

## Exercise sheet

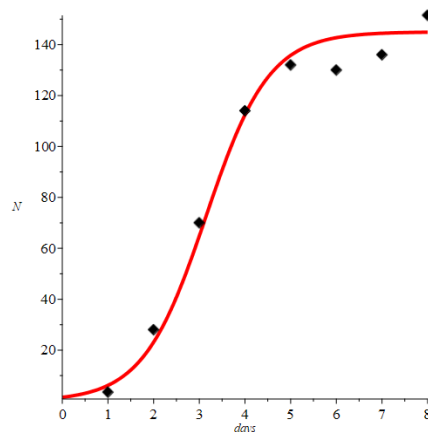
In this practical we will work on:

- Determining steady states by hand and by using simulations;
- Determining the stability of steady states by hand, by using direction fields, and/or by using the Jacobian matrix and eigenvalues;
- Determining a bifurcation using steady state analysis;
- Eyeball fitting a model on some data.

Solutions will be made available on Brightspace.

### THE LOGISTIC EQUATION

The logistic growth has long been used for describing the growth of a population under limitation. For instance, Gause et al. (1934) published fits of the logistic model to data of *Paramecium* in batch experiments. Below we see a reproduced fit to the population density of *P. caudatum* ( $N$ ) changing in time ( $t$ ).



Consider first the equation

$$\frac{dX}{dt} = rX \quad \text{Eq. (1)}$$

We drop the time-dependence of population size  $X$  in the notation, but this variable depends on time.

The explicit solution of Eq. (1) can be found by separation of variables, followed by antidifferentiation:

$$\Rightarrow \int \frac{1}{X} dX = \int r dt \Rightarrow \ln(|X|) = rt + C \Rightarrow X = ce^{rt}$$

We would then have an explicit description of the population size as function of time. We consider  $t \geq 0$ , and  $c$  is a constant that can be determined by setting an initial condition.

It is obvious that the state variable in Eq. (1) will continue to grow exponentially. In other words, it has no steady state other than  $X = 0$ . This makes no biological sense,

because at some point a limitation will kick in, as seen in the data published by Gause et al. (1934).

The Pearl-Verhulst or logistic equation is a way to impose such a restriction. It is given as

$$\frac{dX}{dt} = \dot{X} = rX \left(1 - \frac{X}{K}\right) \quad \text{Eq. (2)}$$

Suppose that  $X$  denotes the total population size in a limited spatial area.

**Q1:** Give the meaning of  $r$  and  $K$ , and give their dimensions.

$r$  is the intraspecific growth rate of the population. Its dimensions are [per time].

$K$  is the carrying capacity. Its dimensions are identical to that of  $X$  [mass X].

The logistic equation has a nontrivial (nonzero) steady state  $X = K$  (see the tutorial slides). Your most likely assumption is that the nonzero steady state will be stable, i.e., the population will eventually grow to  $K$ . We can prove this by looking at the derivative of the ODE, because this gives the direction of change.

**Q2:** Algebraically determine the derivative of Eq. (2). (*Hint: You can check your result, for instance, by using  $D(f, 'X')$  in R.*)

Let's expand:  $\dot{X} = rX - rX^2/K$

Then the sum rule for differentiation:  $f = \frac{d\dot{X}}{dX} = r - \frac{2rX}{K}$

Even without knowing the exact values of  $r$  and  $K$ , we can determine the stability of the nonzero state.

**Q3:** Verify the stability of the nonzero steady state of Eq. (2), using the derivative found in the previous question. Assume  $r, K > 0$ . (*Hint: Verify your results by evaluating the derivative.*)

The state is  $X = K$ , so  $f(X) = f(K) = r - 2r \cdot \frac{K}{K} = -r < 0$

Or numerically in R:

```
# Parameters
r <- 1.3
K <- 10

# Equations (2a-d)
f <- expression(r*X*(1-X/K))

# Partial derivatives
df <- D(f, 'X')
```

```
# Steady state
X <- 10

# Check: should be (near) zero, i.e., steady state
eval(f)

# Evaluate derivative for stability; check: should be negative
eval(df)
```

There is also an algebraic solution of Eq. (2). It is given as

$$X(t) = \frac{KX_0}{X_0 + (K - X_0)e^{-rt}} \quad \text{Eq. (3)}$$

We encourage you to verify it by hand (see, for instance, the book for Mathematics 2 (MAT14903), p. 206).

