) *Let the differential equation $y' = f(t, y)$ with initial condition $y(a) = y_0$ be given. Then the numerical method which advances from (t_k, y_k) to (t_{k+1}, y_{k+1}) according to the formula*

$$y_{k+1} = y_k + h\left((1-\lambda)f(t_k, y_k) + \lambda f\left((t_k + \frac{h}{2\lambda}, y_k + \frac{hf(t_k, y_k)}{2\lambda}\right)\right)$$

is 2nd order accurate for any nonzero value of the parameter λ, provided f and its derivatives up to order two are continuous and bounded for $t \in [a, b]$ and $y \in \mathbb{R}$.

Theorem 1.6.2 (Fourth order Runge-Kutta method) *Suppose the differential equation* $y' = f(t, y)$ *with initial condition* $y(a) = y_0$ *is given. The numerical method given by the formulas*

$$\begin{cases} k_0 = f(t_k, y_k), \\ k_1 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2}k_0), \\ k_2 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2}k_1), \\ k_3 = f(t_k + h, y_k + hk_2), \\ y_{k+1} = y_k + \frac{h}{6}(k_0 + 2k_1 + 2k_2 + k_3), \end{cases}$$

is 4th order accurate provided the derivatives of f *up to order four are continuous and bounded for* $t \in [a, b]$ *and* $x \in \mathbb{R}$.

Chapter 2

The Lotka-Volterra System

After the first mondial war in a period where fishing had been clearly reduced where the proportion of sharks and other predators (unsuitable for consumption) was clearly higher than it was before the war. The responsible of Italian fishing in Trieste had noticed that and consult an Italian mathematician called Vito Volterra, who designed a model to describe a phenomenon which is: "the predator fish's proportion increases when the fishing dicreases", and in another hand an equivalent model was published independently by an American mathematician, Alfred James Lotka. They called it Lotka-Volterra model.

2.1 Lotka-Voltera Model

We will present a graph that shows the specific Lotka-Volterra system:

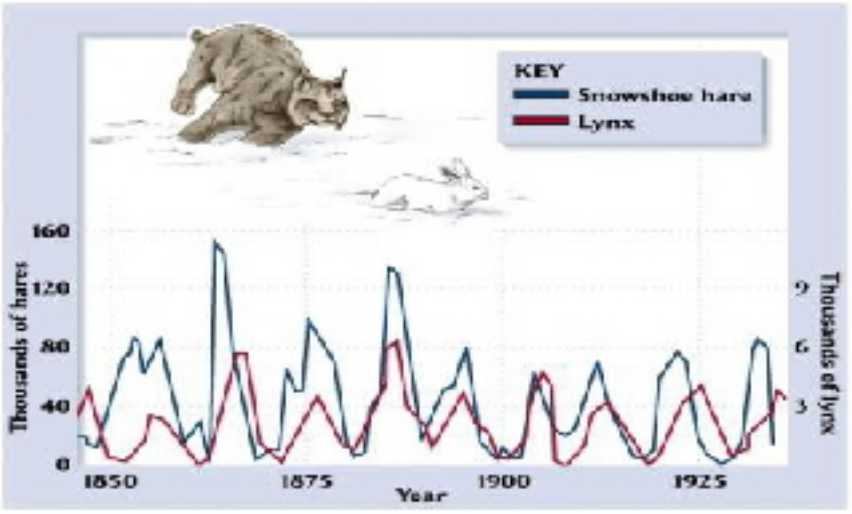


Figure 2.1: Example of Lotka-Volterra model.

As the graphic 2.1 shows, the lynx and the snowshoe hare populations follow globally periodic trajectories, one is being delayed by the other. Is it due to the specific dynamic of the system, or provoked by external causes?. The model of Lotka Volterra choose the specific dynamic of the system as an answer.

The model is based on two populations whose numbers at time t are respectively noted $x(t)$ and $y(t)$, the second (predators) feeding on the first (prey).

The following hypothesis are made :

- The prey $x(t)$ have unlimited food, only the predators $y(t)$ oppose their increasing and in the absence of predators the prey population has an exponential increasing (Malthus law).
- The number of predators is limited by the amount of prey available for feeding and in the absence of prey, the predator population has an exponential decreasing (Malthus law).
- The number of encounters between prey and predators is both proportional to $x(t)$ and $y(t)$ and therefore proportional to the product $x(t) \times y(t)$.
- The rate of prey's decreasing and the rate of predator's increasing due to these encounters are both proportional to the number of encounters between the two populations.

This leads to the following model:

$$\begin{cases} x'(t) = ax - bxy. \\ y'(t) = cxy - dy. \end{cases}$$

Where $a > 0$ is the prey's birth rate, $d > 0$ is the predator's death rate, $b > 0$ and $c > 0$ are interactions coefficients between the two populations. for obvious reasons, we are only interested in this system for positive valuesof x and y.

