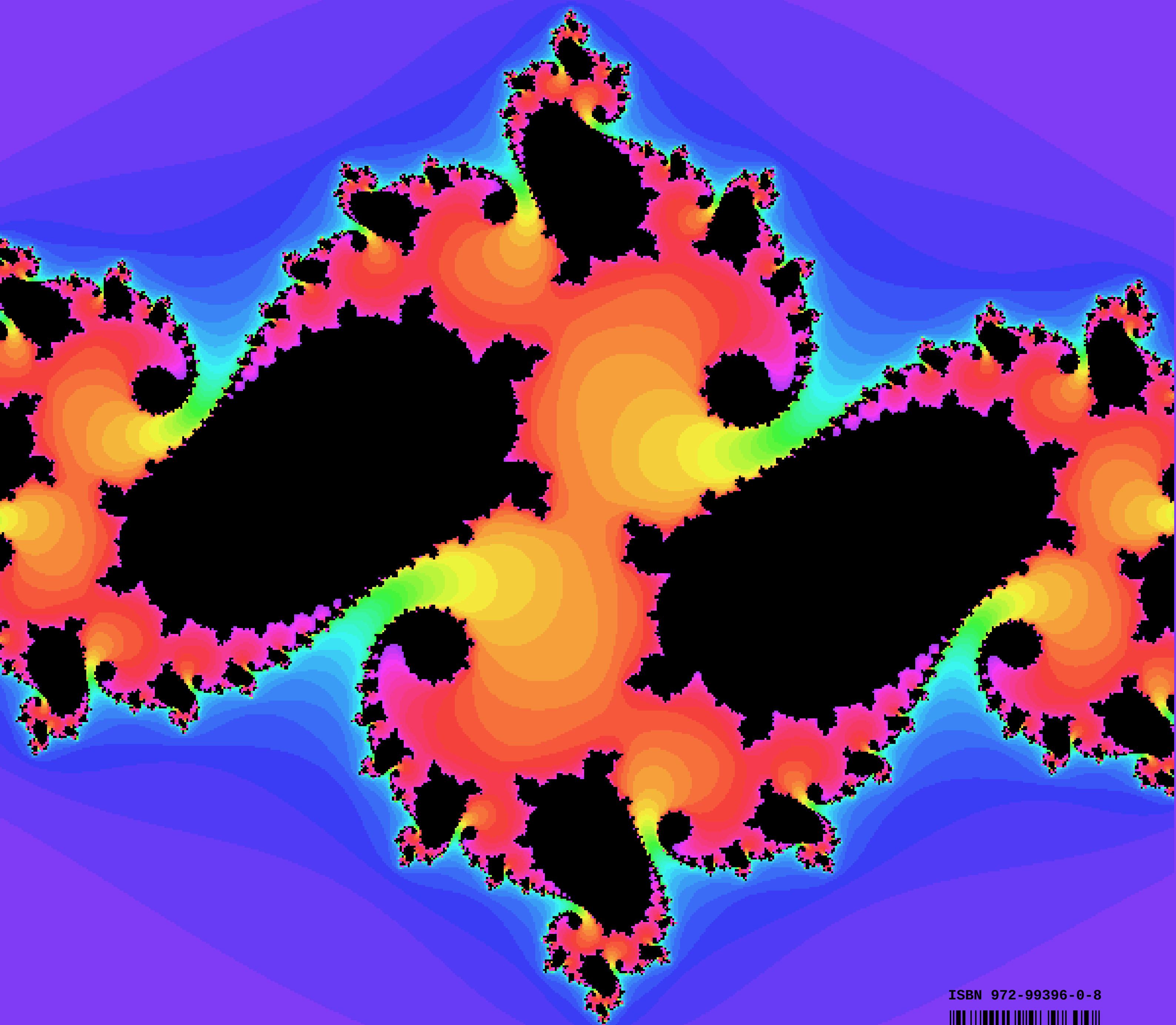
Introduction to Dynamical Systems

A HANDS-ON APPROACH WITH MAXIMA



ISBN 972-99396-0-8

Jaime E. Villate

Introduction to Dynamical Systems

A Hands-on Approach with Maxima

Jaime E. Villate
College of Engineering
University of Porto, Portugal

Introduction to dynamical systems: a hands-on approach with Maxima

E-mail: villate@fe.up.pt

ISBN: 972-99396-0-8

This is a partial translation of Portuguese version 1.1 of November 23, 2006

The cover figure is the Julia set for the complex number $-0.75+i\,0.1$, with 48 iterations, as explained in chapter 12.

Contents

List of Figures

Preface

This book is updated very often. The number of the current version can be found on the second page and the most recent version can always be found on the Web at http://fisica.fe.up.pt/maxima/dynamicalsystems. This version has been written to be used with Maxima's version 5.10. (http://maxima.sourceforge.net).

This book started as the lecture notes for a one-semester course on the physics of dynamical systems, taught in the period 2003-2006 at the College of Engineering of the University of Porto. The field of dynamical systems is at the borderline of physics, mathematics and computing. Being a scientific field that is not taught with the traditional axiomatic method used in other physics and mathematics courses, it is appropriate to use a practical teaching approach based on projects done with a computer.

The study of dynamical systems advanced very quickly in the decades of 1960 and 1970, giving rise to a whole new area of research with an innovative methodology that gave rise to heated debates within the scientific community. The innovative impulse was ignited by the rapid development of computers.

A new generation of researchers was born, who used their computers as laboratories for experimenting with equations discovering new phenomena. The traditional mathematics criticized that approach for the lack of a rigorous mathematical theory that explained those new results. Many of those results were found within the framework of physical problems: non-linear dynamics, condensed-matter physics and electromagnetism. However, many physicists regarded that new research field simply as a computer simulation of old physical concepts long established, without any new physics in it. A usual comment would be: "this is all very nice, but where is the physics?".

Thus, the pioneers in this new field of dynamical systems would be often confronted with rejected publications in scientific journals and negative assessments of their work. On the other hand, their activities awakened a strong interest that grew up exponentially and was viewed by some as a fresh air current; the methods used in the study of dynamical systems match well with the working environment of modern scientists.

The new paradigm spread to teaching and the traditional courses on physics and mathematics have been gradually "infected" with this new experimental/ computational methodology, contrasting with the traditional axiomatic method. As it happened in the scientific community, the new methodology has also led to some debate among teachers; at the same time, it has awakened big interest as a better way to motivate today's students. Subjects such as chaos and fractals are very appealing to students.

In this book we intend to explore some topics on dynamical systems, using an active teaching approach, supported by computing tools and trying to avoid too may abstract details. The use of a

Computer Algebra System (CAS) does not eliminate the need for mathematical analysis from the student; using a CAS to teach an engineering course does not turn it into a purely technical subject either. One of the difficulties inherent to any Computer Algebra System is the fact that there are no unique solutions to the problems they solve. Different methods to solve a problem may lead to solutions that look very different but might be equivalent. Or the solutions can be really different that are only equivalent within some domain. In some cases the system does not give any answer or it might even give a wrong answer.

It is necessary to gain some experience to be able to use CAS tools successfully and to be able to test the validity of its results. In the process of gaining that experience, the user will also gain a better insight into the mathematical methods used bay the system.