Algebraic solutions present a strong verification for your numerical code.

**Q4:** Implement Eq. (2) in an R or Python notebook. Use  $r = 0.5$  and  $K = 1$ . Use  $X(0) = 0.2$ . Also use some other initial conditions, including  $X(0) = 0.1$ . Confirm your numerical results using your algebraic solutions and Eq. (3).

```
In R:
library(deSolve)
time <- c(seq(0.01, 20, by = 0.01))
parms1 <- c(r=0.5, K=1., y10 = 0.2)
model <- function(time, y, parms) {
  with(as.list(parms), {
    dX <- r*y[1]*(1-y[1]/K)
    list(c(dX))
  })
}
out1 <- ode(y = c(X = parms1[["y10"]]), times = time, func = model, parms =
parms1)
```

#### Determining stability of steady states: direction fields and plots

An important way of studying the stability of a steady state is by looking at the direction field of the model. We evaluate the function Eq. (2) (i.e., we substitute a particular value of  $t$  and  $X$  and calculate the value of  $dX/dt$ ) at different points and plot these derivatives in a plot. For instance, for  $r = 0.5$  and  $K = 1$ , we have the following:

$t$	$X(t)$	$dX(t)/dt$
0	0.2	0.08
1	0.29186	0.10334
2	...	...
5	0.75282	...
10	...	...
25	...	$7 \cdot 10^{-6}$

**Q5:** Complete the above Table using Eq. (2) and Eq. (3).

$t$	$X(t)$	$dX(t)/dt$
0	0.2	0.08
1	0.29186	0.10334
2	0.40461	0.12045
5	0.75282	0.09304
10	0.97376	0.01278
25	0.99999	$7 \cdot 10^{-6}$

We are plotting  $X(t)$  (the 'y coordinate') as function of  $t$  (the 'x coordinate'). A direction field is constructed from drawing the direction vector at each support point  $t = a$ , or alternatively, draw a small part of the tangent line to the graph in that point. Recall, that the vector representation of a tangent line to a point is

$$\begin{pmatrix} t \\ X(t) \end{pmatrix} = \begin{pmatrix} a \\ f(a) \end{pmatrix} + \lambda \begin{pmatrix} 1 \\ f'(a) \end{pmatrix}$$

Here  $f(a)$  is Eq. (3), and  $f'(a)$  is Eq. (2) with the value of Eq. (3) substituted. We can convert the vector representation to an algebraic expression by rewriting the upper row

$$t = a + \lambda \Rightarrow \lambda = t - a$$

Then substituting this into the lower row

$$X(t) = f(a) + \lambda f'(a) = f(a) + f'(a)(t - a)$$

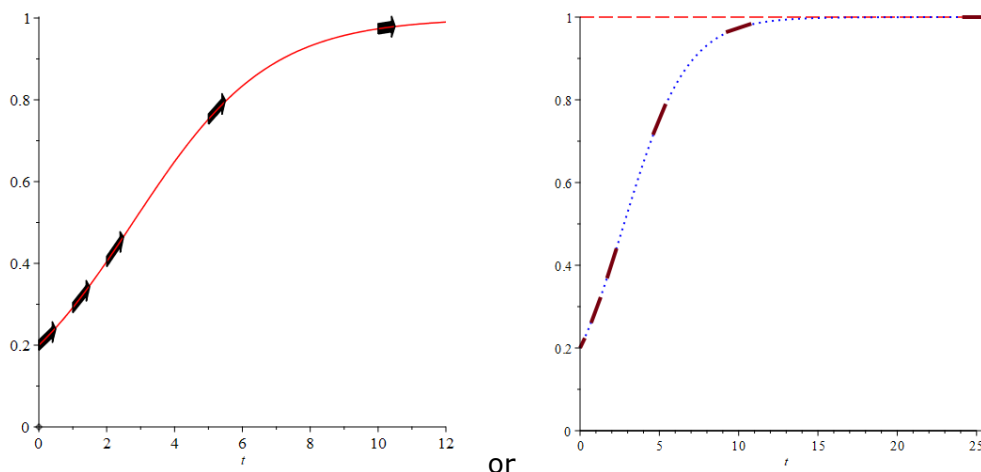
You recognize this as the algebraic representation of a tangent line in the point  $t = a$ . For instance, in the point (1,0.29186) we have

$$X(1) \cong f(1) + f'(1)(t - 1) = 0.29186 + 0.10334(t - 1) = 0.18853 + 0.10334t$$