In addition, a term due to predation is added to the prey's equation, also another term of reproduction in the predator's equation, in order to model the meetings between two species and their evolution.

2.2 Problem of Existence

2.2.1 Local Existence

We have an autonomous differential system

$$\begin{cases} u^{'}(t) = f(u(t)) \\ u(0) = u_0 \end{cases} \tag{2.1}$$

where $f : \Omega \subset \mathbb{R}^n \to \mathbb{R}^n$ and $u_0 \in \Omega$.

Theorem 2.2.1 *(Cauchy-Lipschitz) If f is locally lipschitzian, so it exists a unique maximum solution $u \in C([0, T[, \mathbb{R}^n)$ of the system 2.1 for $t < T$.*

Definition 2.2.1 *We call a trajectory starting with u_0 the set*

$$\tau_{u_0} = \{u(t)/t > 0\}$$

where $u(t)$ is the maximum solution corresponding to the intial condition $u(0) = u_0$.

Cauchy-Lipschitz leads to the following result :

Corollary 2.2.1 *If $u_0 \neq u_1$, then the trajectories τ_{u_0} and τ_{u_1} are distinct.*

2.2.2 Existence and Positivity of Solution

In a mathematic point of vue, the obtained system is a first-order nonlinear autonomous differential system, coupled in x and y. let's put:

$$F : \left| \begin{array}{c} \mathbb{R} \times \mathbb{R}^2 \longmapsto \mathbb{R}^2 \\ (t, x, y) \longmapsto (ax - bxy, cxy - dy) \end{array} \right.$$

F being polynomial in the coordinates, therefore it's of class C^1 in $\mathbb{R}^3$ so it's locally Lipschitzian. The cauchy-Lipschitz theorem assures us that for any initial condition (t_0, x_0, y_0) the Cauchy problem below has a unique maximal solution :

$$\begin{cases} (x^{'}(t), y^{'}(t)) = F(t, x(t), y(t)) \\ (x(t_0), y(t_0)) = (x_0, y_0) \end{cases}.$$

We have another result:

Theorem 2.2.2 (Solution's positivity) *Let x_0, y_0 strictly positive real numbers and t_0 an*

unknown real number. Then, the solution of the Cauchy problem

$$\begin{cases} (x^{'}(t), y^{'}(t)) = F(t, x(t), y(t)) \\ (x(t_0), y(t_0)) = (x_0, y_0) \end{cases}$$

is verified fo all real t $\begin{cases} x(t) > 0 \\ y(t) > 0 \end{cases}$.

To proof this theorem we use the following lemma:

Lemma 2.2.1 *In each of the following cases, the function* (x, y) *thus defined is the only solution to the Cauchy problem:*

$$\begin{cases} (x^{'}(t), y^{'}(t)) = F\left(t, x(t), y(t)\right) \\ (x(t_1),y\left(t_1\right)) = (x_1, y_1) \end{cases}.$$

1. If $x_1 = 0$ and $y_1 = 0$ then for any real t, we define $(x(t), y(t)) = (0, 0)$.
2. If $x_1 = 0$ and $y_1 > 0$ then for any real t, we define $(x(t), y(t)) = (0, y_1 e^{-d(t-t_1)})$.
3. If $x_1 > 0$ and $y_1 = 0$ then for any real t, we define $(x(t), y(t)) = (x_1 e^{a(t-t_1)}, 0)$.

Proof of the lemma. It's easy to verify that the functions are solutions in each case and that they verify the initial conditions. Uniqueness is given by Cauchy-Lipschitz's theorem (these solutions are moreover defined on $\mathbb{R}$ therefore they are global). ■

Proof of the theorem. Using proof by contradiction and assuming that there exists t_1 where $x(t_1) \leq 0$ or $y(t_1) \leq 0$. Rather, let's suppose that there's t_1 where $x(t_1) = 0$ or $y(t_1) = 0$ (i.e. a solution crosses the axis (Ox) or the axis (Oy)). If we ever get a contradiction, the continuity of solutions will prevent their coordinates from becoming negative, knowing that the initial conditions are strictly positive.

- If $x_1 = 0$ and $y_1 = 0$, then (x, y) verifies the Cauchy problem :

$$\begin{cases} (x^{'}(t), y^{'}(t)) = F\left(t, x(t), y(t)\right) \\ (x(t_1),y\left(t_1\right)) = (0, 0) \end{cases},$$

so according to the lemma, $(x(t),y\left(t\right)) = (0, 0)$ for any real t. and this is a contradiction because for $t = t_0$ we have $x(t_0) > 0$.