Nowadays the great majority of engineering and exact sciences professionals depend on a calculator to calculate the square root of a real number, for instance, 3456. Some of us were taught in School how to do that with pencil an paper, in a time when there were no calculators. I do not believe that this new dependence is a serious handicap, and I'm not in favor of teaching kids how to calculate square roots with paper and pencil before they are allowed to use the calculator. What I find very important is that the algorithm we used to calculate square roots with paper and pencil remains available and well documented in the literature; it is an important piece in our legacy of algorithms.

On the other hand, now that students have calculators to compute square roots, they can move faster into other topics such as the study of quadratic equations; and in doing so, they might even gain a deeper insight of the function $\sqrt{x}$, which they did not attain when they had to spend a lot of time learning the algorithm to calculate square roots. In the case of differential equations and difference equations, with the help of Computer Algebra Systems students can advance faster into subjects such as chaos and fractals, instead of dedicating a whole semester to learn several algorithms to obtain analytical solutions for a few types of equations.

I would like to acknowledge the help of my colleagues Helena Braga and Francisco Salzedas, with whom I have taught the course on Physics of Dynamical Systems; I would also like to thank our students in that course throughout the last 3 years; their positive comments have encouraged me to undertake the task of writing this book. The students have been asked to make projects for that course, and some of those projects were very interesting and helped me learn some of the subjects covered in this book. Special thanks go to the student Pedro Martins who made a careful review of the manuscript.

Jaime E. Villate
Porto, November 2006

Chapter 1

Introduction

1.1 Differential equations

Differential equations play a very important role in Engineering and Science. Many problems lead to one or several differential equations that must be solved. Most attention has been given to linear equations in the literature; several analytical methods have been developed to solve that type of equations.

Non-linear differential equations are much harder to analyze and there are no general solution techniques for those equations. Problems that lead to linear equations are easier to study. From the last half of the 20th century, the rapid development of the computer made it possible to solve non-linear problems using numerical methods. Non-linear systems lead to a wealth of new and interesting phenomena not present in linear systems.

A new approach, that relies more on geometric interpretation rather than analytical analysis, has gained popularity for the study of non-linear systems. Many of the concepts used in that geometrical approach, such as the phase space, have long be used in dynamics to study the motion of a mechanical system.

In order to give a short introduction to that methodology to study differential equations, in the next chapters we will consider several problems specific dynamics and electrical circuit theory. Before we begin, we will introduce a Computer Algebra System (CAS), Maxima, which will be used extensively throughout the book.

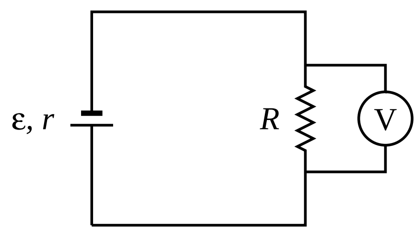
1.2 Solving physics problems with Maxima

Maxima is a software package in the category of Computer Algebra Systems (CAS), namely, a system that can be used not just for numerical calculation but also to deal with algebraic equations with abstract variables. There are various CAS packages available; we have decided to use Maxima because it is Free Software. That means that it can be installed and used by our students without having to obtain a license for it, and they can even study its source code to get a better understanding of how that system works. Another important advantage is the possibility of modifying the original package to better suit our needs; we took advantage of that facility to add new features needed for this book.

Maxima includes several functions to manipulate mathematical functions, including differentiation, integration, power series approximation, Laplace transforms, solving ordinary differential equations and graph plotting in 2 and 3 dimensions. It can also work with matrices and vectors. Maxima can be used to solve problems numerically and write down programs as done with traditional programming languages.

The following examples should serve to give a first glimpse at the way Maxima can be used. In the next chapters we will go deeper into the subject, but readers who are not familiar with Maxima and would like to have a general overview from the beginning can start by going through appendix A. The examples that we will solve in this section are in the area of dynamics of a particle and direct-current circuits, which are the main subjects in this book. A minimum knowledge in those two subjects will be necessary in order to follow those examples.

Example 1.1
A battery is connected to an external resistor with resistance R and the voltage across the resistor is measured with a voltmeter V. To find the electro-motive force ε and the internal resistance r of the battery, two external resistors of 1.13 kΩ and 17.4 kΩ were used. The voltage drop in both cases were 6.26 V e 6.28 V. Find the intensity of the current in both cases. Obtain the values of ε and r. Plot a graph of the power dissipated in the external resistance, as a function of R, for values of R within 0 and $5r$.



Solution: The current through R is found from **Ohm's law**:

$$I = \frac{\Delta V}{R} \tag{1.1}$$

With the values given for the potential difference, ΔV, and the resistance, R, we can use Maxima to find the currents:

```
(%i1) 6.26/1.13e3;
(%o1)                         .005539823008849558
```

When Maxima's console is started, the `(%i1)` label appears indicating that the system is ready to accept a command; the letter "i" stands for *input*. The expression 1.13e3 is the form used to represent the number 1.13×10^3 in Maxima. Each command must end with a semi-colon. When the "Enter" key is pressed, the system responds with a label `(%o1)` followed by the result of the first command `(%i1)`; "o" stands for *output*.

The current in the second case is computed in a similar way:

```
(%i2) 6.28/17.4e3;
(%o2)                        3.609195402298851E-4
```

Thus, the current in the 1.13 kΩ resistor is 5.54 mA, and in the 17.4 kΩ resistor is 0.361 mA.

To obtain the battery's electro-motive force and internal resistance we should use the **voltage-current characteristic** for a battery:

$$\Delta V = \varepsilon - rI \tag{1.2}$$

replacing the two set of values given for ΔV and R we will get a system of two equations with two variables. We will save those two equations in two Maxima variables that we will dub as `eq1` and `eq2`

```
(%i3) eq1: 6.26 = emf - r*%o1;
(%o3)                 6.26 = emf - .005539823008849558 r
(%i4) eq2: 6.28 = emf - r*%o2;
(%o4)                 6.28 = emf - 3.609195402298851E-4 r
```

notice that the symbol used to save a value in a variable is a colon and not an equal sign. A maxima variable can store a numerical value or something more abstract as a mathematical equation in this case. The equal sign makes part of the equation that is being stored. To avoid having to write the numerical values of the currents obtained previously, we used the symbols `%o1` and `%o2` that stand for the value of those previous results.

The last two equations constitute a linear system of equations with two variables. That kind of system can be solved in Maxima, using the command `solve`:

```
(%i5) solve([eq1,eq2]);
                               983100        79952407
(%o5)                     [[r = ------, emf = --------]]
                               254569        12728450
(%i6) %,numer;
(%o6)          [[r = 3.861821352953423, emf = 6.281393806787158]]
```

The syntax `[eq1,eq2]` was used to create a **list** with two elements, which is what the command `solve` expects when there are more than one equation to be solved. Some warning messages given by Maxima were omitted above. The command `solve` gives an exact result, in the form of two rational numbers. The command in `%i6` was used to approximate those rational numbers with fixed-point numbers. The symbol `%` stands for the output of the last command executed; in this case it is equivalent to `%o5`. We thus conclude that the electro-motive force is approximately 6.2814 V and the internal resistance is 3.8618 Ω.

