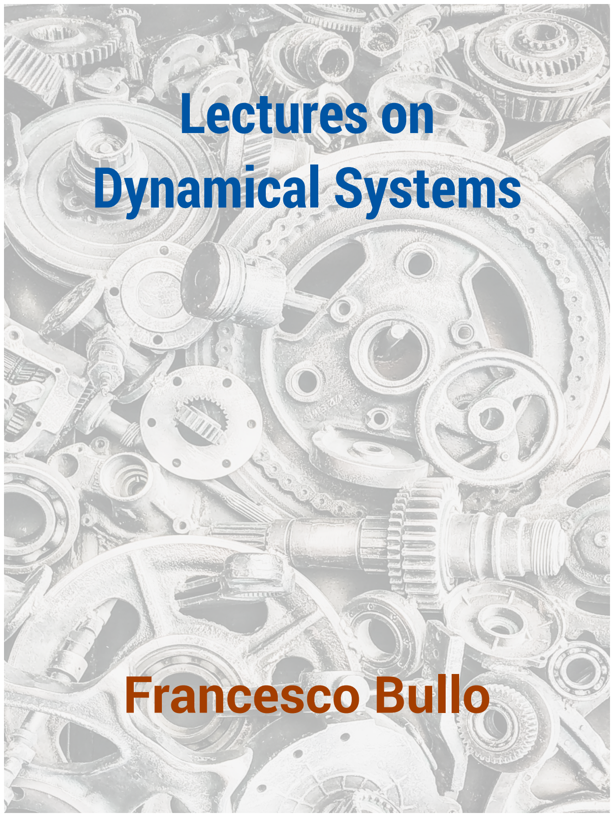
UC Santa Barbara, Department of Mechanical Engineering

ME103 Dynamical Systems. Slides by Chapter.

Francesco Bullo

http://motion.me.ucsb.edu/ME103-Fall2024/syllabus.html



Contents

1 Dynamical Systems: Definition and Scientific Examples **3**
1.1 Introduction 4
1.2 One dimensional examples: exponential and logistic growth 8
1.3 A two-dimensional dynamical system with periodic evolutions 20
1.4 Visualization of dynamical systems 22
1.5 Summary 27
1.6 Historical notes, further reading, and online resources 29
1.7 Appendix: The logistic equation in business management and sociology 31
1.8 Appendix: A bistable toggle switch in synthetic biology 32
1.9 Exercises 35

Bibliography **44**

Chapter 1

Dynamical Systems: Definition and Scientific Examples

1.1 Introduction

Dynamical systems frequently emerge in science and engineering, where they model a wide range of phenomena. These systems can describe anything from planetary motion and the oscillation of a pendulum to the growth dynamics of a bacterial population or fluctuations in stock market prices.

Figure 1.1: Long-exposure sky photo, sourced from `https://unsplash.com`. The movement of the stars across the night sky has captivated human imagination since the birth of mankind. This image is a static visualization of a periodic behavior.

Loosely speaking, a dynamical system is a way to understand how things change over time based on rules.

Definition 1.1. *A dynamical system is a mathematical concept that describes how variables of interest changes over time.*
A dynamical system is a set of rules or equations that dictates how a system's state evolves from one moment to the next.

The *state* of a dynamical system is a set of variables that completely describe the current condition or position of the system. For example,

(i) for a mechanical pendulum, the state might be the angle and angular velocity of the pendulum,

(ii) for an electric circuit, the state could be the voltage across a capacitor or the current through an inductor,

(iii) in a fluid dynamics problem, the state might include variables such as pressure, temperature, velocity, and density at every point in a fluid.

The number of state variables is the *dimension of the system*.

Knowledge of the state at a given time and knowledge of the system's governing equations allow one to predict the system's future behavior. Typically:

(i) *all dynamical systems can be described by ordinary differential equations (ODEs) and their variations* (e.g., partial differential equations for variables defined over a spatial domain, or difference equations for systems in discrete time),

(ii) given an initial condition (and, if present, an external input stimulus), *there exists one unique solution* (aka evolution, trajectory) originating from it, and

(iii) the behavior of the dynamical system is affected by values of *a single or multiple parameters*.

Examples in mechanical engineering

Many dynamical systems of interest in mechanical engineering exhibit rich and complex behavior. Here are some examples that are widely studied because of their importance or interesting behavior:

- The classic *mass-spring-damper system* models a wide range of structures and electro-mechanical systems. Mass-spring-damper systems exhibit underdamped or overdamped responses.
- The *double pendulum* is a classic example of a system with chaotic behavior. Even with two simple links, its motion can become highly unpredictable and sensitive to initial conditions.
- *Vibrating beams and plates*, such as those found in structures or mechanical components, exhibit complex dynamics, especially when subject to large deflections or nonlinear boundary conditions.
- *Rotating systems* like turbines or helicopter blades involve gyroscopic effects and can exhibit nonlinearities, whirling, and even chaotic vibrations under certain operating conditions.
- *Fluid-structure interactions*, such as aeroelastic flutter in aircraft wings or vortex-induced vibrations in bridges and pipelines, show rich dynamics due to the coupling between fluid flow and structural response.
- *Multi-link robotic arms* with flexible joints and components can exhibit rich nonlinear dynamics, especially when considering real-world effects like joint friction, flexible elements, and time-varying forces.
- *MEMS devices*, which operate on a micro-scale, often show nonlinear dynamics due to electrostatic forces, material nonlinearities, and damping effects, leading to intricate vibrational behaviors.

1.2 One dimensional examples: exponential and logistic growth

We consider two simple intuitive examples describing the *dynamics of populations in ecology*. Both examples are 1-dimensional, in the sense that the system state is described by only a single scalar variable. Both examples systems are modeled using differential equations.

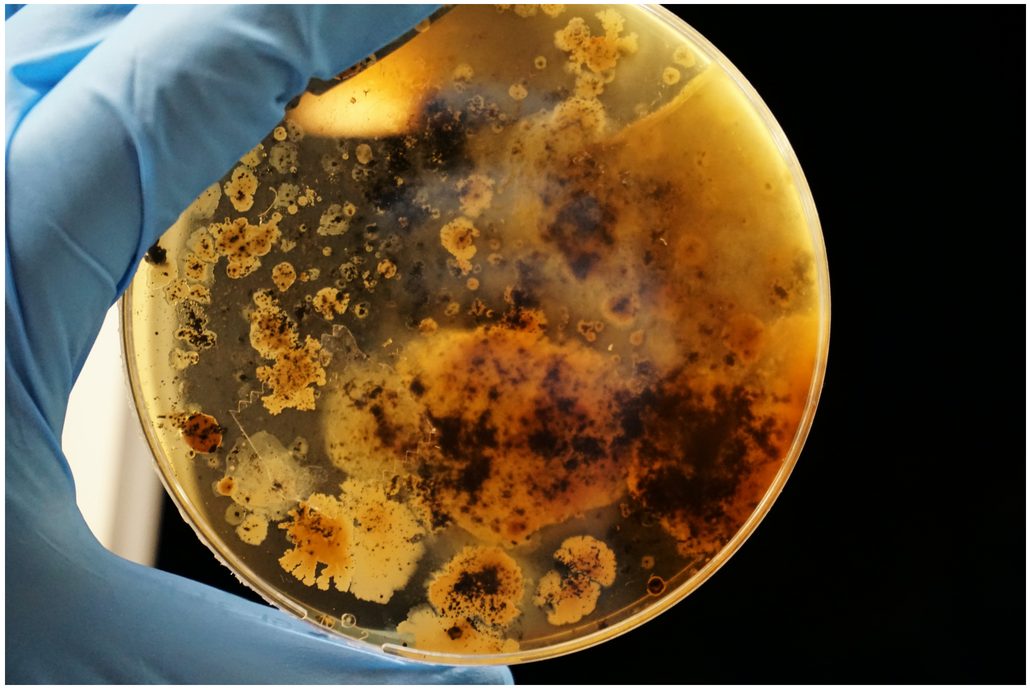


Figure 1.2: Bacterial colonies are shown growing on a plate, starting from just a few cells and expanding rapidly. The colony grows at an exponential rate, but this growth slows as resources become limited. Image sourced from `https://unsplash.com`.

1.2.1 A first model: the linear growth/decay model

Let us now make a number of assumptions to obtain a quantitative model:

(i) we let t denote time, the independent variable;

(ii) let $x(t)$ denote the *number*[1] *of individuals* in a population at time t (or a suitably scaled version of this quantity);

(iii) assume that each individual has a given chance of reproducing or dying, so that the *birth rate* and *death rate* are proportional $x(t)$, that is,

$$\text{birth rate}(t) = r_{\text{birth}} x(t) \quad \text{and} \quad \text{death rate}(t) = r_{\text{death}} x(t),$$

where $r_{\text{birth}} > 0$ and $r_{\text{death}} > 0$ are average *per-individual birth/death rate* and are assumed to be constant and positive;

(iv) we compute $\text{birth rate}(t) - \text{death rate}(t) = \big(r_{\text{birth}} - r_{\text{death}}\big) x(t)$ and define $r = r_{\text{birth}} - r_{\text{death}}$.