- If $x_1 = 0$ and $y_1 > 0$, then (x, y) verifies the Cauchy problem:

$$\begin{cases} (x^{'}(t), y^{'}(t)) = F\left(t, x(t), y(t)\right) \\ \quad (x(t_1), y\left(t_1\right)) = (0, y_1) \end{cases},$$

  according to the lemma, $(x(t), y\left(t\right)) = (0, y_1 e^{-d(t-t_1)})$ for any real t, and this is a contradiction too because we have $x(t_0) = 0 > 0$.

- The proof is the same for the last case $x(t_1) > 0$ and $y(t_1) = 0$.

■

The application of Cauchy-Lipschitz theorem gives us the theorem below.

Theorem 2.2.3 *Let the maximum solution u of 2.1 defined for $t < T$. If u is bounded in $[0, T[$, then $T = +\infty$.*

2.2.3 Periodicity

Theorem 2.2.4 (Periodicity of solutions) *Any solution of the Lotka-Volterra system is **periodic**.*

Remark 2.2.1 *This result is sometimes called "Volterra's First Law".*

2.2.4 Equilibrium Points

Proposition 2.2.1 *The points $(0, 0)$ and $(\frac{d}{c}, \frac{a}{b})$ are the only equilibrium points of the Lotka-Volterra system. Therefore the first solution shows that if both populations are extinct, then they will continue being extinct until an external factor can change that. The second solution represents a fixed point (population equilibrium). Which means that levels of population where this equilibrium is achieved depend on the chosen values of the parameters.*

Proof. Let t, x_0, y_0, x and y be real numbers.

$$\begin{aligned} (x, y) \text{ is an equilibrium point} &\iff F(t, x, y) = 0. \\ &\iff x(a - by) = 0 \text{ and } y(cx - d) = 0. \\ &\iff (x, y) = (0, 0) \text{ or } (x, y) = (\frac{d}{c}, \frac{a}{b}). \end{aligned}$$

Let us denote (x_0, y_0) one of these equilibrium points and linearize F in the neighborhood of (x_0, y_0). Taylor's formula gives $F(x, y) \sim DF_{(x_0,y_0)}(x - x_0, y - y_0)$ where DF is the differential of F. The Jacobian matrix of F is then:

$$J(x_0, y_0) = \begin{pmatrix} a - by_0 & -bx_0 \\ cy_0 & cx_0 - d \end{pmatrix}.$$

If $(x_0, y_0) = (0, 0)$, the eigenvalues of J are $a > 0$ and $-d < 0$, hence the results. If $(x_0, y_0) = (\frac{d}{c}, \frac{a}{b})$, the characteristic polynomial of J is $X^2 + ad$, so the eigenvalues (X) of J are therefore pure imaginary and opposite, therefore $(\frac{d}{c}, \frac{a}{b})$ is a center. ■

Remark 2.2.2 *The center point is sometimes called " Volterra's second law".*

It's possible to know the period of solutions in the neighborhood of $(\frac{d}{c}, \frac{a}{b})$, as the next proposition shows:

Proposition 2.2.2 *In the neighborhood of the equilibrium point $(\frac{d}{c}, \frac{a}{b})$, the solutions of the Lotka-Volterra equations are periodic with period $\frac{2\Pi}{\sqrt{ad}}$ and are written in the form:*

$$\forall t \epsilon \mathbb{R}, \qquad \begin{cases} x(t) = \lambda_1 \cos(\sqrt{ad}t) + \mu_1 \sin(\sqrt{ad}t) + \frac{d}{c} \\ y(t) = \lambda_2 \cos(\sqrt{ad}t) + \mu_2 \sin(\sqrt{ad}t) + constante \end{cases}.$$

Proof. In matrix terms, the Lotka-Volterra system in the neighborhood of $(\frac{d}{c}, \frac{a}{b})$ becomes:

$$\begin{pmatrix} x'(t) \\ y'(t) \end{pmatrix} = \begin{pmatrix} 0 & -\frac{bd}{c} \\ \frac{ac}{b} & 0 \end{pmatrix} \begin{pmatrix} x(t) - \frac{d}{c} \\ y(t) - \frac{a}{c} \end{pmatrix}.$$

Therefore it's possible to solve it by substitution. Thus by deriving the first equation and replacing $y_0(t)$ by its value according to the second equation, we find that x satisfies the second order linear differential equation with constant coefficients:

$$f'' + adf = \frac{ad^2}{c}.$$

The characteristic equation has solutions $\pm i\sqrt{ad}$, so the homogeneous equation has solutions $t \mapsto \lambda \cos(\sqrt{ad}t) + \mu \sin(\sqrt{ad}t)$, with λ and μ constants. Since $\frac{d}{c}$ is a particular solution, we

deduce required form for x. We deduce the form of y knowing that:

$$y^{'}(t) = \frac{ac}{b}\left(x(t) - \frac{d}{c}\right) = \frac{ac}{b}\lambda\cos(\sqrt{ad}t) + \mu\sin(\sqrt{ad}t)$$

Then the period is the same for x and y and is equal to $\frac{2\Pi}{\sqrt{ad}}$. ■

Here is another interesting result about the periodicity and the equilibrium point $(\frac{d}{c}, \frac{a}{b})$.