The electric power dissipated in the resistance R is

$$P = RI^2$$

the current I across the external resistor can be calculated in terms of the electro-motive force and the resistances r and R

$$I = \frac{\varepsilon}{R + r}$$

therefore, the power dissipated in the external resistor is

$$P = R\left(\frac{\varepsilon}{R+r}\right)^2$$

to draw the plot of P, as a function of R, we can use the following command

```
(%i7) plot2d(R*(6.2814/(R+3.8618))^2, [R, 0, 5*3.8618]);
```

the result is shown in figure 1.1. Moving the cursor in the graphic window, it is possible to read the coordinates of the point where the cursor is. We can check that the power dissipated in an external resistor reaches its maximum value when the external resistance equals the internal one.

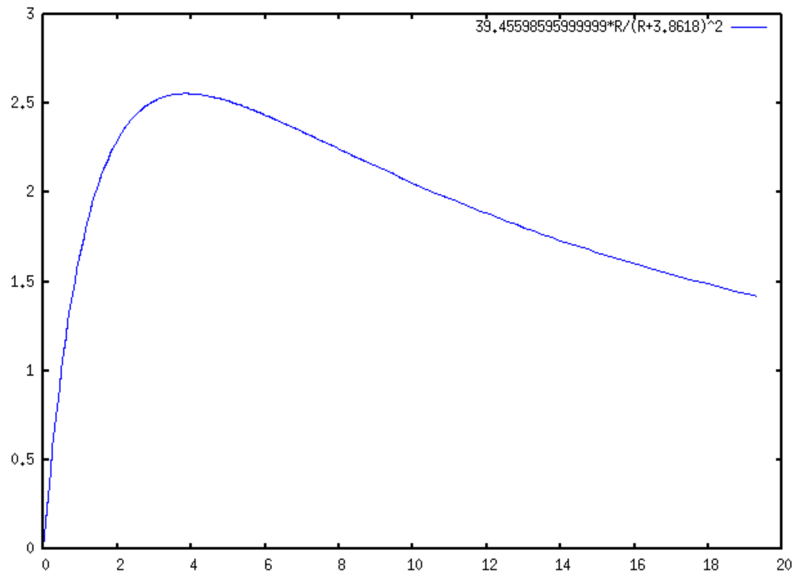


Figure 1.1: Power dissipated in the external resistor as a function of the external resistance.

Two frequent errors should be avoided:

- An expression such as

  ```
  a = 3;
  ```

  will not assign any value to the variable a. After that expression is written, variable `a` remains as an undefined variable. To assign a value of 3 to the variable `a`, the following command must be used:

  ```
  a : 3;
  ```

- New users of Maxima should become aware of the distinction between equations and expressions. An example of an equation is

```
x^2 - 3*x = 2*x + 5
```

while an expression is something like

```
2*x + 5
```

Some commands in Maxima accept only equations or expressions as their input values. For instance, the command `plot2d` used in the previous example accepts only expressions and not equations. The command `solve` requires one equation, or a list with several equations, but it will also accept expressions instead of equations: each expression given will be automatically converted into an equation by equating it to zero; for instance, the command

```
solve(x^2 - 5*x + 5);
```

will find the two values of x that will solve the equation

$$x^2 - 5*x + 5 = 0$$

Example 1.2
The position vector of a particle, as a function of time t, is given by the equation:

$$\vec{r} = \left(5 - t^2\, e^{-t/5}\right)\vec{e}_x + \left(3 - e^{-t/12}\right)\vec{e}_y$$

in MKS units. Find the vectors for the position, velocity and acceleration at $t = 0$, $t = 15$ *s, and when time approaches infinity. Plot the trajectory of the particle during the first 60 seconds of its motion.*

Solution: We start by representing the position vector as a list with two elements; the first element will be the x coordinate and the second one will be the y coordinate. We will save that list in a variable named r, so we can use it later on.

```
(%i8) r: [5-t^2*exp(-t/5),3-exp(-t/12)];
                                 2   - t/5         - t/12
(%o8)                      [5 - t  %e     , 3 - %e      ]
```

the vector velocity equals the derivative of the position vector and the vector acceleration is the derivative of the vector velocity. The command used in Maxima to find the derivative of an expression is `diff`. The input to that command can also be a list with several expressions; thus, the velocity and acceleration are:

```
(%i9) v: diff(r,t);
                         2   - t/5                     - t/12
                        t  %e               - t/5   %e
(%o9)                  [---------- - 2 t %e     , --------]
                            5                        12
(%i10) a: diff(v,t);
```

```
                  2   - t/5          - t/5                   - t/12
                 t  %e         4 t %e              - t/5    %e
(%o10)         [- ---------- + ----------- - 2 %e     , - --------]
                     25            5                        144
```

the constant `%e` in Maxima represents the Euler number, *e*. To find the position, velocity and acceleration at $t=0$, we use the following commands

```
(%i11) r, t=0, numer;
(%o11)                              [5, 2]
(%i12) v, t=0, numer;
(%o12)                      [0, .08333333333333333]
(%i13) a, t=0, numer;
(%o13)                   [- 2, - .006944444444444444]
```

The argument `numer`, was used to get the result in floating-point form. In vector notation, the results we obtained are:

$$\begin{aligned}\vec{r}(0) &= 5\vec{e}x+2\vec{e}_y\\ \vec{v}(0) &= 0.08333\vec{e}_y\\ \vec{a}(0) &= -2\vec{e}x-0.006944\vec{e}_y\end{aligned}$$

For $t=15$ s the results are obtained in a similar way

```
(%i14) r, t=15, numer;
(%o14)              [- 6.202090382769388, 2.71349520313981]
(%i15) v, t=15, numer;
(%o15)              [.7468060255179592, .02387539973834917]
(%i16) a, t=15, numer;
(%o16)                  [0.0497870683678639, - .001989616644862431]
```

The limiting values when times goes to infinity can be calculated with Maxima's command `limit`; the symbol used in Maxima to represent infinity is `inf`

```
(%i17) limit(r,t,inf);
(%o17)                              [5, 3]
(%i18) limit(v,t,inf);
(%o18)                              [0, 0]
(%i19) limit(a,t,inf);
(%o19)                              [0, 0]
```

Thus, a particle will approach the point $5\vec{e}_x+3\vec{e}_y$, where it will remain at rest.

To plot the graph of the trajectory we will use the option `parametric` of the command `plot2d`. the *x* and *y* components of the position vector will be given separately; the command `plot2d` will not accept them inside a list as we have been using them. To get the first element of the list `r` (*x* component) is labelled as `r[1]` and the second element `r[2]`.

```
(%i20) plot2d([parametric,r[1],r[2],[t,0,60],[nticks,100]]);
```

The time domain, from 0 to 60, is defined with the notation `[t,0,60]`. The option `nticks` was used to increase the number of intervals of t, because the default value of 10 intervals would render a broken curve instead of a continuous trajectory. The graph obtained is shown in figure 1.2.