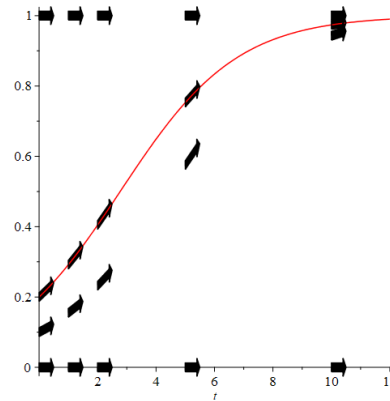
The vector for that point is

$$\begin{pmatrix} 1 \\ 0.29186 \end{pmatrix} + \lambda \begin{pmatrix} 1 \\ 0.10334 \end{pmatrix}$$

After drawing the direction vectors, you get something like the following (in the left panel with vectors, in the right panel in blue dotted the ODE solution and the tangents in red):



The red curve in the left graph (similar to the blue dotted curve in the right panel) is the simulation result for  $r = 0.5$ ,  $K = 1$ , and  $X(0) = 0.2$ . You see that the vectors indicate the direction in which the simulation develops. We can extend this left graph to include, e.g., also vectors for the initial conditions  $X(0) = 0$ ,  $X(0) = 0.1$ , and  $X(0) = 1$ :



**Q6:** Explain how you can conclude from this that the steady state  $X = K = 1$  is stable, and  $X = 0$  is unstable.

All vectors for  $X=0$  are horizontal and lie along the x-axis, indicating there is no change, i.e.,  $X=0$  is a steady state. Vectors for  $X=K=1$  are again horizontal along  $y=1$ , indicating no change and hence  $X=K$  is a steady state. All vectors not for  $X(0) = 0$  form a sequence deviating from  $X=0$  and towards  $X=K$ . Hence,  $X=0$  is unstable, and  $X=1$  is stable.

In practice, evaluating the derivatives for many different values of  $(t, X(t))$  is cumbersome. In addition, you often do not have the explicit solution, such as Eq. (2) as explicit solution of Eq. (1). If you want to draw a direction field, you can resort to numerical means for drawing it, so we can still graphically determine the stability of steady states. For instance, in R, you can use the package 'phaseR' for this. We then use the following commands:

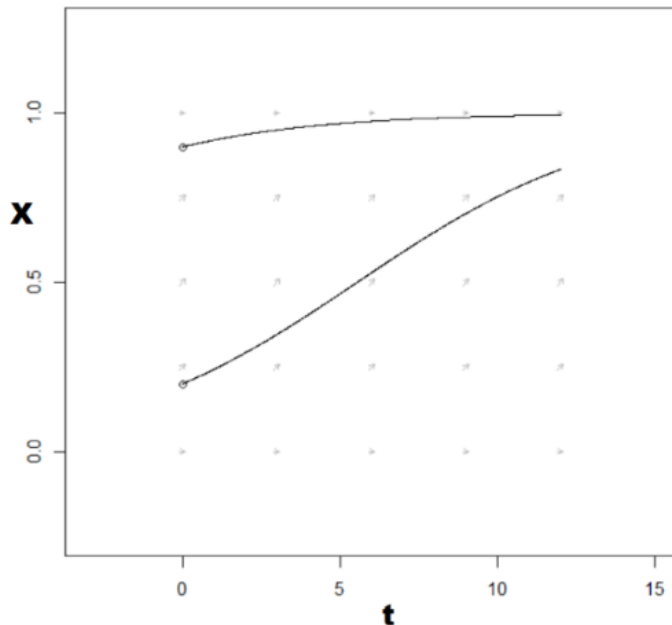
```
library(phaseR)

equations <- function(t, y, parameters)
{
  r = parameters[1]
  K = parameters[2]
  list(r*y*(1-y/K))
}

flowField(equations, xlim, ylim, parameters, system = "one.dim", points, add = FALSE)
trajectory(equations, y0, tlim, parameters, system = "one.dim")
```

**Q7:** Make a direction field using Python or R, and verify with the earlier graph whether you see the same general pattern.