Theorem 2.2.5 *Let (x, y) be a solution of the Lotka-Volterra system whose elements do not cancel. If we note X and Y the average of x and y in order over a period T, then $(X, Y) = \left(\frac{d}{c}, \frac{a}{b}\right)$.*

Proof. Let the real number t_0. x, y doesn't cancel, we can write for every real t:

$$\frac{x^{'}(t)}{x(t)} = a - by(t) \text{ and } \frac{y^{'}(t)}{y(t)} = cx(t) - d.$$

We can integer every equation from t_0 to $t_0 + T$, we remark that $\ln(x(t_0 + T)) - \ln(x(t_0)) = 0$ due to the periodicity of x, same for y, it gives us:

$$0 = aT - bTX \text{ and } 0 = cTY - dT,$$

where we get the result after the simplification by T. ■

Chapter 3

Matlab Resolution of Lotka-Volterra Model

Since we can't solve the Lotka-Volterra system (Prey-Predator model) analytically, we are interested in solving it numerically. So in this chapter we will solve this system using two famous numerical methods called : Euler method and fourth-order Runge kutta method.

Figure 3.1: Prey-Predator example

3.1 Example of Lotka-Volterra Model

In the previous Chapter we saw that the Lotka-Volterra model is of the form

$$\begin{cases} x'(t) = ax - bxy. \\ y'(t) = cxy - dy. \end{cases}$$

Where x is the population of prey, y is the population of predators and the parameters a, b, c and d are biologically determined, In this example we are going to consider a system where $a = 1.2$, $b = 0.6$, $c = 0.3$ and $d = 0.8$. We will take the initial conditions of the system as follows: $x(0) = 2$ and $y(0) = 1$. Which gives us the following predator-prey model:

$$\begin{cases} \frac{dx}{dt} = 1.2x - 0.6xy. \\ \frac{dy}{dt} = 0.3xy - 0.8y. \\ x(0) = 2;\ y(0) = 1. \end{cases}$$

On this example we'll answer these questions:

1. Solve the system for $0 \leq t \leq 30$ using Euler's method (try step sizes of $h = 0.05$ and 0.001).
2. Plot and compare the phase-plane trajectories [i.e., plot $y(t)$ versus $x(t)$].
3. Solve the predator-prey model using ode45.
4. Plot $x(t)$ and $y(t)$ also, generate the phase-plane trajectory for $y(t)$ versus $x(t)$.

3.2 Solving the System Using Euler's Method

To solve the prey-predator's model by Euler method, we first re-label x as $y1$ and y as $y2$. Then, we use ode_eul2 as follows:

3.2.1 Euler's Code for the step size h=0.05

The results obtained by Euler's method for the step size $h = 0.05$ are presented on the following graph

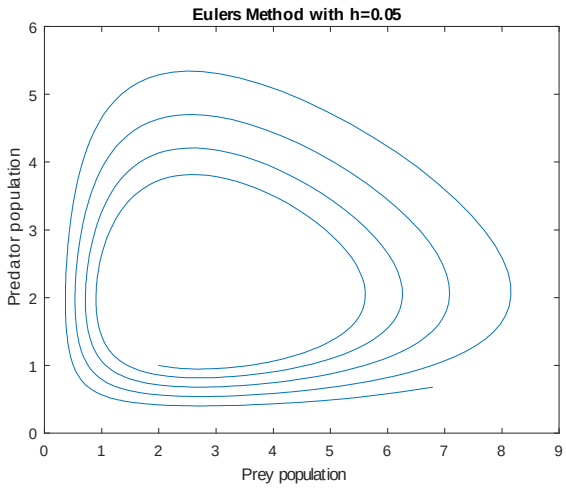


Figure 3.2: Solutions of prey-predator system using Euler's method with step size h=0.05

We see that the method diverges (in a counter-clock-wise direction) for $h = 0.05$ due to excessive global error, as shown in the previous phase-plane plot.

So to obtain a meaningful solution we try to take a smaller step which is $h = 0.001$.

