Introduction to Computational Physics – Exercise 7

Simon Groß-Bölting, Lorenz Vogel, Sebastian Willenberg

June 12, 2020

Population dynamics

The simple equation of growth of a population, as proposed by Malthus (1798), has been improved by Verhulst (1836) including a growth limiting term, which represents the finite amount of resources available. In this exercise we study a modified form of Verhulst's equation for population dynamics:

$$\frac{\mathrm{d}N}{\mathrm{d}t} = rN(1 - N/K) - \frac{BN^2}{A^2 + N^2} \tag{1}$$

where all parameters r, K, A and B are positive. It is a more complex example, in which the growth behaviour depends on whether N is smaller or larger than a critical populations size A.

Dimensional analysis

Task: Determine the dimension of the parameters and rewrite the equation in dimensionless form. Note that there are different possibilities. Please formulate a dimensionless time τ that is not defined on the basis of r. Use $n = N/A$ as the dimensionless version of N.

Table 1: Dimension of the parameters

description	parameter	dimension
time	t	$[t] = \mathrm{s}$
population number	N	$[N] = 1$
growth rate	r	$[r] = \mathrm{s}^{-1}$
growth limiting number	K	$[K] = 1$
critical population size	A	$[A] = 1$
—	B	$[B] = \mathrm{s}^{-1}$

By dimensional analysis we can renormalize the variables, such that the above equation for population dynamics becomes dimensionless: The steps required for this are well described in the Wikipedia article "Nondimensionalization". First of all we have to identify all the independent and dependent variables in the given equation:

$$\frac{\mathrm{d}N}{\mathrm{d}t} = rN\left(1 - \frac{N}{K}\right) - \frac{BN^2}{A^2 + N^2} = rN - \frac{rN^2}{K} - \frac{BN^2}{A^2 + N^2} \tag{2}$$

In this equation the independent variable is t (time) and the dependent variable is $N = N(t)$ (population number). Now we replace each of the two variables with the product of a dimensionless variable (τ and n) and a characteristic unit (N_c and t_c):

$$N := N_\mathrm{c} n \quad \Leftrightarrow \quad n := N/N_\mathrm{c} \qquad \text{and} \qquad t := t_\mathrm{c}\tau \quad \Leftrightarrow \quad \tau := t/t_\mathrm{c} \tag{3}$$

The subscripted "c" added to a variable-name is used to denote the characteristic unit used to scale that variable. As required in the task, we set $N_\mathrm{c} := A$, so that we use $n := N/A$ as the dimensionless version of the population number N. Then we obtain the following equation:

$$\frac{A}{t_\mathrm{c}}\frac{\mathrm{d}n}{\mathrm{d}\tau} = rAn - \frac{rA^2}{K}n^2 - \frac{BA^2n^2}{A^2 + A^2n^2} = rAn - \frac{rA^2}{K}n^2 - B\frac{n^2}{1 + n^2} \tag{4}$$

Dividing the entire equation by the coefficient A/t_c in front of the first derivative term and using the relation

$$\frac{n^2}{1+n^2} = \left[\frac{1+n^2}{n^2}\right]^{-1} = \left[1+\frac{1}{n^2}\right]^{-1} = \left[1+n^{-2}\right]^{-1} = \frac{1}{1+n^{-2}} \tag{5}$$

gives us:

$$\frac{\mathrm{d}n}{\mathrm{d}\tau} = t_c r n - \frac{t_c r A}{K}\, n^2 - \frac{t_c B}{A}\,\frac{1}{1+n^{-2}} \tag{6}$$

Now the characteristic unit for each variable can be defined, such that the coefficients of as many terms as possible become 1. Since we have already determined the dimensionless version of N, we only have to define the characteristic unit t_c of the time t. Because we are supposed to formulate a dimensionless time $\tau = t/t_c$ that is not defined on the basis of the growth rate r, we have to use the coefficient in front of the last term:

$$\frac{t_c B}{A} \overset{!}{=} 1 \qquad \Longrightarrow \qquad t_c := \frac{A}{B} \tag{7}$$

Now we can rewrite the given equation for population dynamics in its dimensionless form:

$$\frac{\mathrm{d}n}{\mathrm{d}\tau} = \frac{rA}{B}\, n - \frac{rA^2}{BK}\, n^2 - \frac{1}{1+n^{-2}} \tag{8}$$

By defining the dimensionless parameter $D := rA/B$, we can further simplify the dimensionless equation:

$$\frac{\mathrm{d}n}{\mathrm{d}\tau} = Dn - \frac{DA}{K}\, n^2 - \frac{1}{1+n^{-2}} \qquad \text{with} \qquad D := \frac{rA}{B} \tag{9}$$

Stationary points

Task: Determine the stationary points n^* for $K/A = 7.3$. Note that for $n^* \neq 0$ these values are solutions of a cubic equation; it depends on n and the remaining free parameter. The cubic equation should be derived by yourself analytically; its zero points you can obtain numerically / graphically by using e.g. *Mathematica*. When do one or three real solutions exist as a function of the remaining free parameter? (Hint: we do not ask for some analytical formula here! It is enough to vary the free parameter and check using *Mathematica* which three solutions for the stationary points you get; as stationary points only real solutions are valid. Only one digit after the comma is enough, in other words you vary the free parameter by about 0.05.).
Which of the stationary points is stable and unstable?