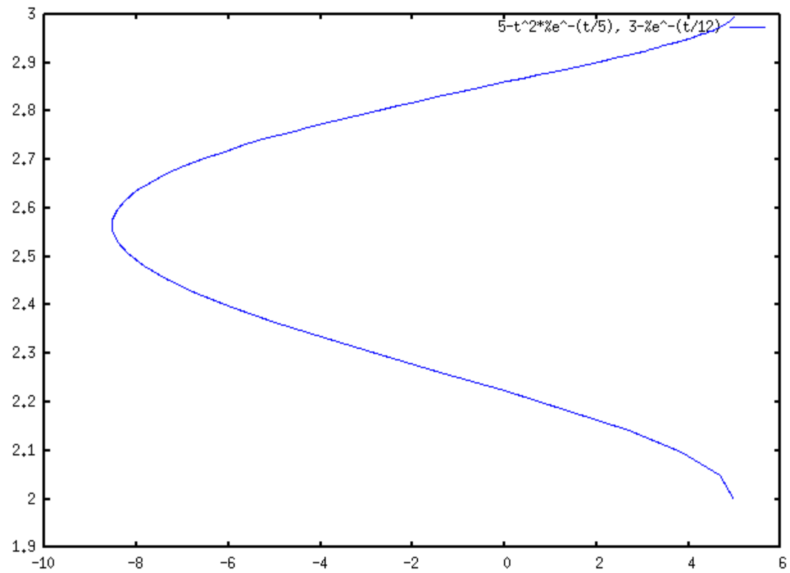


Figure 1.2: Trajectory of the particle during the first 60 seconds, from the initial instant when t was equal to 5.2 .

1.3 References

For more information about Maxima, see appendix A and the Maxima Book (de Souza et al., 2003).

1.4 Multiple-choice questions

1. Only one of the following Maxima commands is correct. Which one?

A. `solve(t-6=0,u-2=0,[t,u]);`

B. `solve(t+4=0,u-4=0,t,u);`

C. `solve([x^3+4=2,y-4],x,y);`

D. `solve(x-6=0,y-2=0,[x,y]);`

E. `solve([t+3,u-4],[t,u]);`

2. Newton's second law was defined in Maxima with:

```
(%i6) F = ma;
```

which Maxima command should be used to compute the value of the force corresponding to a mass of 7 with an acceleration of 5 (SI units).

A. `solve(F, m=7, a=5);`

B. `solve(F, [m=7, a=5]);`

C. `solve(%o6, m=7, a=5);`

D. `%o6, m=7, a=5;`

E. `solve(F: m=7, a=5)`

3. If we input the following commands in Maxima:

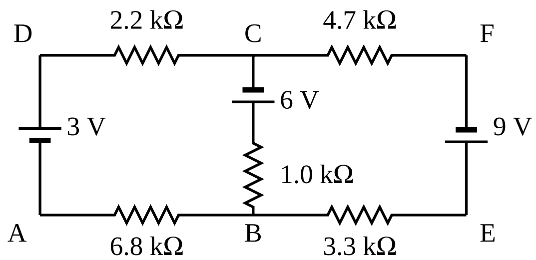
```
(%i1) x:3$
(%i2) x=5$
(%i3) x;
```

which will be the output (%o3)?

A. 5
B. x
C. 3
D. true
E. 0

1.5 Problems

1. An ammeter was used to measure the current at points D and F in the circuit which diagram is shown in the figure. At point D the value obtained for the current was 0.944 mA, in the direction ADC, and at point F it was 0.438 mA, in the direction CFE. (*a*) Store the equation for Ohm's law, $V = IR$, in a Maxima's variable "ohm". (*b*) Give a value to variable I equal to the current at point D, and substitute the resistance in Ohm's law with each of the values 2.2 kΩ and 6.8 kΩ, to compute the potential difference in each of the resistors; repeat the same procedure to calculate the potential difference in each resistor.



2. The position of a particle moving along the x axis is given by the equation $x = 2.5\,t^3 - 62\,t^2 + 10.3\,t$, where x is measured in meters and t in seconds. (*a*) Find the expressions for the velocity and the acceleration as a function of time. (*b*) Find the values of the time, position and acceleration in all the instants when the particle is at rest ($v = 0$). (*c*) Draw the plots for the position, velocity and acceleration as a function of time, for t in the interval between 0 and 20 seconds.

3. The position vector of a particle, as a function of time, is given by the equation:

$$\vec{r} = \left(5.76 - \mathrm{e}^{-t/2.51}\right)\vec{e}_x + \mathrm{e}^{-t/2.51}\cos(3.4\,t)\,\vec{e}_y$$

in SI units. (*a*) Compute the position, velocity and acceleration in the instants t=0, t=8 s, and when time goes to infinity. (*b*) Plot the graphs of the x and y components of the position, as a function of time, for t in the interval between 0 and 15 seconds. (*c*) Plot the graph of the trajectory, on the plane xy, in the interval of t between 0 and 15 s.

Chapter 2

Discrete dynamical systems

A discrete dynamical system is a system which state only changes during a discrete sequence of instants $\{t_0, t_1, t_2, \ldots\}$. In the interval between two of those instants the state remains constant.

In this chapter we will consider only discrete systems in one dimension as an introduction to the study of continuous dynamical systems in the next chapters. We will come back to the subject of second-order discrete systems in a later chapter. The state of a one-dimensional discrete system is defined by a variable y. The values of that variable during the instants $\{t_0, t_1, t_2, \ldots\}$ will be a sequence $\{y_0, y_1, y_2, \ldots\}$ (that sequence is dubbed an **orbit** of the system). The interval of time between different pairs of consecutive instants t_n and t_{n+1} does not have to be the same.

The **evolution equation** allows us to compute the state y_{n+1}, at an instant t_{n+1}, from the value of the state y_n at the previous instant t_n:

$$y_{n+1} = F(y_n) \tag{2.1}$$

where $F(y)$ is some known function. The equation above is a **difference equation** of the first order. Given an initial state y_0, successive application of the function F will generate the sequence of states y_n which determine the evolution of the system. In some cases it might be possible to find a general expression for y_n as a function of n.

Example 2.1

Fond the first four terms in the evolution of the system $x_{n+1} = \cos x_n$, *with initial state* $x_0 = 2$

Solution: Applying the difference equation three times, we obtain the first four terms in the sequence:

$$\{2,\ \cos(2),\ \cos(\cos(2)),\ \cos(\cos(\cos(2)))\} \tag{2.2}$$

Example 2.2

A loan of \$ 500 is obtained from a bank, which charges a 5% yearly interest rate. The loan is to be paid in 20 months, with monthly payments of \$ 26.11. What will be the amount in debt after 10 months?

Solution: During the month n the amount in debt, y, will be equal to the amount in debt in the previous month, y_{n-1}, plus the interests due for that month, minus the monthly payment p:

$$y_n = y_{n-1} + j\,y_{n-1} - p \tag{2.3}$$

where j is the monthly interest rate (0.05/12). Using Maxima, the sequence of amounts in debt y_n can be obtained by applying the above recurrence relation several times:

```
(%i1) j: 0.05/12$

(%i2) y: 500$

(%i3) y: y + j*y - 26.11;

(%o3)                         475.9733333333333
(%i4) y: y + j*y - 26.11;

(%o4)                         451.8465555555555
(%i5) y: y + j*y - 26.11;

(%o5)                         427.619249537037
```

it would be necessary to repeat the command `(%i3)` ten times. Another method consists in defining a function depending on an integer variable, using the recurrence relation, and to use that function to calculate y_{10} directly:

```
(%i6) y[0]: 500$

(%i7) y[n] := y[n-1] + j*y[n-1] - 26.11;

(%o7)                  y  := y      + j y      - 26.11
                        n     n - 1       n - 1
(%i8) y[10];

(%o8)                         255.1779109580579
```

Some care should be taken in Maxima when using functions of a integer argument. In the previous example, when we calculated `y[10]`, the values of `y[9]`, `y[8]`,..., `y[1]`, were also implicitly calculated and stored in memory. If we changed the recurrence relation, those values that were already calculated and stored would not be updated. Thus, before we change the recurrence relation, or the initial value `y[0]`, it is necessary to erase the previously calculated sequence, by using the command `kill`.