The result should look something like this (note, that the arrowheads for  $X=0$  are horizontal to the right; for  $X>0$  they go upward to the state  $X=K$ ):



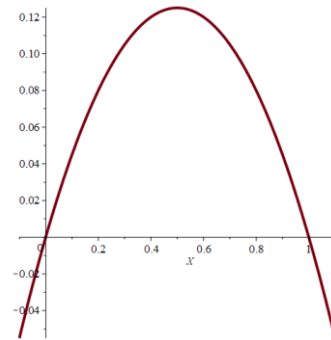
```
In R:
.libPaths("library")
#install.packages("phaseR")
library(phaseR)

out1 <- function(t, y, parameters)
{
  r = parameters[1]
  K = parameters[2]
  list(r*y*(1-y/K))
}

flow.field <- flowField(out1,
xlim = c(0, 12),ylim = c(0, 1.),
parameters = c(0.5, 1),
system = "one.dim", points = 5,add = FALSE)

trajec.tory <- trajectory(out1,
y0 = c(0.2, 0.9),
tlim = c(0, 12),
parameters = c(0.25, 1),
system = "one.dim")
```

A direction field may allow you to identify steady states you have missed, or that are attracting from one side, but not attracting from another side (so-called saddle-node points). In the case of one ODE, like with Eq. (2), we can also use another alternative graphical means of determining steady states and their stability. We can plot the ODE as function of the state variable. In this case, if we plot Eq. (2) on the y-axis, and  $X$  on the x-axis, we get something like this:



**Q8:** Make such a plot with Eq. (2) on the y-axis, and  $X$  on the x-axis. Explain how we can determine what are the steady states, and what is the stability of each of them.

When  $0 < X < 1$ , the derivative  $\dot{X}$  is positive, indicating growth. When  $X < 0$  or  $X > 1$ ,  $\dot{X}$  is negative, indicating (further) decrease. So, all orbits will diverge from  $X = 0$  and converge to  $X = K = 1$ . An example of achieving the plot in R could be:

```
f <- function(x,r,K) {
  dx <- r*x*(1-x/K)
  return(dx)
}
i.vec <- c(seq(-0.1, 1.1, by = 0.01))
length.i.vec <- length(i.vec)
nul.line <- rep(0, length.i.vec)
data.vec <- c(seq(1, length.i.vec, by = 1))

for (i in 1:length.i.vec) {
  data.vec[i] <- f(i.vec[i],0.5,1)
}
png("picture.png")
plot(data.vec ~ i.vec, xlab = "X", ylab = "dX/dt", pch = 20, cex = 1)
lines(data.vec ~ i.vec, lwd = 5)
lines(nul.line ~ i.vec)
dev.off()
```

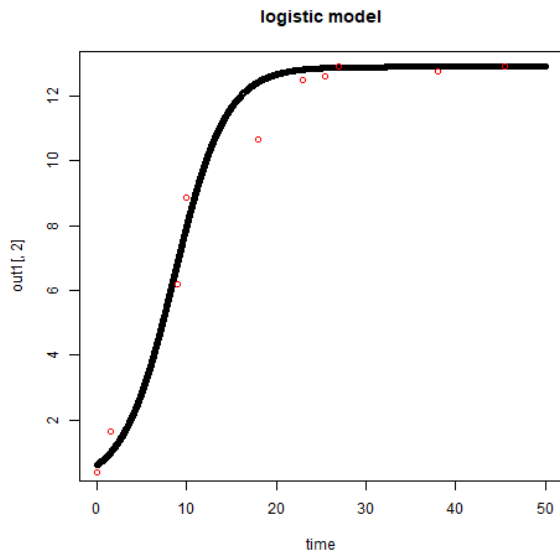
### *Eyeball fitting*

Now consider the following data file from an experiment by Gause et al. (1935). The data are measurements of yeast cell volumes as a function of time (hours):

Time (hours)	Yeast cell volume
0	0.37
1.5	1.63
9	6.2
10	8.87
18	10.66
23	12.5
25.5	12.6
27	12.9
38	12.77
45.5	12.9

**Q9:** By trying out different values of  $r, K$ , and the initial condition, see if you can get an appropriate fit to the data. (*Hint: Think about a logical starting point for the initial condition and the value of  $K$ . Start, e.g., with  $r = 0.5$ ,  $K = 10$ , and  $0 < X(0) < 1$ .)*)

An eyeball fit may look something like this:



Values used for the graph are  $r=0.35$ ,  $K=12.9$ ,  $y_{10} = 0.6$ . The value of  $K$  should be something like the last few values in the second column, while a logical starting point for an initial condition would be the first volume value = 0.37. Comparable code is given for Q6.

```
.libPaths("library")
#install.packages("deSolve")
library(deSolve)

DT <- read.table("data31.dat", header = -1)
t <- DT$time.hours.
x <- DT$Volume

time <- c(seq(0.01, 50, by = 0.01))
parms1 <- c(r=0.35, K=12.9, y10 = 0.6)
model <- function(time, y, parms) {
  with(as.list(parms), {
    dX <- r*y[1]*(1-y[1]/K)
    list(c(dX))
  })
}

out1 <- ode(y = c(X = parms1[["y10"]]), times = time, func = model, parms =
parms1)

png("picture.png")
plot(time, out1[,2], col = 1, main = "logistic model")
points(x ~ t, col = 2)
dev.off()
```



In the logistic model only the population is explicitly considered. We now turn our attention to another model, in which we explicitly consider substrate dynamics in a chemostat reactor, i.e., we now have a model consisting of two ODEs:

$$\frac{dX}{dt} = D(X_{in} - X) - \frac{eXN}{k + X} \quad \text{Eq. (3a)}$$

$$\frac{dN}{dt} = y \frac{eXN}{k + X} - DN \quad \text{Eq. (3b)}$$

Here  $D$  is the dilution rate (with units 'per time') through a well-mixed chemostat reactor, such that the influx and outflux rates are identical;  $X$  is the explicitly modelled substrate (in mol per volume) needed for population  $N$  (in mol or number of cells per volume, depending on how the counting is done) to grow, where  $X_{in}$  is the concentration of the substrate entering the chemostat; the growth of  $N$  follows a Holling type II functional response (i.e., Michaelis-Menten kinetics), with  $e$  an ingestion rate, and  $y$  a yield efficiency of the conversion from the substrate to  $N$ . For  $N = 0$ , there is a trivial steady state  $X = X_{in}$ , which essentially says that if there is no population to feed on the substrate, it will flow out of the chemostat reactor in the same density per time as it enters. Eqs. (3a-b) also have a nontrivial steady state, given as

$$\tilde{X} = \frac{Dk}{ye - D}, \tilde{N} = \frac{y(DX_{in} + Dk - yeX_{in})}{D - ye}$$

**Q10:** Algebraically determine the partial derivatives of Eqs. (3a-b). (*Hint: If you struggle with partial derivatives, consider MAT14903 (Mathematics 2) and in particular tutorial 9 of the course MAT15003 (Mathematics 3) again.*)

You have four. They are:

$$\frac{\delta}{\delta X} \left( D(X_{in} - X) - \frac{eXN}{k + X} \right) = -D - \frac{eN(k + X) - eXN}{(k + X)^2} = -D - \frac{eNk}{(k + X)^2}$$

(Quotient rule;  $X_{in}$  is a constant)

$$\frac{\delta}{\delta N} \left( D(X_{in} - X) - \frac{eXN}{k + X} \right) = -\frac{eX}{k + X}$$

( $-\frac{eX}{k+X}$  is simply a constant)

$$\frac{\delta}{\delta X} \left( y \frac{eXN}{k + X} - DN \right) = \frac{yeN(k + X) - yeNX}{(k + X)^2} = \frac{yeNk}{(k + X)^2}$$

(Quotient rule again;  $-DN$  is a constant)

$$\frac{\delta}{\delta N} \left( y \frac{eXN}{k + X} - DN \right) = \frac{yeX}{k + X} - D$$

(Sum rule, as  $-\frac{eX}{k+X}$  is a constant again)

Given are the following parameter values:  $X_{in} = 10, D = 0.1, e = 1, y = 0.5, k = 1$ .

**Q11:** Substitute the above parameter values and determine the eigenvalues (manually or by using R or Python) of the above nontrivial steady state, and indicate whether this steady state is stable or not.