Definition: A fixed point (FP) or stationary point of a differentiable function of one variable is a point on the graph of the function where the function's derivative is zero.

According to the definition for stationary points

$$n = n^* \text{ stationary point of } n = n(\tau) \qquad \Longleftrightarrow \qquad \left.\frac{\mathrm{d}n}{\mathrm{d}\tau}\right|_{n=n^*} = 0 \tag{10}$$

we have to determine the zero points of the dimensionless equation for population dynamics:

$$\frac{\mathrm{d}n}{\mathrm{d}\tau} = 0 \qquad \xLeftrightarrow{\text{Eq. (9)}} \qquad 0 = Dn - \frac{DA}{K}\, n^2 - \frac{1}{1+n^{-2}} \qquad \text{with} \qquad D := \frac{rA}{B} \tag{11}$$

For $n \neq 0$ we can transform the above equation into a cubic equation:

$$\begin{aligned}
& 0 \overset{!}{=} Dn - \frac{DA}{K}\, n^2 - \frac{1}{1+n^{-2}} && \Big| \cdot (1+n^{-2}) && (12)\\
\Leftrightarrow \quad & = D(1+n^{-2})n - \frac{DA}{K}\,(1+n^{-2})n^2 - 1 && && (13)\\
\Leftrightarrow \quad & = D(n+n^{-1}) - \frac{DA}{K}\,(n^2+1) - 1 && && (14)\\
\Leftrightarrow \quad & = D\left(\frac{n^2+1}{n}\right) - \frac{DA}{K}\,(n^2+1) - 1 && \Big| \cdot n && (15)\\
\Leftrightarrow \quad & = D(n^2+1) - \frac{DA}{K}\,(n^3+n) - n && && (16)\\
\Leftrightarrow \quad & = -\frac{DA}{K}\, n^3 + Dn^2 - \frac{DA}{K}\, n - n + D && && (17)\\
\Leftrightarrow \quad & = -\frac{DA}{K}\, n^3 + Dn^2 - \left(\frac{DA}{K} + 1\right) n + D && && (18)
\end{aligned}$$

Finally, we multiply the entire equation by $-K/(DA)$. So we get that for $n \neq 0$ the stationary points n^* are solutions of a cubic equation:

$$0 \overset{!}{=} n^3 - \frac{K}{A}\,n^2 + \left(1 + \frac{1}{D}\frac{K}{A}\right) n - \frac{K}{A} \tag{19}$$

If we set $K/A = 7.3$ as required in the task, the cubic equation above only depends on n and the remaining free parameter $D = rA/B$:

$$0 \overset{!}{=} n^3 - 7.3\,n^2 + \left(1 + \frac{7.3}{D}\right) n - 7.3 \tag{20}$$

Let's have a look at the asymptotic behavior of the function:

$$\lim_{n\to-\infty}\left[n^3 - 7.3\,n^2 + \left(1 + \frac{7.3}{D}\right) n - 7.3\right] = -\infty \tag{21}$$

$$\lim_{n\to+\infty}\left[n^3 - 7.3\,n^2 + \left(1 + \frac{7.3}{D}\right) n - 7.3\right] = +\infty \tag{22}$$

Due to the sign change in the asymptotic behavior for $n \to \pm\infty$, the function has at least one real zero point for every value of the parameter $D = rA/B$. We use the following procedure to determine the zeros numerically using `Python`:

- We first determine a zero point using the `SciPy` function `brentq`. The `brentq` method finds a zero point of a given function in a bracketing interval using Brent's method.
- If we know a zero point of the cubic equation, we can use polynomial division to transform the cubic equation into a quadratic equation.
- The zero points of the quadratic equation can now be calculated using the quadratic formula.

Using the quadratic formula and the coefficients of the quadratic equation, we can directly determine the number of real zero points. This allows us to vary the free parameter D and to see for which values of D one or three real solutions to the function exist (see Python-Code 1).