3.2.2 Euler's Code for the step size h=0.001

The resolution of Lotka-Volterra system by Euler's method with the step $h = 0.001$ is gived by

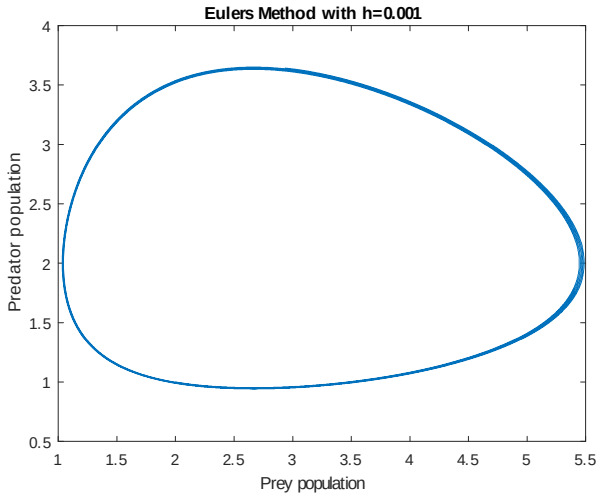


Figure 3.3: Solutions of prey-predator system using Euler's method with step size h=0.001

As the plot 3.3 shows, the method converges to an orbit for $h = 0.001$ (and even for smaller values). What captivates our attention is the eternal periodic change in populations that the model reveals.

3.2.3 The Phase-Plane Trajectories using Euler Method

Now we'll generate the phase-plane trajectory plot for $y(t)$ versus $x(t)$ as follows:

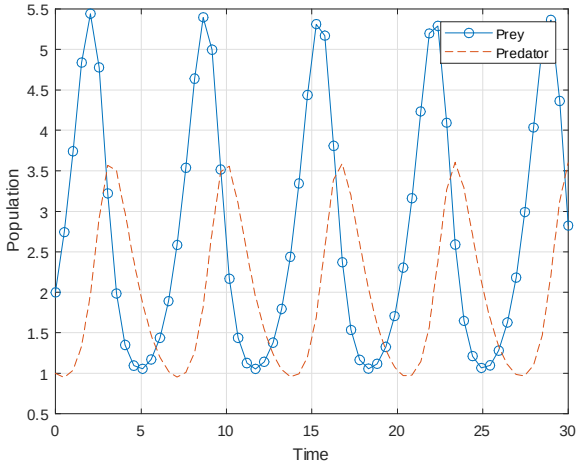


Figure 3.4: Solutions' trajectories with Euler's method

The graph 3.4 shows that the increase of prey drives the growth of predators. However when the population of predators increase, the prey's decreases and reaches a minimum, so that predators also decrease. When predators decrease a new growth of prey happens and so on. This dynamic thus leads to a continuous cycle of growth and decline.

We note that when the predator's population decreases the prey's increases. So we can say that they have a negative relationship.

3.3 Solving the System using Runge-Kutta Fourth-Order Method

3.3.1 Solving the Predater-Prey Model using ode45

ode45 is a particular built in ODE solver on MATLAB, which is based on fourth- and fifth-order Runge-Kutta methods. The function can solve single ordinary differential equation or a system of ordinary differential equations, and the syntax of the basic call is

$$[t, y] = ode45(fun, tspan, y0)$$

where y is the solution array that contains on each column one of the dependent variables, t is the column vector of the corresponding time and the right-hand-side array of the ordinary differential system $f(t, y(t))$ is defined by the user fun. The initial and final values of the desired

solution interval are specified by $tspan = [ti\ tf]$, and $y0$ is the initial values vector. There is no need to specify the value of the step size h in the function call, bacause $ode45$ computes h adaptively, the default value is set to 0.01, but if we need to change the step size h we can use the command 'options'.

The plot obtained on the application of ode45 on this example is shown as follows

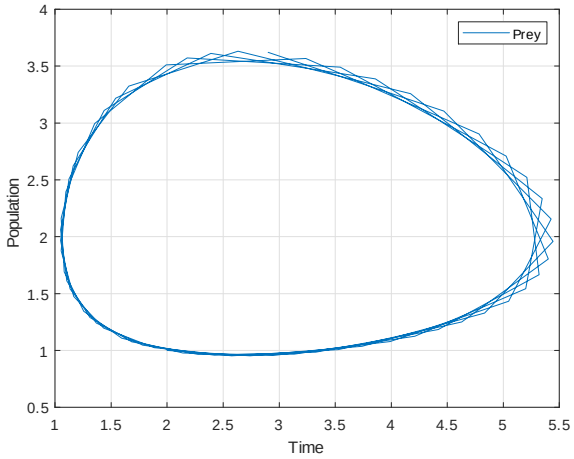


Figure 3.5: Solutions of prey-predator system using RK-4 method (ode45)

To get the previous graph we re-labeled x as $x(1)$ and y as $x(2)$. As we can see in graphic 3.5 the RK4 method gives us better results after a considerable time interval, and the plot can be made smoother if we restrict the size of the largest step size h that the algorithm generates.