Substitute the parameter values in the steady state values and in the partial derivatives, then construct the Jacobian matrix:

$$J = \begin{pmatrix} -3.22 & -0.2 \\ 1.56 & 0 \end{pmatrix}$$

Determine eigenvalues:

$$\begin{aligned} \text{Det} \begin{pmatrix} -3.22 - \lambda & -0.2 \\ 1.56 & -\lambda \end{pmatrix} &= -\lambda(-3.22 - \lambda) - 0.2 \cdot 1.56 = \lambda^2 + 3.22\lambda + 0.312 = 0 \\ \Rightarrow \lambda_{1,2} &= \frac{-3.22 \pm \sqrt{(3.22)^2 - 4 \cdot 1 \cdot 0.312}}{2 \cdot 1} = \frac{-3.22 \pm \sqrt{9.1204}}{2} = \frac{-3.22 \pm \sqrt{3.02}}{2} \end{aligned}$$

Hence:  $\lambda_1 = -3.12$  or  $\lambda_2 = -0.1$

**Q12:** Implement the model in Python or R, and use simulation to confirm the results of the previous question.

Substituting the parameter values ( $X_{in} = 10, D = 0.1, e = 1, y = 0.5, k = 1$ ) in the algebraic steady states:

$$\tilde{X} = \frac{Dk}{ye-D} = \frac{0.1 \cdot 1}{0.5 \cdot 1 - 0.1} = \frac{0.1}{0.4} = 0.25,$$

$$\tilde{N} = \frac{y(DX_{in} + Dk - yeX_{in})}{D - ye} = \frac{0.5 \cdot (0.1 \cdot 10 + 0.1 \cdot 1 - 0.5 \cdot 1 \cdot 10)}{0.1 - 0.5 \cdot 1} = 4.875$$

Based on the eigenvalues and the simulations (if run for an adequate number of simulation steps) it should be confirmed the steady state is stable.

Now consider the same parameter values as above, except  $X_{in} = 0.1$ .

**Q13:** Determine the steady states and their stability as you did above. Do you get the same **qualitative** results as before? Give a biological interpretation of the results.

No, you do not. There is only one non-negative steady state:  $(0.1, 0)$ . The other steady state is now negative:  $(X = 0.25, N = -0.075)$ . Their stability is:

$$J|_{(0.1, 0)} = \begin{pmatrix} -0.1 & -0.9091 \\ 0 & -0.5454 \end{pmatrix}, \text{ which gives eigenvalues } \lambda_1 = -0.1, \lambda_2 = -0.05454, \text{ so stable}$$

$J|_{(0.25, -0.075)} = \begin{pmatrix} -0.052 & -0.2 \\ -0.024 & 0 \end{pmatrix}$ , which gives eigenvalues  $\lambda_1 = -0.1, \lambda_2 = 0.048$ , so unstable

The trivial steady state  $(X_{in}, 0)$  is now stable, i.e., no consumer population can settle.

There apparently is a bifurcation in this system, a critical value of  $X_{in}$  where the system behaviour undergoes a qualitative change.

**Q14:** In Jupyter, implement a systematic scan in which you step-wise vary the value of  $X_{in}$  and calculate the eigenvalues. Locate the bifurcation value of  $X_{in}$ . (*Hint: you can verify the result by substituting the parameter values in the two steady state values: these should be identical.*)

The bifurcation occurs at  $X_{in} = 0.25$ . This is the point where the two steady states coincide. In other words, substitution of  $X_{in} = 0.25$  in the steady state values should give the same results  $(0.25, 0)$ .

**Q15:** In the previous question, you have calculated the eigenvalues. Explain what you notice with regard to the eigenvalues, and use this to define a criterion based on the eigenvalues for a bifurcation like we have here.

The eigenvalues in the bifurcation point  $X_{in} = 0.25$  are  $\lambda_1 = -0.1, \lambda_2 = 0$ . The bifurcation is characterised by one eigenvalue equal to zero.

References:

[https://www.jstor.org/stable/pdf/1145.pdf?casa\\_token=-Nx4A3JIJfMAAAAA:AZAJMzBb9PBGWKOsuYZv1u8e\\_O0pn\\_KaO3V4zVaXHZgMn-1Wj87r1vO\\_oyM2UJV1CoI1WmEwHIs-r-7\\_EqU2zQ11q301X6DCfe936OPrAbeMsXZtEFWb](https://www.jstor.org/stable/pdf/1145.pdf?casa_token=-Nx4A3JIJfMAAAAA:AZAJMzBb9PBGWKOsuYZv1u8e_O0pn_KaO3V4zVaXHZgMn-1Wj87r1vO_oyM2UJV1CoI1WmEwHIs-r-7_EqU2zQ11q301X6DCfe936OPrAbeMsXZtEFWb)

#