Python-Code 1: Numerical determination of the stationary points

```python
# -*- coding: utf-8 -*-
"""
Introduction to Computational Physics
- Exercise 07:  Population Dynamics - Stationary Points
- Group: Simon Groß-Bölting, Lorenz Vogel, Sebastian Willenberg
"""

import numpy as np; import matplotlib.pyplot as plt
from scipy.optimize import brentq

def cubic_function(n,coefficients):
    ''' Cubic function where n represents an unknown (variable)
        and a, b, c and d represent known numbers (coefficients) '''
    a,b,c,d = coefficients[0],coefficients[1],coefficients[2],coefficients[3]
    return (a*n**3)+(b*n**2)+(c*n)+d

def quadratic_formula(coefficients):
    ''' Function to compute the solutions of a reduced quadratic equation
        using the quadratic formula '''
    p = coefficients[1]/coefficients[0]
    q = coefficients[2]/coefficients[0]
    if ((p/2)**2 >= q): # as stationary points only real solutions are valid
        n2 = -(p/2)+np.sqrt((p/2)**2-q)
        n3 = -(p/2)-np.sqrt((p/2)**2-q)
        return (n2,n3)
    else: return None

def count_zeros(coefficients):
    ''' Function to compute the number of real zeros of a quadratic equation
        Input:  NumPy-Array with the coefficients of the quadratic equation
        Output: number of real zeros of the given quadratic equation '''
    p = coefficients[1]/coefficients[0]
    q = coefficients[2]/coefficients[0]
    if ((p/2)**2 < q):
        # in this case the square root becomes imaginary and
        # the quadratic equation has no real zero points
        return 0
    elif ((p/2)**2 == q):
        # in this case the square root becomes zero and
        # the quadratic equation has a real 'double zero point'
        return 1
    else: return 2

def find_solutions(D):
    ''' Function to answer the question: When do one or three real solutions
        exist as a function of the remaining free parameter?
        We vary the free parameter D and check for which value of D we
        get one or three real solutions
        Input:  starting value for the free paramter D '''

    # We use the coefficients of the cubic equation, which
    # we have already derived analytically
    coefficients = np.array([1.,-7.3,1+(7.3/D),-7.3])

    # compute first zero point of the cubic equation numerically
    n1 = brentq(cubic_function, -2., 8., args=coefficients)

    # compute the quadratic function using polynomial division
    quadratic_function = np.polydiv(coefficients, np.array([1.,-n1]))[0]

    D_lower = D; D_upper = D
    while count_zeros(quadratic_function) != 2:
        D += 1e-3   # increase free paramter
        coefficients = np.array([1.,-7.3,1+(7.3/D),-7.3])

        # compute first zero point of the cubic equation numerically
        n1 = brentq(cubic_function,-2.,8.,args=coefficients)

        # compute the quadratic function using polynomial division
        quadratic_function = np.polydiv(coefficients, np.array([1.,-n1]))[0]
        D_lower = D

```

```
    while count_zeros(quadratic_function) == 2:
        D_upper = D; D += 1e-3   # increase free paramter
        coefficients = np.array([1.,-7.3,1+(7.3/D),-7.3])

        # compute first zero point of the cubic equation numerically
        n1 = brentq(cubic_function,-2.,8.,args=coefficients)

        # compute the quadratic function using polynomial division
        quadratic_function = np.polydiv(coefficients, np.array([1.,-n1]))[0]

    return (D_lower,D_upper)

# We determine the stationary points for K/A = 7.3
# The stationary points are solutions of a cubic equation; it depends on
# the variable n and the remaining free parameter D
D = find_solutions(0.1)
n = np.linspace(0,5,1000)
print(D)


fig, ax = plt.subplots()
ax.set_title('Titel')
ax.set_xlabel(r'$n$'); ax.set_ylabel(r'$\frac{\mathrm{d}n}{\mathrm{d}\tau}$')

coefficients = np.array([1.,-7.3,1+(7.3/D[0]),-7.3])
ax.plot(n,cubic_function(n,coefficients), 'r.', markersize=1, label='1')
coefficients = np.array([1.,-7.3,1+(7.3/D[1]),-7.3])
ax.plot(n,cubic_function(n,coefficients), 'g.', markersize=1, label='1')
coefficients = np.array([1.,-7.3,1+(7.3/(0.5*(D[0]+D[1]))),-7.3])
ax.plot(n,cubic_function(n,coefficients), 'b.', markersize=1, label='1')

ax.grid(); ax.legend(loc='best', markerscale=8)
ax.set_xlim((0,5)); ax.set_ylim((-7.5,12.5))
#fig.savefig('figures/Eigenvalues_Probability-Amplitude.pdf', format='pdf')
plt.show(); plt.clf(); plt.close()
```