For example, if the value of the loan was duplicated to $ 1000, and the monthly payment was also duplicated, will the amount in debt after the tenth payment would also duplicate? let us see:

```
(%i9) kill(y)$
```

```
(%i10) y[0]: 1000$

(%i11) y[n] := y[n-1] + j*y[n-1] - 52.22;

(%o11)                  y  := y      + j y      - 52.22
                         n     n - 1      n - 1
(%i12) y[10];

(%o12)                         510.3558219161157
```

thus, the amount in debt also doubles.

Another question that we might ask in the original example is: what will the monthly payment should be if instead of 20 months we would want to pay the loan in 40 months?

To answer that question, we use a variable p to represent the monthly payment, we calculate the amount in debt after 40 months, as a function of p, and we equal that expression to zero to calculate the value of p.

```
(%i13) kill(y)$

(%i14) y[0]: 500$

(%i15) y[n] := expand(y[n-1] + j*y[n-1] - p)$

(%i16) solve(y[40] = 0, p);

RAT replaced 590.4757124853373 by 352514//597 = 590.4757118927973

RAT replaced -43.428341992962 by -26969//621 = -43.4283413848631

                                     72970398
(%o16)                          [p = --------]
                                     5366831
(%i17) %, numer;

(%o17)                      [p = 13.59655222979818]
```

The monthly payment should be \$ 13.60. The function `expand` had to be used to force Maxima to calculate the products in every step, avoiding large expressions with several parenthesis in the calculation of y_n. The additional messages written by Maxima tell us that some floating point numbers were represented as fractions, to prevent numerical errors.

2.1 Discrete systems evolution

The evolution of a first-order discrete system:

$$y_{n+1} = F(y_n) \tag{2.4}$$

is obtained by applying successively a function F, to the initial state $y_0 = c$:

$$\{c, F(c), F(F(c)), F(F(F(c))), \ldots\} \tag{2.5}$$

or in a more compact form:

$$\{c, F(c), F^2(c), F^3(c), \ldots y_n = F^n(c)\} \tag{2.6}$$

2.2 Graphical analysis

A graphical method to represent the evolution of a system consists on plotting a point for each step in the sequence, with x-coordinate equal to the index n and y-coordinate equal to y_n. In Maxima, the function `evolution` in the additional package `dynamics` will plot such a graph.

Three arguments should be given to that program. The first argument must be an expression that depends only on the variable y; that expression will specify $F(y)$ from the right hand side of the difference equation 2.1. The second argument should be the initial value y_0, and the third argument is the number of elements, of the sequence, that should be plotted.

For instance, in example 2.1, using the variable y, we have $F(y) = \cos y$, with initial value $y_0 = 2$. To obtain the evolution graph with the first 20 terms, we use the commands:

```
(%i18) load("dynamics")$

(%i19) evolution(cos(y), 2, 20)$
```

The graph obtaine i `(%i19)` is shown in figure 2.1.

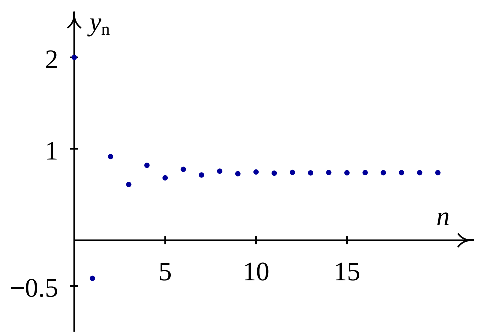


Figure 2.1: Evolution of $y_{n+1} = \cos(y_n)$ with $y_0 = 2$.

Another type of diagram which will be very useful to analyze discrete dynamical systems in one dimension is the so-called **staircase diagram**,[1] which consists in plotting the functions $y = F(x)$ and $y = x$, as well as an alternating sequence of horizontal and vertical segments joining the points

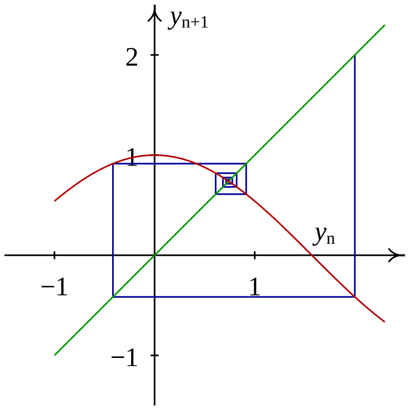


Figure 2.2: Staircase diagram for $x_{n+1} = \cos(x_n)$ with $x_0 = 2$.

(y_0, y_0), (y_0, y_1), (y_1, y_1), (y_1, y_2), and so on. For example, figure 2.2 shows the staircase diagram for the sequence represented in figure 2.1

The function `staircase`, included in the additional package `dynamics`, plots staircase diagrams. That function needs the same three arguments as the function `evolution`; namely, function $F(y)$ from the right-hand side of the difference equation 2.1, the initial value y_0 and the number of steps in the sequence. Notice that the independent variable in the expression for F should always be y. You might need to make the appropriate change if the state variable is something different from y in your problem.

For example, the graph 2.2 was obtained with the command

```
(%i20) staircase(cos(y),2,8)$
```

Notice that we did not have to load the package `dynamics` again, because it was already loaded in `(%i18)`. the staircase diagram allows us to understand when a sequence will converge or diverge. For instance, consider the system $y_{n+1} = y_n^2 - 0.2$. If we start with a value $y_0 = 1.1$, we obtain the graph on the left side of figure 2.3; we see that the sequence will converge to a negative value y, which is at the intersection of the functions $F(y) = y^2 - 0.2$ and $G(y) = y$, namely, $y = (5 - 3\sqrt{5})/10$.

The two functions intersect in another point, positive, equal to $y = (5 - 3\sqrt{5})/10$. In the graph we can see that even though the initial value was close to the second intersection point, the sequence moved away from it and towards the second intersection point, due to the fact that between the two intersection points the function $y^2 - 0.2$ is under $G(y) = y$. if we used an initial value to the right of the second intersection point, for instance, $y_0 = 1.5$, the sequence grows quickly towards infinity (right-hand side in figure 2.3). To make the sequences converge to the second intersection point, it would have been necessary that between the two intersection points $F(x)$ were above $G(y) = y$; that is to say, the slope of $F(y)$ should be less than 1, rather that greater than 1, at the second intersection point.

[1] Also know as cobweb diagram.