[1] Since the population size $x(t)$ is assumed to be sufficiently large, treating it as a continuous variable rather than as an integer is an acceptable approximation.

In summary, the *linear growth/decay model* is

$$\dot{x}(t) = rx(t), \tag{1.1}$$

where the parameter r is *per-individual growth rate*.

- This system has 1 independent variable t, 1 dependent variable $x(t)$, and 1 parameter r.
- We call $x(t)$ the *state of the system* and r a *parameter of the system*.
- Since there is only 1 independent variable and the dependent variable is differentiated only once the system is a 1st order ordinary differential equation.

Mathematical analysis of the linear growth/decay model

Note that the model (1.1) is linear, in the sense that the right hand size depends upon the state x in a linear way. For such a linear model, the solution is straightforward:

$$x(t) = \mathrm{e}^{rt}\, x(0). \tag{1.2}$$

This simple model makes the following three *predictions* about the long-term evolution of the population:

(i) if $r < 0$, then the population will become extinct,

(ii) if $r = 0$, then the population remains constant, and

(iii) if $r > 0$, then the population will experience a demographic explosion (exponential growth, to be precise).

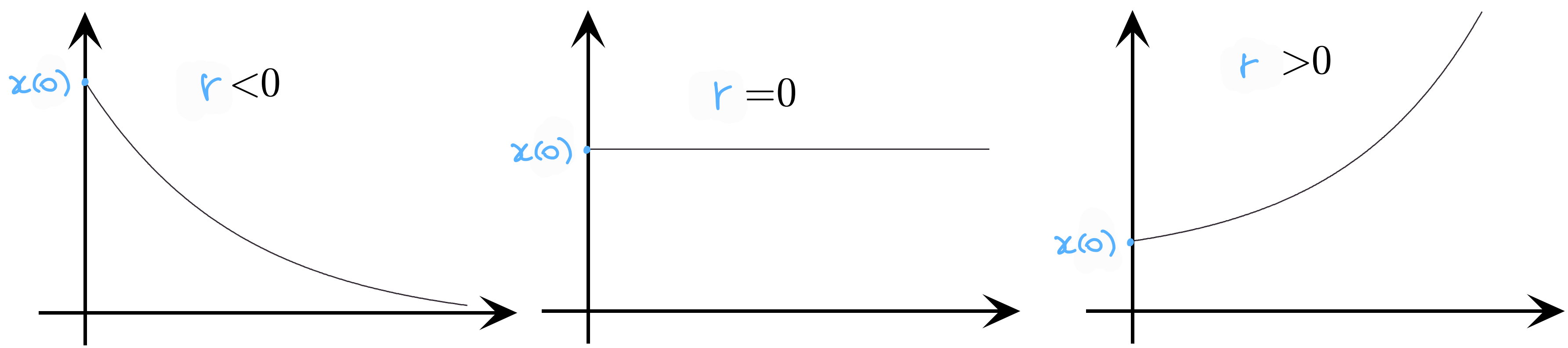


Figure 1.3: Solutions to the linear growth model in equation (1.2).

1.2.2 A second model: the logistic equation

Next, we consider a slightly more realistic population dynamics model. We start by observing that, for $r > 0$, exponential unbounded growth is unreasonable in an environment with finite resources. Therefore, it makes sense to make birth and death rates dependent on the population $x(t)$.

Specifically, we assume that the per-individual growth rate is a decreasing function of $x(t)$, meaning that, *the larger the population, the fewer are the available resources*. Instead of the linear model rx, a simple model for this *diminishing growth phenomenon* is

$$\text{per-individual growth rate} = r\Big(1 - \frac{x(t)}{K}\Big), \tag{1.3}$$

where the parameter K defines a *threshold behavior*:

- growth is positive for $x(t) < K$,
- growth is zero for $x(t) = K$, and
- growth turns negative, hence growth turns into decay, for $x(t) > K$.

This threshold behavior indicates the existence of a critical population threshold separating the situation of a positive rate (increasing population) from that of a negative rate (decreasing population).

In summary, the *logistic growth model* is

$$\dot{x}(t) = r\left(1 - \frac{x(t)}{K}\right)x(t) = \underbrace{rx(t)}_{\text{linear growth}} + \underbrace{\left(-\frac{r}{K}\right)x_i^2(t)}_{\text{logistic correction}}. \tag{1.4}$$

with the parameters:

- r is the *intrinsic growth rate* of the population: how fast the population would grow without any constraints, and
- K is the *carrying capacity* of the environment: the maximum population that can be sustained over the long term.

Note: the logistic growth model (1.4) is a *nonlinear dynamical system*, whereas the growth/decay model (1.1) is a *linear dynamical system*.

Numerical analysis of the logistic growth model

We now perform a numerical analysis, i.e., a simulation-based analysis of the logistic equation.

```python
# Python libraries
import numpy as np; import matplotlib.pyplot as plt;
from scipy.integrate import odeint

# Logistic equation
def logistic_equation(x, t, r, K):
    return r * x * (1 - x / K)

# Parameters
r = 0.1  # growth rate
K = 100  # carrying capacity

# Initial conditions and time range of integration
initial_conditions = [1, 5, 10, 25, 50, 75, 125, 150, 175]
colors = ['#752d00', '#a43e00', '#d35000', '#ff6100', ...
    '#ff8800', '#ffaf00', '#ffcc00']

T = 125  # maximum time
dt = 0.1  # time step
times = np.arange(0, T, dt)  # array of time values

# Numerically solve the ODE and plot solution
plt.figure(figsize=(10, 6))
for i, x0 in enumerate(initial_conditions):
    solution = odeint(logistic_equation, x0, times, args=(r, K))
    plt.plot(times, solution, label=f'Initial condition: ...
        {x0}', color=colors[i % len(colors)])

plt.title("Numerical solutions to the logistic equation")
plt.xlabel("Time t"); plt.ylabel("Population size x")
plt.legend(); plt.grid(True);
plt.xlim(0, 120); plt.ylim(0, 180)

# Save to PDF
plt.savefig("logistic-equation.pdf", bbox_inches='tight')
```

Listing 1.1: Python script generating Figure 1.4. Available at: logistic-equation.py

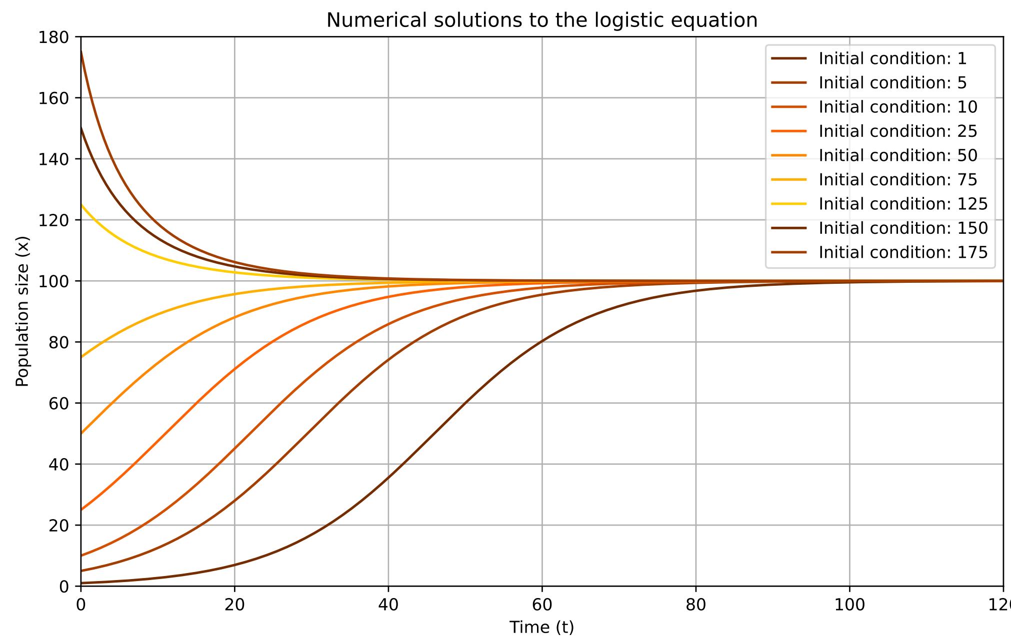


Figure 1.4: Solutions from multiple initial conditions $x(0)$ to the logistic equation (1.4): $\dot{x} = r(1 - x/K)x$, for $r = 0.1$ and $K = 100$.
Note:
(i) For each $x(0) > 0$, $x(t)$ approaches the carrying capacity K as $t \to \infty$.
(ii) Each solution with $0 < x(0) < K$ is monotonically increasing, whereas each solution with $K < x(0)$ is monotonically decreasing.
(iii) For $x(0) = 0$, the solution is the equilibrium solution 0. We do not consider negative initial conditions.

In class assignment

What happens to the solutions to initial conditions $x(0)$ and $x(K)$?
Do solutions have inflection points?

Mathematical analysis of the logistic equation

Beside the numerical analysis (based upon numerical integration and visualization), it is useful to perform some mathematical analysis of the logistic equation $\dot{x}(t) = r\left(1 - \frac{x(t)}{K}\right)x(t)$. Let us introduce some useful concepts.