3.3.2 The Phase-Plane Trajectories using RK4 Method

Ploting $x(t)$ and $y(t)$ and generating the phase-plane trajectory for $y(t)$ versus $x(t)$

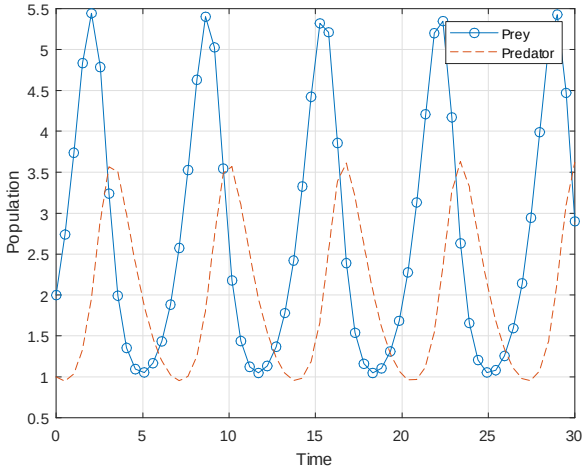


Figure 3.6: Solutions' trajectories with RK4's method

The graph 3.6 shows that the size of predator's population has a negative effect on the prey's population, and the size of the prey's population's has a positive effect on the predator's population.

3.4 Concluding Remarks

In this example, when we used Euler's method for the step $h = 0.05$ the method diverges, but for the step $h = 0.001$ we found that the method converges to an orbit. So in numerical solutions, based on Euler's method, in order to get a meaningful solution, we are advised to use smaller step sizes.

As it's shown in graph 3.4, when the prey population increases, the abundance of food flourishes the predator population. But, as the predator population increases at a rate faster than can be supported by the prey, the prey population decreases. This cycle repeats itself when both populations reach their initial population levels, $x(0)$ and $y(0)$.

It also should be noted that the initial conditions for this model are scaled. For example, if we say $x\,(0) = 1$ and $y(0) = 2$ this means that initial prey and predator populations are of 1000 and 2000, respectively.

Additionally, there is no equilibrium point for this particular dynamical system at which both populations become constant. On other hand some sets of coefficient $\{a, b, c, d\}$, that can lead to a critical equilibrium point where the populations are constant. This equilibrium point can be derived by setting both rates, $y^{'}$ and $x^{'}$, to zeros:

$$\begin{cases} 0 = ax - bxy \\ 0 = -cy + dxy \end{cases}$$

Then, we get the two solutions: $(x, y) = (0, 0)$ and $(x, y) = (\frac{c}{d}, \frac{a}{b})$. Thus, if we start with the trivial case (no population), there will never be predators nor prey. And for the other solution, it implies that if we start the previous model with the initial populations $x(0) = \frac{c}{d} = \frac{0.8}{0.3} = \frac{8}{3}$ and $y(0) = \frac{a}{b} = \frac{1.2}{0.6} = 2$, then the populations will never change.

For the RK4's method we used the particular solver ode45, we found a superior result (in accuracy) to the one obtained using Euler's method. However, one should keep in mind that the slope estimate of RK4 is more elaborate than the simple one used by Euler's method.

General Conclusion

The Lotka-Volterra system is a set of two first-order nonlinear autonomous differential equations that describes the biological phenomenon of feeding relationship between two species (Predator and prey).

On this thesis we solved this system by using two numerical methods: Euler and RK4's method and we found that RK4 is so much better than Euler method because RK4 is a lot more stable and a lot more accurate, in addition they take pretty much the same amount of time to compute, RK4 is a little bit slower but u won't really notice the difference.

Even though Euler and RK4's method gives good results, they suffer of being classique, slow and they are not suitable for conservative systems that takes a lot of time.

As a perspective study, we propose to use new methods like metaheuristic algorithms bacause they are:

- Easy to use in programmation and manipulation.
- Adaptable for every optimization's probelm even the difficult one, so we can apply them in any problem we want to solve.
- More precise then other classic methods (getting better results).
- Fast in optimizing (gain of time).
- Efficient and robust.

Bibliography

[1] Bhargava, S. C. (1989). *Generalized Lotka-Volterra equations and the mechanism of technological substitution. Technological Forecasting and Social Change*, 35(4), 319-326.

[2] Ginoux, J. M. (2017). *The paradox of Vito Volterra's predator-prey model. Lettera Matematica*, 5(4), 305-311.

[3] Hassoun, M. (2020). *Numerical Solution of Differential Equations.* https://docplayer.net/187416096-Ece-3040-lecture-22-numerical-solution-of-differential-equations-prof-mohamad-hassoun.html

[4] Li, L. and Li, W. T. & Wang, M. (2022). *Dynamics for nonlocal diffusion problems with a free boundary. Journal of Differential Equations*, 330, 110-149.

[5] Madjidi, G. and Rahal, I. (2019). *Système D'équations Lotka-Volterra et Modèle Prédateur-Proie. Mem. Acad, Université Hamma Lakhdar D'El Oued, Alger.*

[6] Dawkins, P. (2007). *Differential Equation,* http://tutorial.math.lamar.edu/terms.aspx.

[7] Sizemore, J. and Paul Mueller, J. (2015). *MATLAB for Dummies*, John Wiley & Sons, Inc.Hoboken, New Jersey.