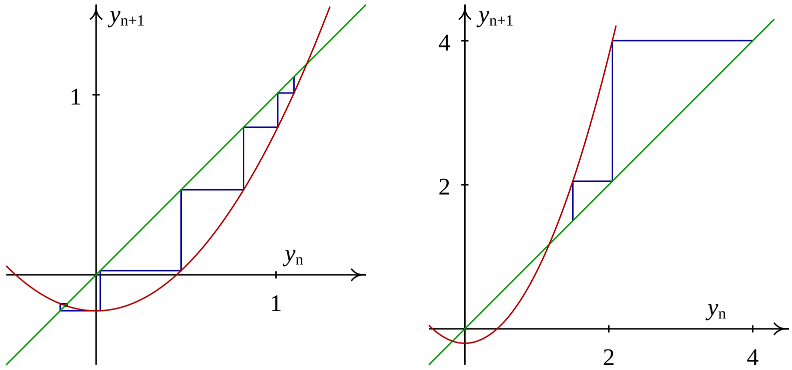


Figure 2.3: Solution to the system $y_{n+1} = y_n^2 - 0.2$ with initial values 1.1 (left) and 1.5 (right).

Example 2.3
Analyze the solutions to the logistic model, which consists in considering a population P with constant natality rate, a, and a mortality rate directly proportional to the population, bP, where a and b are constants.

Solution: The population under study could be a group of specimens from some animal species, where the sequence $\{P_0, P_1, P_2, \ldots\}$ represents represents the number of specimens throughout several consecutive years.

Let P_n represent the number of specimens at the beginning of period n. during that period of time, an average aP_n new specimens are born, and bP_n^2 specimens die. Thus, in the beginning of the next period, $n+1$, the population will be

$$P_{n+1} = (a+1)\,P_n\left(1 - \frac{b}{a+1}P_n\right) \tag{2.7}$$

It is convenient to define a variable y such as $y_n = b\,P_n/(a+1)$. Thus, we obtain an equation with a single parameter $c = a+1$

$$y_{n+1} = c\,y_n(1-y_n) \tag{2.8}$$

Figure 2.4 shows the solutions obtained with an initial value $y_0 = 0.1$, in the cases $c = 2$ and $c = 4$. With $c = 2$, the solution converges quickly to the fixed point $y = 0.5$.

With $c = 4$, the state of the system goes through many different values between 0 and 1, and it does not seem to follow any regular pattern. That type of behavior is called **chaotic**. the state in any given period is perfectly determined from the state in the previous period, but a very small modification of the state in an initial period will lead to a completely different pattern in later periods.

2.3 Stationary points

A **stationary point** of the system 2.1 is a point c at which the state of the system would stay constant. For that to happen, it will necessary and sufficient that

$$F(c) = c \tag{2.9}$$

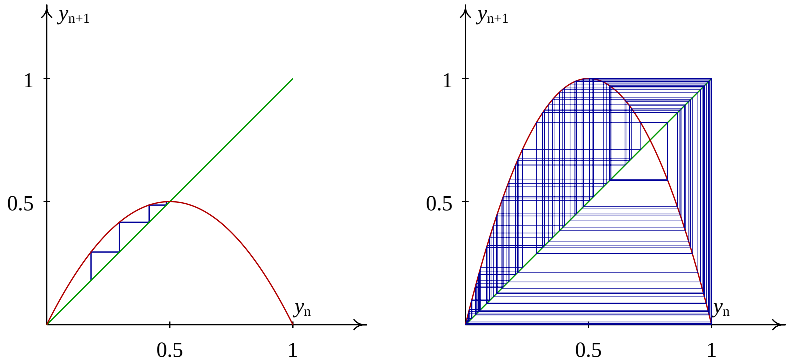


Figure 2.4: Solutions of the logistic model with an initial value 0.1. With $c = 2$ (left) the sequence converges, while with $c = 4$ (right) it becomes chaotic.

From the graphical point of view, the stationary points are all those points where the curve $F(x)$ intersects the right line $y = x$ in the stairway diagram. For example, in the case of he logistic model, figure 2.4 shows that in the two cases $c = 2$ and $c = 4$ there are two stationary points, one of them at $y = 0$. We can use Maxima's command `solve` to find the stationary points; in the case $c = 4$, it goes like this

```
(%i21) flogistic: 4*y*(1-y);

(%o21)                           4 (1 - y) y
(%i22) stationary: solve(flogistic - y);

                                    3
(%o22)                         [y = -, y = 0]
                                    4
```

The two stationary points are 0 and 0.75.

Let us consider a stationary point, where the curve $y = F(x)$ intersects the straight line $y = x$, such that the derivative $F'(y)$ is bigger than 1 at that point. Namely, as the curve $F(x)$ crosses the straight line from left to right, it goes from under the line to above it, and if we draw the staircase diagram starting at a point near that stationary point, the sequence will move away from that point. We say that the stationary point is repulsive, or unstable.

If the derivative is greater than 1, the trajectory in the staircase diagram moves always away from the stationary point, as in the case of the right-hand side of figure2.3; that kind of stationary point is called a **repulsive node**. If the derivative is less than -1, the trajectory also moves away from the stationary point, both alternating to both sides of the point, describing a "spider web"; we call that kind of point a **repulsive focus**.

If the absolute value of the derivative of the function, at the stationary point, is less than 1, $F(y)$ takes the points near the stationary point closer to it. The stationary point will either be an **attractive node**, if the derivative is positive, (the left-hand side of figure 2.4 shows an example), or an **attractive focus** if the derivative is negative (for example, the stationary point in figure 2.2). To summarize, we have the following kinds of stationary points y_s:

1. Attractive node, if $0 \leq F'(y_f) < 1$
2. Repulsive node, if $F'(y_f) > 1$
3. Attractive focus, if $-1 < F'(y_f) < 0$
4. Repulsive focus, if $F'(y_f) < -1$

Continuing with our previous example of the logistic model (see (%i21) through (%o22) above), the value of the derivative of F at the stationary points is:

```
(%i23) dflogistic: diff(flogistic, y);

(%o23)                      4 (1 - y) - 4 y
(%i24) dflogistic, stationary[1];

(%o24)                            - 2
(%i25) dflogistic, stationary[2];

(%o25)                             4
```

Thus, in the case $c = 4$, both stationary points are repulsive. At $y = 0$ there is a repulsive node, and there is a repulsive focus at $y = 0.75$.

2.4 Periodic points

If the sequence $\{y_0, y_1, y_2, \ldots\}$ is a solution of the dynamical system

$$y_{n+1} = F(y_n) \tag{2.10}$$

any element in the sequence can be obtained from y_0, applying the composed function F^n

$$y_n = F^n(y_0) = \underbrace{F(F(\ldots(F}_{n \text{ times}}(y)))) \tag{2.11}$$

A solution is dubbed a **cycle** of period 2, if it is a sequence of only two alternating values: $\{y_0, y_1, y_0, y_1, \ldots\}$, with $y_0 \neq y_1$. The two points y_0 and y_1 are **periodic points** with period equals to 2. Since $y_2 = F^2(y_0) = y_0$, it is necessary that $F^2(y_0) = y_0$. And since $y_3 = F^2(y_1) = y_1$, then $F^2(y_1) = y_1$. Furthermore, since $F(y_0) = y_1 \neq y_0$ and $F(y_1) = y_0 \neq y_1$, it is also necessary that $F(y_0) \neq y_0$ and $F(y_1) \neq y_1$.

Those conditions can be summarized by saying that both y_0 and y_1 are stationary points for the function $F^2(y)$, but they cannot be stationary points for the function $F(y)$.