Definition of equilibrium point for a dynamical system: For a dynamical system $\dot{x} = f(x)$

(i) a point x^* is an *equilibrium point* if it is a solution to the *equilibrium equation* $f(x^*) = 0$,

(ii) when x^* is an equilibrium point, the constant curve $x(t) = x^*$ is a solution of the dynamical system,

(iii) an equilibrium point x^* is *stable* if nearby trajectories converge to x^*,

(iv) an equilibrium point x^* is *unstable* if nearby trajectories move away from x^*.

The equilibrium points of the logistic equation: The logistic equation model (1.4) has two equilibrium points, corresponding to the solutions of the equation $a(1 - \frac{x}{K})x = 0$, which are

$$x_1^* = 0 \quad \text{and} \quad x_2^* = K. \tag{1.5}$$

At x_1^* we have extinction of the population and at x_2^* we are at carrying capacity.

$x_1^* = 0$ is unstable

$x_2^* = K$ is stable

Remark 1.2. *As demonstrated in Exercise E1.1, the exact solution to the logistic equation is given by*

$$x(t) = \frac{K x_0 \, \mathrm{e}^{rt}}{K + x_0(\mathrm{e}^{rt} - 1)}, \tag{1.6}$$

where x_0 represents the population at time $t = 0$, i.e., $x(0) = x_0$. From this equation, it is possible to understand when the solution has an inflection point.

- *Equation (1.6) provides a closed-form solution for the logistic growth model (1.4),*
- *In general, finding a closed-form solution for a nonlinear dynamical system is not possible. As a result, one typically relies either on numerical simulations or on qualitative analysis techniques.*
- *In this text, we will primarily focus on linear dynamical systems that are more tractable and for which solutions can be derived analytically.* •

The structure of a numerical integrator and visualization program

Here is a breakdown of what the Python script does:

(i) loads useful software libraries
(ii) defines the differential equation (ODE) describing the dynamics
(iii) defines values for all parameters
(iv) defines a set of initial conditions and a time range
(v) computes numerical solutions to the ODE, e.g., via the routine `odeint`
(vi) plots the component(s) of the solution as a function of time
(vii) saves the plot to a PDF file.

Now, to be precise, it is important to clarify: `odeint` computes only a numerical approximation to the precise solution. Useful reading: consult the manual and options for `odeint`.

Programming notes

It is useful to briefly compare Python and Matlab:

- Python is a general-purpose programming language. Any editor can be used to write a Python script. Matlab is a high-level language and interactive environment for numerical computations. Matlab is designed for engineers and scientists, with an intuitive interface and simplified programming syntax.
- Python is open-source and, as such, it will remain at your disposal on any arbitrary device and operating systems. The open-source community provides extensive resources, tutorials, and documentation.
- Since Python is general purpose, it is perceived to have a steeper learning curve than Matlab. However, large-language models (such as ChatGPT) provide a simple way to generate scripts that can function at initial working prototypes and can then be tweaked.
- Numerous leading libraries are freely available to enhance Python scripts. For example, Matplotlib is a Python library for creating static, animated, and interactive visualizations. Other notable examples in machine learning and data science include Pandas, TensorFlow, and PyTorch, and in robotics include the Robot Operating System.
- To draw phase portraits, these lecture notes adopt the function in Python's Matplotlib library. Similar outcomes are possible with the `streamline` function in Matlab.

  The streamplot of a vector field is a visual representation of the vector field. It provides a snapshot of the direction and magnitude of the field at various points in the phase space. The streamlines are not guaranteed to represent exact trajectories, but give a qualitative idea about them.

1.3 A two-dimensional dynamical system with periodic evolutions

Figure 1.5: The Canadian lynx (*Lynx canadensis*) is a major predator of the snowshoe hare (*Lepus americanus*). Historical records of animals captures indicate that the lynx and hare numbers rise and fall periodically; see (Odum, 1959).
Public domain image from Rudolfo's Usenet Animal Pictures Gallery (no longer in existence).

In this section we consider the classic predator-prey dynamical model. In this model, we let x_1 denote the number of preys and x_2 denote the number of predators. The *Lotka-Volterra predator-prey model* in the variables x_1 and x_2 is:

$$\begin{aligned} \dot{x}_1 &= \alpha x_1 - \beta x_1 x_2, \\ \dot{x}_2 &= -\delta x_2 + \gamma x_1 x_2, \end{aligned} \tag{1.7}$$

with the parameters:

- $\alpha > 0$ is the intrinsic growth rate of the prey,
- $\delta > 0$ is the intrinsic decay rate of the predator,
- $\beta > 0$ and $\gamma > 0$ describe the inter-species interactions.

For general values of the parameters, no simple closed form solution is available. Therefore, we perform a numerical analysis.

Numerical analysis of the Lotka-Volterra predator-prey model

```python
# Python libraries
import numpy as np; import matplotlib.pyplot as plt
from scipy.integrate import odeint

# Define the Lotka-Volterra equations:
def lotka_volterra(z, t, alpha, beta, gamma, Δ):
    x, y = z
    dxdt = alpha * x - beta * x * y
    dydt = -Δ * y + gamma * x * y
    return [dxdt, dydt]

# Parameters:
alpha = 0.1; beta = 0.02; gamma = 0.01; Δ = 0.1

# Solve the equations:
initial_condition = (20, 5)
T = 200; dt = 0.1; times = np.arange(0, T, dt)
solution = odeint(lotka_volterra, initial_condition, times, ...
    args=(alpha, beta, gamma, Δ))

# Plot the solution as a function of time and save to PDF
plt.figure(figsize=(10, 5));
plt.plot(times, solution[:, 0], label='Prey')
plt.plot(times, solution[:, 1], label='Predator')
# plt.title("Lotka-Volterra Predator-Prey Model: sample solutions ...
    (population sizes vs time) and phase portrait")
plt.xlabel("Time (t)"); plt.ylabel("Population Size")
plt.legend(); plt.grid(True); plt.xlim(0, 200); plt.ylim(0, 22.25)
plt.savefig("lotka-volterra-predator-prey.pdf", bbox_inches='tight')

# Phase portrait
u, v = np.meshgrid(np.linspace(0, 25, 20), np.linspace(0, 15, 20))
du, dv = lotka_volterra([u, v], 0, alpha, beta, gamma, Δ)
magnitude = 2*np.sqrt(du**2 + dv**2)

plt.figure(figsize=(10,6)); plt.xlim(0, 25); plt.ylim(0, 15)
plt.streamplot(u, v, du, dv, density=.95, ...
    color='#0085ff',linewidth=magnitude)
# plt.streamplot(u, v, du, dv, density=.5, ...
    color='#0085ff',linewidth=magnitude, broken_streamlines=False)

# Adding the specific solution trajectory to the phase portrait
plt.plot(solution[:, 0], solution[:, 1], color='#752d00', ...
    label='Trajectory from (20,5)')
plt.scatter(10, 5, color='black', s=50, zorder=5)

# plt.title('Phase Portrait of the Lotka-Volterra Predator-Prey Model');
plt.xlabel('prey population x1'); plt.ylabel('predator population x2'); ...
    plt.grid(True)
plt.savefig("lotka-volterra-predator-prey-phase.pdf", bbox_inches='tight')
```

Listing 1.2: Python script generating Figure 1.6. Available at lotka-volterra-predator-prey.py