[8] Tribut, K. and Dyron, Y. (2013). *Modèle proi-pédateur,* Dossier de Modélisation.

[9] Volterra, V. (1926). *Variazioni e uttuazioni del numero d'individui in specie animali conviventi,* Mem. Acad. Lincei 3, 6 , 31-113.

[10] Volterra, V. (1962). *Fluctuations in the abundance of a species considered mathematically,* Nature, 118, 558-560 , (1927). Sotto lo stesso titolo furono poi pubblicate due lettere, una del LOTKA e una del VOLTERRA, Ibidem,119, 12-13.

[11] Volterra, V. (1928), *Variation and uctuations of the number of individuals in animal species living together*. Translated by Miss MARY EVELYN WELLS, Journal du Conseil international l'exploration de la mer, Copenhague, 3, n. 1, 3-51.

[12] Volterra, V. (1931), *Leçons sur la Théorie Mathématique de la Lutte pour la Vie*, Gauthier-Villars, Paris.

[13] Zhu, C., & Yin, G. (2009). *On hybrid competitive Lotka–Volterra ecosystems. Nonlinear Analysis: Theory, Methods & Applications*, 71(12), e1370-e1379.

Appendix A: MATLAB

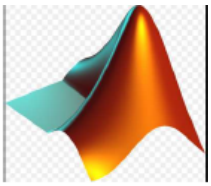


Figure 3.7: MATLAB icon

MATLAB is a programming language that built with a working environment. This includes facilities to manage the variables for importing and exporting data of work source. **MATLAB** also includes many supporting features like tools for developing and managing M-files based on the **MATLAB** applications depend on specific platform. The **MATLAB** can run on Microsoft Windows information or UNIX information.

We use **MATLAB** as more than a calculator to create a program that helps you perform tasks consistently, easily and quickly. **MATLAB** has a rich and large toolbox for exploring science, technology, engineering and mathematics (STEM) that includes:

- Statistics
- Simulation
- Image processing
- Simbolic processing
- Numerical analysis.

So why you need **MATLAB**?

- Relying on structure for better organization : writing programs is all about telling the computer to perform a task one step at a time. The better your language tells the computer

what to do, the easier the computer will be to use and the less time you'll spend getting it to preform a given task.

- Avoiding the complexity of object-oriented programming (OOP) : it's a discipline that helps developers create applications based on real-world models. Every element of an application becomes an object that has specific characteristics and can perform specific tasks. This technology is quite useful to developers because it helps the create extremely complex applications with fewer errors and less coding time.

- Using the powerful toolbox: here is just a small sample of the areas that are addressed by the tools you find in the **MATLAB** toolbox: algebra, linear algebra, calculus, differential equations, statistics, curve fitting, graphing and preparing reports.

- Reducing programming effort with the fourth-generation language: **MATLAB** employs a fourth-generation language to make your job a lot easier. The language isn't quite human, but it's also a long way away from the machine code that developers used to write to make computers work. Using **MATLAB** makes you more efficient because the language is specifically designed to meet the needs of STEM users

Appendix B: Abreviations and Notations

Les différentes abbréviations et notations utilisées durant cette thèse est représentées selon ce tableau:

Abbreviations	Notations
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PVI	Initial Value Problem.
$RK-4$	Runge-Kutta fourth order
$\mathbb{N}$	Set of natural numbers
$\mathbb{R}$	Set of real numbers
STEM	Science Technology Engineering and Mathematics
OOP	Object-Oriented Programming

Appendix C : MATLAB's Code Used

3.5 Euler's Code

3.5.1 Euler's Code for the step size h=0.05

```
f1=@(t,y1,y2)1.2*y1-0.6*y1*y2;
f2=@(t,y1,y2)-0.8*y2+0.3*y1*y2;
[t,y1,y2]=ode_eul2(f1,f2,[0 30],2,1,0.05);
plot(y1,y2)
xlabel('Prey population')
ylabel('Predator population')
title('Eulers Method withh=0.05')
function[t,y1,y2]=ode_eul2(f1,f2,tspan,y10,y20,h)
ti=tspan(1);tf=tspan(2);
t=(ti:h:tf)';n=length(t);
y1=zeros(1,n)';
y2=zeros(1,n)';
y1(1)=y10;
y2(1)=y20;
for k=1:n-1
y1(k+1)=y1(k)+h*f1(t(k),y1(k),y2(k));
y2(k+1)=y2(k)+h*f2(t(k),y1(k),y2(k));
end
end
```