The cycle of period two will be attractive or repulsive (stable or instable) according to the value of the derivative of F^2 at each point in the cycle.

To calculate the derivative of F^2 at y_0, we use the chain rule

$$(F^2(y_0))' = (F(F(y_0)))' = F'(F(y_0))F'(y_0) = F'(y_0)F'(y_1) \tag{2.12}$$

thus, the derivative of F^2 takes the same value in the two points y_0 and y_1 of the cycle, and it is equal to the product of the derivatives of F in those two points.

Generalizing the definition of a cycle, there will be a cycle of period m, formed by m values y_0, y_1, …, y_m, if $F^m(y_0) = y_0$ and $F^j(y_0) \neq y_0$, for $j < m$. All of the points y_0, y_1, …, y_m are stationary points for the function F^m but they cannot be stationary points of any of the functions $F^j(y_0) \neq y_0$ with $j < m$.

If the absolute value of the product of $F'(y_i)$ for all the m points of the cycle is greater than 1, then the cycle is repulsive; if the product is less than 1, the cycle is attractive, and if the product is identical to 1, the cycle is neutral.

Example 2.4
Find the cycles with period 2 for the logistic system

$$y_{n+1} = 3.1\, y_n(1 - y_n)$$

and say whether they are attractive or repulsive.

Solution: We start by defining function $F(y)$ and the composed function $F^2(y)$

```
(%i26) flog: 3.1*y*(1-y)$

(%i27) flog2: flog, y=flog;

(%o27)          9.610000000000001 (1 - y) y (1 - 3.1 (1 - y) y)
```

The stationary points with period equal to 2 will be among the solutions of the equation $F^2(y) - y = 0$

```
(%i28) periodic: solve(flog2 - y);

                   sqrt(41) - 41      sqrt(41) + 41      21
(%o28)       [y = - -------------, y = -------------, y = --, y = 0]
                        62                 62             31
```

The last two points, namely 0 and $21/31$ are stationary points (this can be confirmed by solving $F(y) - y = 0$).

The other two points then form a cycle of period two; if we use any of them as initial state, the sequence will oscillate between those two points.

To find out whether the cycle is attractive or repulsive, we compute the derivative in each of the points in the cycle

```
(%i29) dflog: diff(flog, y);

(%o29)                        3.1 (1 - y) - 3.1 y
```

```
(%i30) dflog, periodic[1], ratsimp, numer;

(%o30)                        - 0.3596875787352

(%i31) dflog, periodic[2], ratsimp, numer;

(%o31)                        - 1.640312421264802
```

The absolute value of the product is less than 1, which implies that the cycle is attractive.

2.5 Solving equations numerically

Discrete, first-oder dynamical systems can be used for solving one-variable equations numerically. The problem to be solved consists on finding the **roots** of a real function f, namely, the values of x that satisfy the equation

$$f(x) = 0 \tag{2.13}$$

For example, suppose we want to find the values of x that solve the equation:

$$3x^2 - x\cos(5x) = 6$$

That kind of equation cannot be solved analytically; it must be solved by numerical methods. The numerical methods to solve that equation consist on defining a dynamical system with convergent orbits, which approach to the solutions of the equation. In the following sections we will study two of those methods.

2.5.1 Iteration method

If the equation 2.13 can be written in the form

$$x = g(x) \tag{2.14}$$

Its solutions are the stationary points of the dynamical system:

$$x_{n+1} = g(x_n) \tag{2.15}$$

To find a stationary point, we choose some initial point and calculate the evolution of the system.

Example 2.5
Find the solution of the equation $x = \cos x$

Solution: This equation is already given in a form that allows us to use the iteration method. We use the dynamical system with recurrence relation

$$x_{n+1} = \cos(x_n)$$

To find an stationary point, we choose an arbitrary initial point and calculate the evolution of the system

```
(%i32) x: 1$

(%i33) for i thru 15 do (x: float(cos(x)), print(x))$

0.54030230586814
0.85755321584639
0.65428979049778
0.79348035874257
0.70136877362276
0.76395968290065
0.72210242502671
0.75041776176376
0.73140404242251
0.74423735490056
0.73560474043635
0.74142508661011
0.73750689051324
0.74014733556788
0.73836920412232
```

The solution of the equation is approximately 0.74. This method was successful in this example, because the stationary point of the dynamical system chosen happened to be attractive. If the point were repulsive, the iteration method would have failed.

Example 2.6
Find the square root of 5, using only additions, multiplications and divisions.

Solution: The square root of 5 is the positive solution of the equation

$$x^2 = 5$$

which can be written as:

$$x = \frac{5}{x}$$

we solve the dynamical system associated to the function

$$f(x) = \frac{5}{x}$$

It can be easily seen that for any initial value x_0, different from $\sqrt{5}$, the solution of that system will always be a cycle with period 2:

$$\left\{x_0, \frac{5}{x_0}, x_0, \frac{5}{x_0}, \ldots\right\}$$

To escape from that cycle, and approach the stationary point at $\sqrt{5}$, we can try to use the middle point:

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{5}{x_n}\right)$$

That new system will converge quickly to the stationary point at $\sqrt{5}$:

```
(%i34) x : 1$

(%i35) for i thru 7 do (x: float((x + 5/x)/2), print(x))$
3.0
2.333333333333334
2.238095238095238
2.236068895643363
2.236067977499978
2.23606797749979
2.23606797749979
```

2.5.2 Newton's method

Newton's method can be used to find the roots of the equation 2.13. We start by assuming that a root of the equation is near the value x_0 and we improve that initial guess by finding the point x_1 where the tangent to the function at $f(x_0)$ intersects the x axis (see figure 2.5)

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} \tag{2.16}$$

We can use the same equation to further improve our guess x_1 to a new guess x_2. In general

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2.17}$$

Excluding some exceptional cases, if the function f is continuous and differentiable, the sequence x_n will approach a root of f. Thus, the roots of f are the stationary points of the dynamical system defined by equation 2.17.

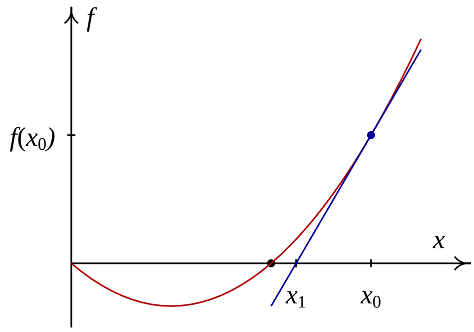


Figure 2.5: Newton's method for finding roots of an equation.

To illustrate the method, we will solve example 2.6 once again, using Newton's method.

The square root of 5 is one of the solutions of the equation $x^2 = 5$. Hence, to find the square root of 5 we can search for the positive root of the function

$$f(x) = x^2 - 5$$

The derivative of that function is

$$f'(x) = 2x$$

substituting it into the recurrence relation 2.17, we obtain

$$x_{n+1} = x_n - \frac{x_n^2 - 5}{2x_n} = \frac{1}{2}\left(x_n + \frac{5}{x_n}\right)$$

which is exactly the same sequence that we have already obtained and solved in the previous section. However, in this case we did not need to introduce any clever tricks; we just applied the standard method.