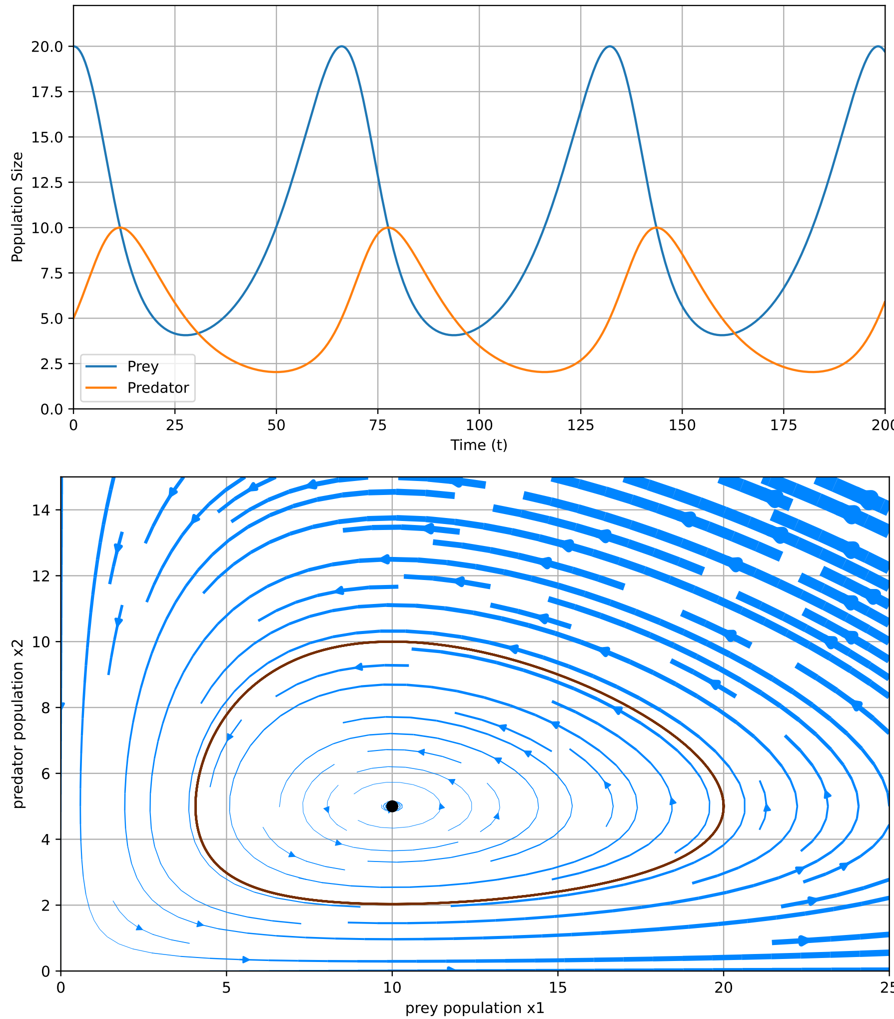


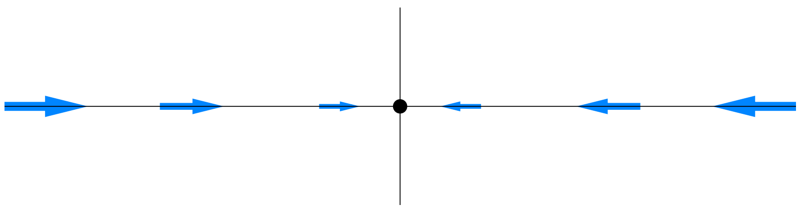
Figure 1.6: Solution and phase portrait for the Lotka-Volterra predator-prey dynamics (1.7).
The black line is the solution originating from initial condition $(20, 5)$.

1.4 Visualization of dynamical systems

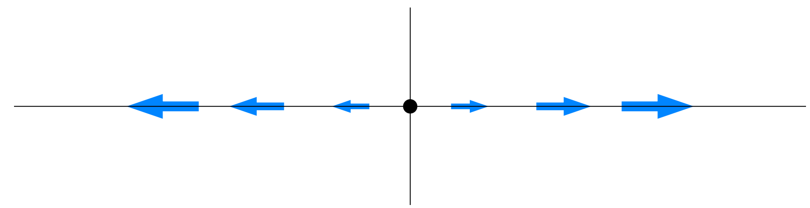
A 1-dimensional ODE defines a *line of vectors* on the real line. Specifically, for a 1-dimensional system,[2] $\dot{x} = f(x)$ with $x \in \mathbb{R}$,

- if $f(x) < 0$, then the solution starting at x moves to the left,
- if $f(x) = 0$, then the solution starting at x does not move, since the point x is an equilibrium, and
- if $f(x) > 0$, then the solution starting at x moves to the right.

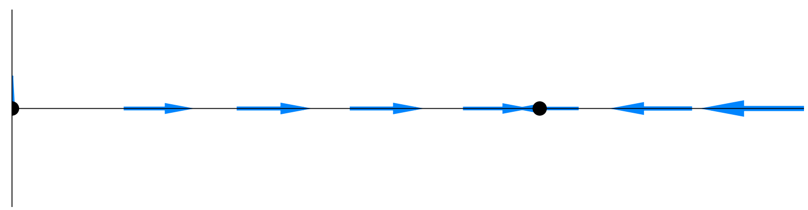
We show this line of vectors for a few simple examples in Figure 1.7.



(a) Linear decay: $\dot{x} = -x$ (that is, $\dot{x} = rx$ with $r = -1$).
Note: 0 is a stable equilibrium.



(b) Linear growth: $\dot{x} = x$ (that is, $\dot{x} = rx$ with $r = +1$).
Note: 0 is an unstable equilibrium.



(c) Logistic equation: $\dot{x} = x - x^2$ (for $x > 0$, $r = 1$, $K = 1$.)

Figure 1.7: Line of vector of simple 1-dimensional ODEs.

[2]We let $\mathbb{R}$ denote the set of real numbers and $\mathbb{R}^d$ denote the set of column vectors with d real entries.

Next, we consider a generic 2-dimensional ODE:

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2), \\ \dot{x}_2 &= f_2(x_1, x_2). \end{aligned} \tag{1.8}$$

This 2-dimensional ODE defines a *vector field* on the plane in the sense:

at each point (x_1, x_2) on the plane, we compute the vector with components $(f_1(x_1, x_2), f_2(x_1, x_2))$, this is the velocity vector of a particle at point (x_1, x_2) that is flowing according to the ODE (1.8).

By flowing according or along the ODE, the particle traces out a curve, that is a solution of the ODE.

For clarity,

- an ODE is the same thing as a vector field,
- the plane is filled with trajectories, since there is a solution to the ODE from each point in the plane, and
- trajectories never intersect because of the uniqueness of solutions to differential equations.

The *phase portrait* (aka *trajectory diagram*) is a graphical representation that shows the trajectories that a 2-dimensional dynamical system can take through the plane. It is an illustrations of the paths that a particle obeying the ODE would follow. We illustrate how to draw the phase portrait in Figure 1.8.

Once the phase portrait is plotted (or its numerical approximate version, the `streamplot` in Python), one can recognize some special solutions and some qualitative features of many solutions:

(i) an equilibrium, including an attractive equilibrium, or just marginally stable, or an unstable,

(ii) a closed orbit = periodic trajectory, or

(iii) a diverging trajectory.

Note: Generalizing from the plane, the *state space* is a multidimensional space in which each axis corresponds to one of the system's variables. But it is not clear how to draw a phase portrait in dimensions larger than 2.

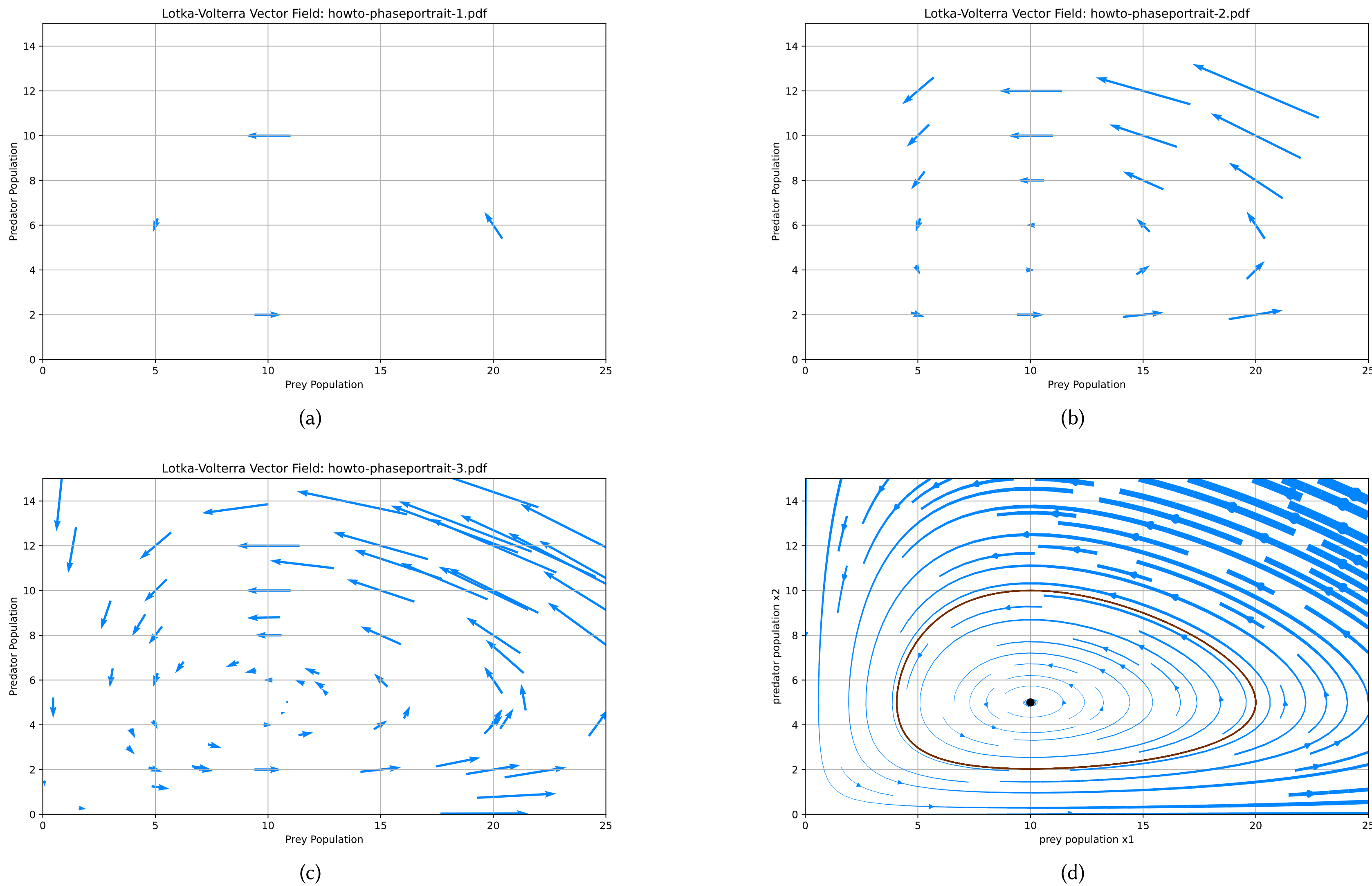


Figure 1.8: In the 2-dimensional Lotka-Volterra predator-prey example, the state space is (x_1, x_2) and the phase portrait from Figure 1.6 is reported here. Relevant questions are: How many equilibria are there? Are they stable? Is there any trajectory diverging to infinity? Is there any trajectory converging to an equilibrium?