3.5.2 Euler's Code for the step size h=0.001

```
f1=@(t,y1,y2)1.2*y1-0.6*y1*y2;
f2=@(t,y1,y2)-0.8*y2+0.3*y1*y2;
[t,y1,y2]=ode_eul2(f1,f2,[0 30],2,1,0.001);
plot(y1,y2)
xlabel('Prey population')
ylabel('Predator population')
title('Eulers Method with h=0.001')
function[t,y1,y2]=ode_eul2(f1,f2,tspan,y10,y20,h)
ti=tspan(1) ; tf=tspan(2);
t=(ti:h:tf)';n=length(t);
y1=zeros(1,n)';
y2=zeros(1,n)';
y1(1)=y10;
y2(1)=y20;
for k=1:n-1
y1(k+1)=y1(k)+h*f1(t(k),y1(k),y2(k));
y2(k+1)=y2(k)+h*f2(t(k),y1(k),y2(k));
end
end
```

3.5.3 The Phase-Plane Trajectories

```
%Predator-prey Model
clc; clear;
y0=[2;1];
soln=ode23(@f2,[0 30],y0)
t=linspace(0,30,60);
y(:,1)=deval(soln,t,1);%Prey
y(:,2)=deval(soln,t,2);%Predator
figure
plot(t,y(:,1),'-o',t,y(:,2),'--');
```

```
hold on ; grid on;
legend('Prey','Predator');
xlabel('Time');
ylabel('Population');
hold off;
%Predator-prey function
function dxdt=f2(t,x)
dxdt=[0;0];
p=1.2;q=0.6;r=0.3;s=0.8;
dxdt(1)=p*x(1)-q*x(1)*x(2);
dxdt(2)=r*x(1)*x(2)-s*x(2);
end
```

3.6 RK4's Code

3.6.1 Solving the Predater-Prey Model using ode45

```
%Pedator-prey Model
clc;clear;
y0=[2;1];
soln=ode45(@f2,[0 30],y0)
t=linspace(0,30,60);
y(:,1)=deval(soln,t,1); %Prey
y(:,2)=deval(soln,t,2); %Predator
figure
plot(y(:,1),y(:,2));
hold on;grid on;
legend('Prey','Predator');
xlabel('Time');
ylabel('Population');
hold off;
%Predator-prey function
function dxdt=f2(t,x)
```

```
dxdt=[0;0];
p=1.2;q=0.6;r=0.3;s=0.8;
dxdt(1)=p*x(1)-q*x(1)*x(2);
dxdt(2)=r*x(1)*x(2)-s*x(2);
end
```

3.6.2 The Phase-Plane Trajectories:

```
%Pedator-prey Model
clc;clear;
y0=[2;1];
soln = ode45(@f2,[0 30],y0)
t= linspace(0,30,60);
y(:,1)=deval(soln,t,1); %Prey
y(:,2)=deval(soln,t,2); %Predator
figure
plot (t,y(:,1),'-o',t,y(:,2),'--');
hold on; grid on;
legend('Prey,'Predator');
xlabel('Time');
ylabel('Population');
hold off;
%Predator-prey function
function dxdt=f2(t,x)
dxdt=[0;0];
p=1.2; q=0.6; r=0.3; s=0.8;
dxdt(1)=p*x(1)-q*x(1)*x(2);
dxdt(2)=r*x(1)*x(2)-s*x(2);
end
```

Abstract

This thesis studies a system of non-linear ordinary differential equations, which describes : The ecological phenomenon Prey-Predator, the relation between them, there increasing and decreasing. It's called the Lotka-Volterra model.

Since we can't solve the problem analytically we used classical numerical methods. In the practical part we focus on the application of two popular methods : Euler and Runge-Kutta fourth order, using numeric computing platform MATLAB. The findings are periodic trajectories with two equilibrium points.

Résumer

Ce mémoire étudie un système d'équation différentielle ordinaire non-linéaire, qui décrit : La phénomène écologique proi-prédateur, la relation entre eux, leurs croissances et décroissances. On l'appelle le modèle de Lotka-Volterra.

Puisqu'on peut pas résoudre ce problème analytiquement on recours vers des méthodes numériques classiques. Dans la partie pratique on concentre, à l'aide du logiciel MATLAB, sur l'application de deux méthodes populaires : Euler et Runge-kutta d'orde 4. Les solutions trouvées sont des trajectoires periodiques avec deux points d'équilibre.

ملخص

تدرس هذه الأطروحة نظام المعادلات التفاضلية العادية غير الخطية، والتي تصف: الظاهرة البيئية فريسة-مفترس، العلاقة بينهما ، تزايدهما وتناقصهما. يطلق عليه بـ: نموذج Lotka-Volterra.

نظرًا لعدم قدرتنا على حل هذه المشكلة تحليليًا ، فإننا نلجأ إلى الأساليب العددية الكلاسيكية. في الجزء العملي نركز و باستخدام برنامج MATLAB ، على تطبيق طريقتين شائعتين: Euler و Runge-kutta 4. الحلول الموجودة هي مسارات دورية مع وجود نقطتي توازن.