2.6 References

Some useful references, with a level similar to the one used here, are *Chaos* (Alligood et al., 1996), *A First Course in Chaotic Dynamical Systems* (Devaney, 1992) and *Chaos and Fractals* (Peitgen et al., 1992).

2.7 Multiple-choice questions

1. The state variable of a first-order, discrete dynamical system takes on the values from the following sequence:
$\{3.4, 6.8, 7.2, 5.1, 6.8, \ldots\}$
what can be concluded about that system:

A. it does not have any cycles with period less than 5.

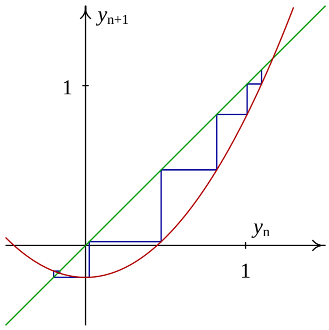
B. it has a stationary point.

C. it is a chaotic system.

D. it has a cycle of period 3.

E. it has a cycle of period 2.

2. The figure shows the staircase diagram of the discrete dynamical system $y_{n+1} = y_n^2 - 0.2$, which has two stationary points $y = -0.17$ and $y = 1.17$.



what type of stationary point is $y = 1.17$?

A. repulsive focus.

B. attractive focus.

C. attractive node.

D. part of a cycle with period 2.

E. repulsive node.

3. A first-order discrete dynamical system has a single stationary point at 0.739, and no cycles. Starting with an initial value 2, the evolution of the system is the sequence: $\{2, 0.915, 0.610, 0.820, 0.683, \ldots\}$ what can be said about that system?

A. it is chaotic.

B. the stationary point is attractive.

C. it has a cycle with period 2.

D. it has a cycle with period 3.

E. the stationary point is repulsive.

4. A function $F(x)$ has the following properties:

$$F^2(2) = 5 \qquad F(5) = 2$$

thus, we can conclude that the discrete dynamical system $x_{n+1} = F(x_n)$ has a cycle with period equal to:

A. 2

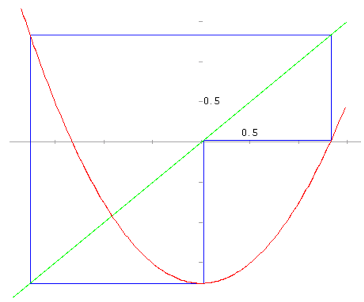
B. 3

C. 4

D. 5

E. 1

5. The figure shows the staircase diagram for the first 40 iterations of a discrete dynamical system.



thus, we can conclude that the system has:

A. an attractive focus.

B. a repulsive focus.

C. a cycle with period 2.

D. a cycle with period 3.

E. a cycle with period 40.

2.8 Problems

1. The sequence obtained in this chapter to calculate square roots,

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$$

was already known by the Sumerians, 4000 years ago. Using that method, calculate $\sqrt{3}$, $\sqrt{15}$ and $\sqrt{234}$. Use any positive initial value and represent the number a as a floating-point number (for instance, 3.0), to force Maxima to give its results also as a floating-point number. In each case, compare the result with the value obtain using function **sqrt()** in Maxima.

2. Assume that the current whale population in the world is 1000 and that every year the normal increase of the population (births minus deaths by natural causes) is 25%. Assuming that the number of whales killed by fishermen every year remained constant at 300 during the next years, how will the whale population evolve during the next 10 years?

3. Compute the first 10 terms of the sequence defined by the equation:

$$x_{n+1} = x_n^2 - 2$$

using the following initial values:

(a) $x_0 = 1$
(b) $x_0 = 0.5$
(c) $x_0 = 2$
(d) $x_0 = 1.999$

Discuss the behavior of the sequence in each case.

4. For each function in the following list, the point $y = 0$ makes part of a cycle for the system $y_{n+1} = F(y_n)$. Determine the period of the cycle for each case and calculate the derivative of the function in order to determine whether the cycle is attractive or repulsive. Draw the staircase diagram of the sequence with initial value 0.

(a) $F(y) = 1 - y^2$
(b) $F(y) = \frac{\pi}{2}\cos y$
(c) $F(y) = -\frac{1}{2}y^3 - \frac{3}{2}y^2 + 1$
(d) $F(y) = |y - 2| - 1$
(e) $F(y) = -\frac{4}{\pi}\text{arctg}(y + 1)$

5. Find the stationary points and the cycles with period 2 of the dynamical system:

$$y_{n+1} = F(y_n)$$

and classify each point and cycle as attractive or repulsive, for each of the following cases:

(a) $F(y) = y^2 - \frac{y}{2}$
(b) $F(y) = \frac{2 - y}{10}$
(c) $F(y) = y^4 - 4y^2 + 2$
(d) $F(y) = \frac{\pi}{2}\sin y$
(e) $F(y) = y^3 - 3y$
(f) $F(y) = \text{arctg}\,(y)$

In each case, start by drawing a staircase diagram using the function **staircase** and use it to find out the position of the stationary points and cycles; use the option `domain` to get a better overview of the position of the stationary points. Then try to find the points analytically. In some cases that will not be possible and the result will have to be obtained approximately from the plots.

6. Considering the sequence $x_{n+1} = |x_n - 2|$

(a) Find all the stationary points. Show those points in a plot of the functions $F(x) = |x-2|$ and $G(x) = x$.

(b) Explain the kind of sequence that will be obtained if x_0 were a integer, either even or odd.

(c) Find the solution for the initial value 8.2.

(d) Find all the cycles with period two. Show all the points in those cycles in a plot of the functions $F^2(x)$ and $G(x) = x$.

7. Consider the function

$$F(x) = \begin{cases} 2x & ,\text{if} \quad x \leq 1 \\ 4-2x & ,\text{if} \quad x \geq 1 \end{cases}$$

(a) Show that $F(x)$ is equivalent to $2-2|x-1|$.

(b) Plot, in the same graph, the functions $F(x)$, $F^2(x)$, $F^3(x)$ and $G(x) = x$. What can you conclude about the stationary points and cycles of the system $x_{n+1} = F(x_n)$?

(c) Make a table or plot a graph of n against x_n, between $n = 0$ and $n = 100$, for each of the initial values 0.5, 0.6, 0.89, 0.893 and 0.1111111111. Discuss the results obtained.

(d) In the previous item, the sequence remains constant, starting at $n = 55$, for each of the initial values. Compute again the sequences obtained in the last item, using the following program, which uses higher numerical precision than the function **evolution** from the **dynamics** module.

```
evolution60(f, x0, n) :=
block([x: bfloat(x0), xlist:[0, x0], fpprec: 60],
for i thru n
do (x: ev(f), xlist: append(xlist, [i, float(x)])),
openplot_curves([["plotpoints 1 nolines"], xlist])
)
```

what can you conclude?

Index

Bibliography

Alligood, K. T., Sauer, T. D. & Yorke, J. A. (1996), *Chaos - an introduction to dynamical systems*, Springer-Verlag.

de Souza, P. N., Fateman, R., Moses, J. & Yapp, C. (2003), *The Maxima book*, http://maxima.sourceforge.net.

Devaney, R. L. (1992), *A first course in chaotic dynamical systems: theory and experiments*, Westview Press.

Peitgen, H.-O., Jürgens, H. & Saupe, D. (1992), *Chaos and fractals: new frontiers of science*, Springer-Verlag.