Example phase portraits

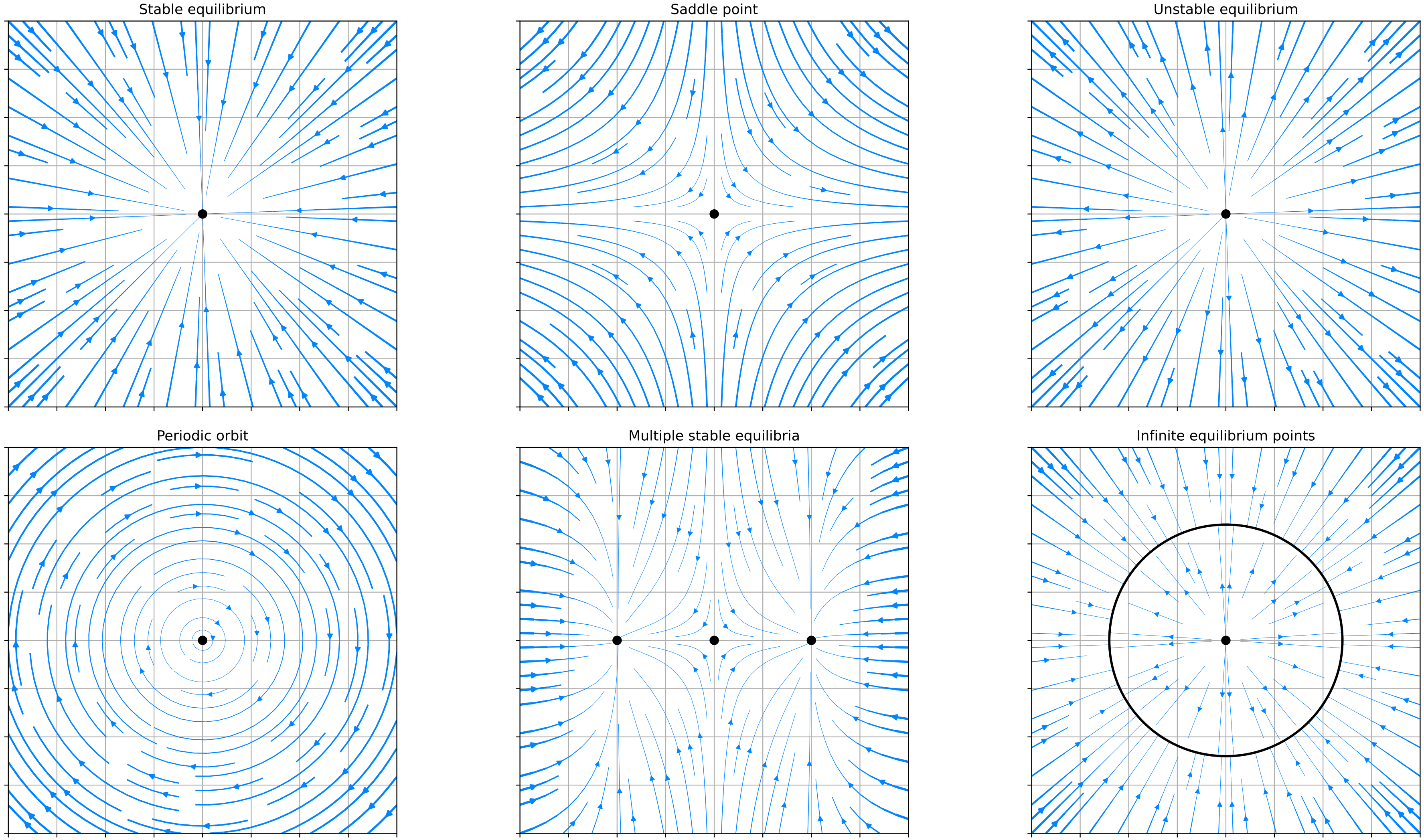


Figure 1.9: Examples of phase portraits.

First row: $\begin{cases} \dot{x} = -x \\ \dot{y} = -y \end{cases}$, $\begin{cases} \dot{x} = x \\ \dot{y} = -y \end{cases}$, and $\begin{cases} \dot{x} = x \\ \dot{y} = y \end{cases}$.

Second row: $\begin{cases} \dot{x} = y \\ \dot{y} = -x \end{cases}$, $\begin{cases} \dot{x} = x(1-x^2) \\ \dot{y} = -y \end{cases}$, and $\begin{cases} \dot{x} = -x(1-(x^2+y^2)) \\ \dot{y} = -y(1-(x^2+y^2)) \end{cases}$.

1.5 Summary

In summary, in this chapter we saw:

Systems

- examples of first and second order dynamical systems
- examples of linear and nonlinear systems
- distinction between a variable and a parameter

Phenomena

- an equilibrium point is a solution where the state is constant
- solutions may diverge to infinity, may converge to an equilibrium point, or may be periodic
- additional phenomena may occur: chaos, synchronization, entrainment, etc

Analysis methods

- mathematical analysis (obtain solution in closed form; possible only some times)
- numerical analysis (numerical integration of differential equations, Python as a flexible open-source language and environment)

Content of these lecture notes:

In the first two parts of this course, we will study:
systems: linear dynamical systems relevant in mechanical engineering,
phenomena: equilibria, stability, responses to basic inputs, vibrations
analysis: the Laplace transform and transfer functions
In the third part of this course, we will study how to control dynamical systems, i.e., how to shape the dynamics of systems.

1.6 Historical notes, further reading, and online resources

Studying dynamical systems was once a perilous pursuit! Galileo Galilei (1564–1642) famously faced severe consequences from the Catholic Church for advocating the heliocentric model, a once-revolutionary idea that the Earth revolves around the Sun. This view directly challenged the long-held geocentric beliefs of Aristotle (384–322 BC) and Ptolemy (c. AD 100–170), which had been adopted and defended by the Church. Galileo's support for the heliocentric theory not only contradicted centuries of philosophical and scientific tradition, but also sparked a broader conflict between emerging scientific thought and religious authority.

For further reading, a wonderful reference is the textbook (Strogatz, 2015) and, specifically, Chapters 2 "Flows on the line" and 6 "Phase plane". Numerous additional example systems can be found in Ogata (2003), in (Åström and Murray, 2021, Chapter 3) and throughout (Strogatz, 2015). Extensive additional information is available on wikipedia, for example, see the page on the logistic equation.

An online application that draws phase portraits (more accurate than the Python's `streamplot`) is "Phase Plane - Nonlinear System with Nullclines" by Patrick Davix.

Interesting youtube videos (links are functional as of Sep 2024):

- Mechanical systems
  - simple harmonic motion and London's Millennium Bridge (9m 10s) and mathematics of harmonic motions (8m 9s)
  - damping and resonance (5m 3s) and safety in skyscrapers (tuned mass damper) (9m 36s)
  - the double pendulum and the chaos phenomenon (short)
  - resonance and the 1940 Tacoma bridge collapse
- Biology
  - simplest genetic circuit: the toggle switch (9m 49s)
  - a white blood cell chases bacteria in real life (29s)
- Ecology
  - schooling and collective intelligence in animals (with intervention by Ian Couzin, 8m 42s)
  - why do fish swim (3m 48s), and why do they swim in harmony (6m 6s)
- Coupled oscillators
  - synchronizing oscillators (short)
  - synchronization out of chaos (with intervention by Stephen Strogatz, 20m 57s)
- Traffic networks
  - traffic causes and control strategies (4m 29s)

1.7 Appendix: The logistic equation in business management and sociology

The logistic equation studied in Section 1.2.2 models not only population dynamics but also *technology adoption*, *spread of innovations*, and other diffusion phenomena that follow an S-curve pattern. (In some cases, the state $x(t)$ is only taken in the range $[0, K]$.)

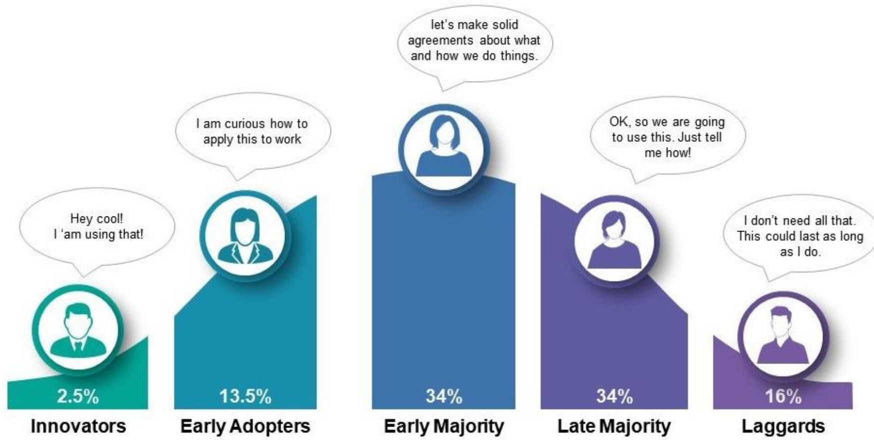


Figure 1.10: Adoption curve: the S-curve is the integral of adoption curve. Image from Commons Wikimedia, licensed under the Creative Commons Attribution-Share Alike 4.0 International.

The S-curve describes a slow initial adoption, followed by a rapid increase in adoption, and finally a slowing down as saturation is approached. The inflection point of the curve is where the adoption rate is fastest. In the context of technology adoption:

- $x(t)$ represents the number of adopters (or the market share of a particular technology) at time t,
- r is the rate of adoption and
- K is the saturation level, or the maximum number of adopters (e.g., the total market size or the total population that can adopt the technology).

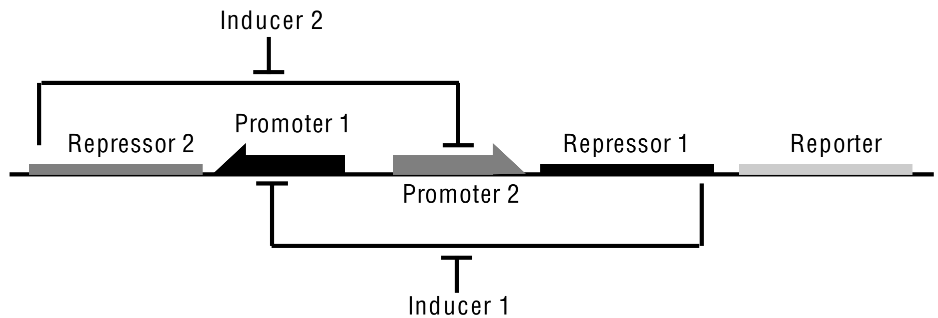
In the logistic equation, the growth rate slows as the number of adopters approaches the saturation level.

1.8 Appendix: A bistable toggle switch in synthetic biology

From Wikipedia: “Synthetic biology is a multidisciplinary field of science that focuses on living systems and organisms, and it applies *engineering principles* to develop new *biological parts, devices, and systems* or to redesign existing systems found in nature”

The field synthetic biology and systems biology studies *regulatory networks*, that is, groups of molecules (genes, proteins, etc) with varying concentration that interact and promote or repress each other. A *toggle switch* is a simple regulatory network that manifests *bistability*, that is, the ability to remain stably in one of two states and to be toggled between these two states using a stimulus.

(a) An electromechanical toggle switch



(b) A diagram illustrating a biological toggle switch is shown. While the specific chemicals and their interactions may not be immediately clear, a simplified mathematical model will be introduced below for clarity. This image is taken from the seminal work by (Gardner et al., 2000), which demonstrates the successful creation of a synthetic, bistable gene-regulatory network within *Escherichia coli* bacteria.

Figure 1.11: Synthetic biology holds the potential to design biological circuits in much the same way we design electrical circuits and electromechanical systems.

We present here a simple (possibly the simplest) dimensionless mathematical model of the toggle switch which exhibits bistability. Consider two genes with concentrations u and v, the *toggle switch dynamics* is

$$\begin{aligned} \dot{u} &= \frac{\alpha_1}{1+v^{n_1}} - \beta u, \\ \dot{v} &= \frac{\alpha_2}{1+u^{n_2}} - \beta v, \end{aligned} \tag{1.9}$$

where:

- the quantity $\dfrac{\alpha_1}{1+v^{n_1}}$ is the *production rate* of u, inhibited by v,
- the quantity $\dfrac{\alpha_2}{1+u^{n_2}}$ is the production rate of v, inhibited by u,
- the quantities n_1 and n_2 are *sensitivity coefficients* indicating the non-linearity of the repression,
- the quantities α_1 and α_2 are the *maximum production rates* of u and v, respectively, and
- the quantity β is the *degradation rate* common for both u and v (we choose it equal for simplicity).

Note: To describe the production rates, we used a *Hill function* $f(u) = \dfrac{\alpha}{1+u^n}$. This expression is derived from the theory of mass action kinetics in chemistry. The sensitivity parameter n plays a key role.

As we show in the next Figure 1.12, the system can exhibit bistability, meaning it can remain stable in a state where u is high and v is low, or vice versa. An external stimulus can be employed to switch between these states.

Numerical analysis of the toggle switch dynamics

```python
# Python libraries
import numpy as np; import matplotlib.pyplot as plt;
from scipy.integrate import odeint

# Toggle switch dynamics
def toggle_switch(Y, t, alpha1, alpha2, n1, n2, beta):
    u, v = Y
    dudt = alpha1 / (1 + v**n1) - beta * u
    dvdt = alpha2 / (1 + u**n2) - beta * v
    return [dudt, dvdt]

# Parameters
alpha1, alpha2 = 1.0, 1.0  # production rates
n1 = 3.53; n2 = 3.55       # Hill coefficients
beta = 0.5                 # degradation rate

# Initial conditions: [u0, v0] and Time array
initial_conditions = [ [0.35, 1.1],  [.9, 0.5],  [1, 2], [2.5, 1] ]
t = np.linspace(0, 10, 1000); plt.figure(figsize=(10, 4))

# Solve and plot for each initial condition
colors = ['b', 'g', 'r', 'k']
for idx, ic in enumerate(initial_conditions):
    Y = odeint(toggle_switch, ic, t, args=(alpha1, alpha2, n1, n2, beta))
    u, v = Y.T; plt.plot(t, u, color=colors[idx], label=f'u0={ic[0]}, ...
        v0={ic[1]} (u)')
    plt.plot(t, v, '--', color=colors[idx], label=f'u0={ic[0]}, ...
        v0={ic[1]} (v)')

# Annotate the plot and save to PDF (plt.xlabel('Time, t'))
plt.title('Toggle Switch: sample solutions (u(t) solid and v(t) dashed ...
      vs   t) and phase portrait')
plt.ylabel('Concentrations (u=solid, v=dashed)'); plt.legend();
plt.xlim(0, 10); plt.ylim(0, 2.25); plt.grid(True)
plt.savefig("toggle-switch.pdf", bbox_inches='tight')

# Phase portrait
u, v = np.meshgrid(np.linspace(0, 4, 20), np.linspace(0, 3, 20))
du, dv = toggle_switch([u, v], 0, alpha1, alpha2, n1, n2, beta)
magnitude = np.sqrt(du**2 + dv**2);
# Plot the phase portrait with trajectories and equilibrium points
plt.figure(figsize=(12,9));
plt.streamplot(u, v, du, dv, density=.75, ...
    color='#0085ff',linewidth=magnitude, broken_streamlines=False)
for idx, ic in enumerate(initial_conditions):
    Y = odeint(toggle_switch, ic, t, args=(alpha1, alpha2, n1, n2, beta))
    plt.plot(Y[:,0], Y[:,1], color=colors[idx], label=f'u0={ic[0]}, ...
        v0={ic[1]}')
eq_points = np.array([[1., 1.], [0.17, 2.], [2., 0.17]])
plt.scatter(eq_points[:, 0], eq_points[:, 1], color='black', s=75, zorder=5)
plt.xlabel('u'); plt.ylabel('v'); plt.xlim(0, 4); plt.ylim(0, 3)
plt.savefig("toggle-switch-phase.pdf", bbox_inches='tight')
```

Listing 1.3: Python script generating Figure 1.12. Available at `toggle-switch.py`

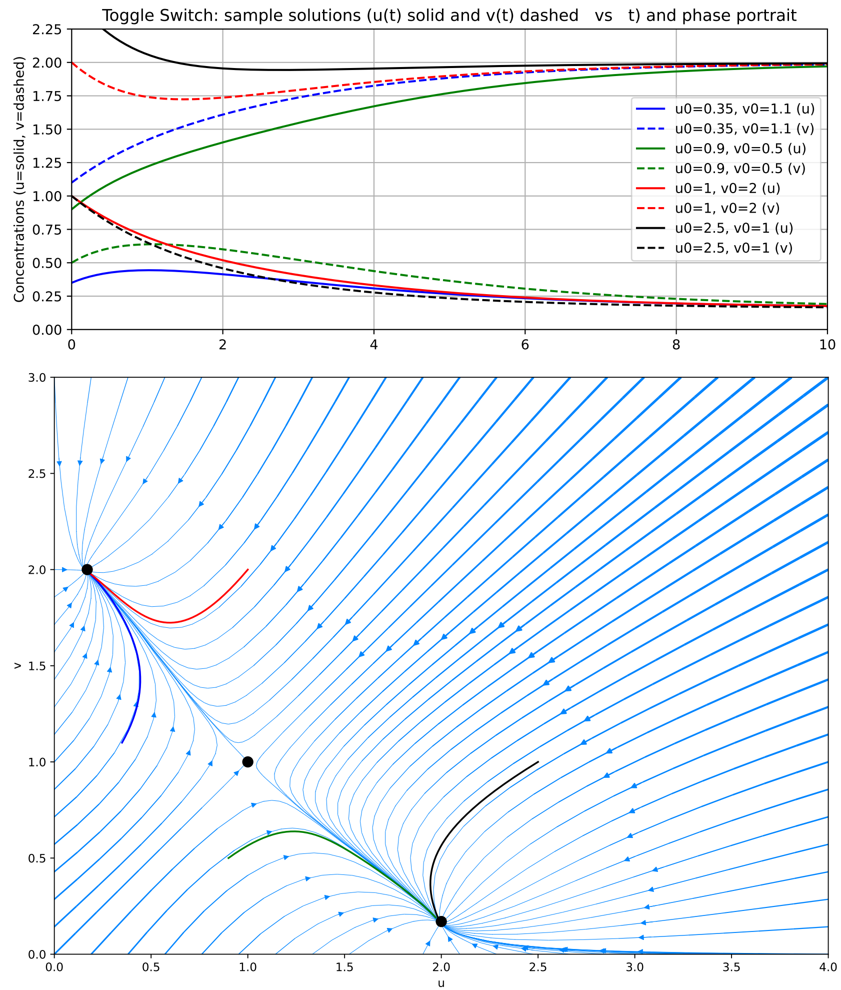


Figure 1.12: Solutions and phase portrait for the toggle switch dynamics (1.9). Four initial conditions are randomly selected. Two trajectories (green and black) converge to the "(u, v) = on/off" equilibrium, the other two (red and blue) to the "(u, v) = off/on" equilibrium.

1.9 Exercises

E1.1 **Closed-form solution of the logistic equation.** Verify that

(i) the exponential function $x(t) = x_0\,\mathrm{e}^{at}$ is the solution to the linear differential equation $\dot{x} = ax$ with initial condition $x(0) = x_0$;

(ii) the logistic function $x(t) = \dfrac{Kx_0\,\mathrm{e}^{rt}}{K + x_0(\mathrm{e}^{rt} - 1)}$ is the solution to the logistic differential equation

$$\dot{x} = rx\left(1 - \frac{x}{K}\right), \qquad x(0) = x_0.$$

Answer:

(i) The solution follows simply by taking the derivative of $x(t) = x_0 e^{at}$ with respect to t:

$$\dot{x} = ax_0 e^{at} = ax.$$

An alternative method is via integration:

$$\frac{dx}{dt} = rx \quad \Longrightarrow \quad \frac{dx}{x} = rdt \quad \Longrightarrow \quad \int_0^t \frac{dx}{x} = rt \quad \Longrightarrow \quad \ln(x(t)) = \ln(x(0)) + rt \quad \Longrightarrow \quad x(t) = x(0)\,\mathrm{e}^{rt}.$$

(ii) To solve the logistic equation, we separate the variables and then integrate. Starting with:

$$\frac{dx}{x(1 - x/K)} = rdt.$$

We integrate both sides, with the right side integrated with respect to t and the left side with respect to x. The integral will yield an equation of the form:

$$-\ln\left(\left|\frac{K}{x} - 1\right|\right) = C - rt.$$

Solving for x and evaluating at $x(0) = x_0$, we compute

$$C = \ln(x_0/(x_0 - K)).$$

Finally, the conclusion follows by substituting C into the solution for x yielding:

$$x(t) = \frac{K}{\frac{K - x_0}{x_0} e^{rt} + 1} = \frac{Kx_0 e^{rt}}{K + x_0\,(e^{rt} - 1)}.$$

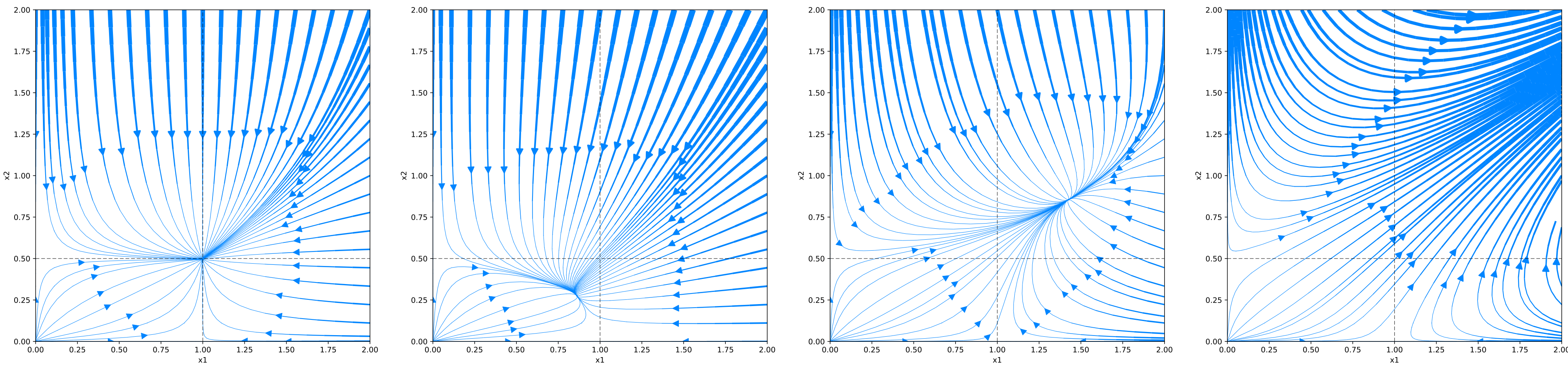
■

E1.2 **Reasonable and unreasonable predictions of a Lotka-Volterra model.** Given an interaction parameter α taking one of the values $\{0, -0.5, 0.5, 2\}$, consider the following Lotka-Volterra system for two species (note this is not the same predator-prey system in Section 1.3):

$$\begin{aligned}\dot{x}_1 &= x_1 - x_1^2 + \alpha x_1 x_2 \\ \dot{x}_2 &= x_2 - 2x_2^2 + \alpha x_1 x_2\end{aligned} \tag{E1.1}$$

Note: when $\alpha = 0$ the two species do not interact with each other. When $\alpha > 0$, each species has a beneficial effect on the other; this is called *mutualism*. When $\alpha < 0$, each species has a detrimental effect on the other; this is called *competition*.

***Note:** For clarity, x_1 and x_2 are normalized numbers, not absolute numbers. For example: the number of individuals of species #1 may be of $x_1 \cdot 10^6$.*



(a) No interaction: $\alpha = 0$ (b) Low competition: $\alpha = -0.5$ (c) Low mutualism: $\alpha = 0.5$ (d) High mutualism: $\alpha = 2$

(i) For the no-interaction case $\alpha = 0$, compute the carrying capacity for each of the two species in isolation and compute each equilibrium point of the two-species system.

(ii) For each case ($\alpha = 0$, $\alpha = -0.5$, $\alpha = 0.5$, $\alpha = 2$), how many equilibria exist?

(iii) For $\alpha \neq 0$, how does the interaction between species affect the location of the equilibria?

(iv) Is the behavior (location of equilibria and asymptotic value of solutions) predicted by the model (E1.1) ecologically reasonable for each value of α?

Answer: First, observe that

$$\begin{aligned}\dot{x}_1 &= x_1 - x_1^2 + \alpha x_1 x_2 = x_1(1 - x_1 + \alpha x_2) \\ \dot{x}_2 &= x_2 - 2x_2^2 + \alpha x_1 x_2 = x_2(1 - 2x_2 + \alpha x_1)\end{aligned}$$

(i) Setting the above equations for $\dot{x}_1$ and $\dot{x}_2$ equal to 0 with $\alpha = 0$, we can see that the fixed points are $x_1 = 0, 1$ and $x_2 = 0, 0.5$. Thus, the carrying capacities are 1 and 0.5, respectively.

(ii) Setting the above equations equal to zero, we can see that we have at least three equilibria in all four cases: $(0, 0)$, $(0, 1/2)$, $(1, 0)$.
Additionally, when $\alpha \neq \pm\sqrt{2}$, a fourth equilibrium also exists, which solves the system of equations

$$\begin{aligned} 0 &= 1 - x_1 + \alpha x_2, \\ 0 &= 1 - 2x_2 + \alpha x_1. \end{aligned}$$

A little linear algebra shows that this system has a unique solution at all four parameter values considered. The fourth equilibrium occurs at the point

$$\left(\frac{2+\alpha}{2-\alpha^2}, \frac{1+\alpha}{2-\alpha^2} \right).$$

(Note that in the case where $\alpha = \pm\sqrt{2}$, this system of equations has no solutions.) When $\alpha = 2$, the solution is $(-2, -3/2)$, which is not physically since the variables (x_1, x_2) are supposed to be positive or zero.

(iii) The first three equilibria are unaffected by the value of α. We can see that as α increases from 0, the fourth equilibrium moves up and to the right until it escapes to (∞, ∞) at $\alpha = \sqrt{2}$. As α decreases from 0, this point first moves down and to the left, and then moves down and to the right until it escapes to $(\infty, -\infty)$ at $\alpha = -\sqrt{2}$.

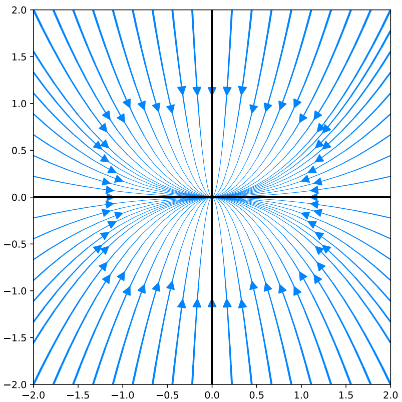
(iv) We at least expect both populations to be nonnegative and finite. We can see that the values for which this holds are $-1 < \alpha < \sqrt{2}$. Thus cases (a), (b), and (c) are arguably reasonable, while case (d) certainly is not.

■

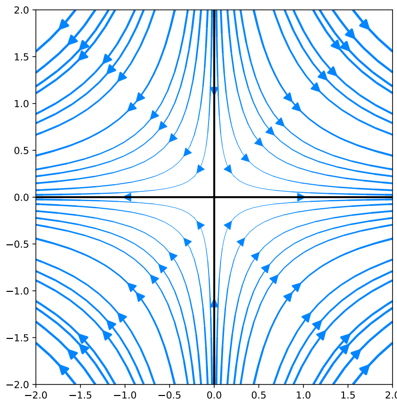
E1.3 **Matching phase portraits to dynamics.** Given a real number r taking six possible values in $\{-2, -1, -0.5, 0, 1, 5\}$, consider the linear dynamical system

$$\dot{x} = rx,$$
$$\dot{y} = -y.$$

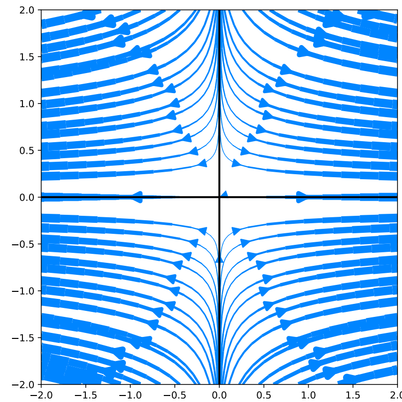
Match the six possible values of r to each of the six images below.



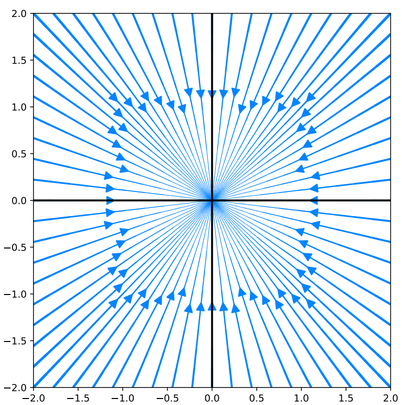
(a) $r =$ ☐



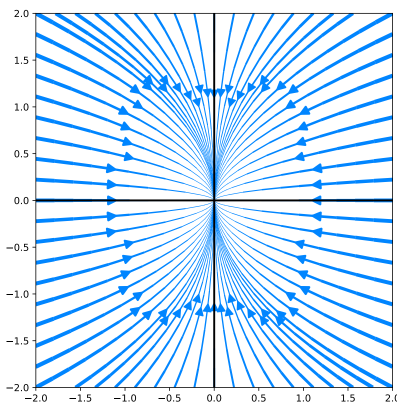
(b) $r =$ ☐



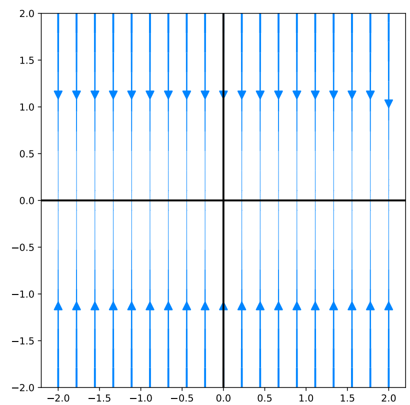
(c) $r =$ ☐



(d) $r =$ ☐



(e) $r =$ ☐



(f) $r =$ ☐

Answer: The answer is:

(a) $r = -0.5$, (b) $r = 1$, (c) $r = 5$, (d) $r = -1$, (e) $r = -2$, (f) $r = 0$

Here is the explanation: If $\dot{x} > 0$, the arrow points to the right and if $\dot{x} < 0$, the arrow points to the left. With the same concept, if $\dot{y} > 0$, the arrow points upward and if $\dot{y} < 0$, the arrow points downward (ref. Chapter 1, slide 23). In this case, the change rate $\dot{y}$ is not affected by r and we can see how the lines pointing towards the x-axis at $y = 0$, curve.

(a) When $r = -0.5, \dot{x} = -0.5x$ and $\dot{y} = -y$. There are two vectors, one points towards the x-axis at $y = 0$ and the other points towards the y-axis at $x = 0$. When the value of x is approaching 0, the change rate $\dot{x}$ is decreasing. Therefore, the lines pointing towards the x-axis at $y = 0$ is slightly curved because $\dot{x} < \dot{y}$.

(b) When $r = 1, \dot{x} = x$ and $\dot{y} = -y$. There are two vectors, one points towards the x-axis at $y = 0$ and the other points away from the y-axis at $x = 0$. Therefore, the lines curved and pointed out from the y-axis at $x = 0$. When the value of x is approaching 2 or -2, the change rate $\dot{x}$ is also increasing and the lines pointing to the x-axis at $y = 0$ are dramatically curved because $\dot{x} > \dot{y}$.

(c) When $r = 5, \dot{x} = 5x$ and $\dot{y} = -y$. There are two vectors, one points to the x-axis at $y = 0$ and the other points away from the y-axis at $x = 0$. Because the change rate $\dot{x}$ is much greater than $\dot{y}$ when x is increasing, it caused the lines to significantly point away from the y-axis at $x = 0$

(d) When $r = -1, \dot{x} = -x$ and $\dot{y} = -y$. There are two vectors, one points towards the x-axis at $y = 0$ and the other points towards y-axis at $x = 0$. The magnitude depends on the value of x and y. Adding two vectors together results in arrows pointing straight to $(0, 0)$ because $\dot{x} = \dot{y}$ everywhere.

(e) When $r = -2, \dot{x} = -2x$ and $\dot{y} = -y$. There are two vectors, one points towards the x-axis at $y = 0$ and the other points towards the y-axis at $x = 0$. When the value of x is approaching 0, the change rate $\dot{x}$ is decreasing. Therefore, the lines pointing to x-axis at $y = 0$ is slightly curved because $\dot{x} < \dot{y}$. However, with a larger x value, the change rate $\dot{x}$ is bigger and curved the lines in a greater magnitude because $\dot{x} > \dot{y}$.

(f) When $r = 0, \dot{x} = 0$ and $\dot{y} = -y$. We can see the straight lines point towards the x-axis at $y = 0$ when we apply $y = -2$ to 0 and $y = 2$ to 0.

■

E1.4 **Discrete-time dynamical systems.** This exercise is meant to illustrate that dynamical systems exist also over the discrete-time domain and not only over continuous time. Instead of the time variable t (taking values in the non-negative real numbers), we let k be the discrete time variable, that is, a non-negative integer. An interesting example of discrete-time dynamical systems arises in the domain of finance, in the context of investments and accumulation of bank deposits.

A single initial deposit: Given a scalar a and an initial condition $x(0)$, consider the discrete-time equation $x(k+1) = ax(k)$, for all nonnegative integers k. Show that the solution is $x(k) = x(0)a^k$.

Note: *Assume you deposit an amount $x(0)$ at a bank today. Let the coefficient a satisfy $a = 1 + i$, where i is the yearly interest offered by the bank. Then $x(k)$ is the value in your bank account k years from now.*

An initial deposit, plus yearly deposits: Consider the discrete-time equation $x(k+1) = ax(k) + b$, where b is a constant input or forcing term. Show that the solution from initial condition $x(0)$ is

$$x(k) = \begin{cases} x(0) + kb, & \text{if } a = 1, \\ a^k x(0) + \dfrac{1-a^k}{1-a} b & \text{if } a \neq 1. \end{cases} \tag{E1.2}$$

Hint: Recall the geometric series $\dfrac{1-a^k}{1-a} = 1 + a + a^2 + \cdots + a^{k-1}$.

Note: *This scenario assumes that, after depositing an amount $x(0)$ at year 0, you deposit the amount b each following year.*

Answer: We prove only equation (E1.2), since the case where $b = 0$ is an immediate consequence. The case when $a = 1$ is trivial. We need to show that $x(k) = a^k x(0) + \frac{1-a^k}{1-a} b$ for $a \neq 1$, when $x(k+1) = ax(k) + b$. (The case $a = 1$ is elementary.) We proceed by induction. First, at $k = 0$, we verify

$$x(0) = a^k x(0) + \frac{1-a^k}{1-a} b \Big|_{k=0} = x(0) + \frac{0}{1-a} b = x(0). \tag{E1.3}$$

Second, we assume the statement is true at k and we need to show that it holds at $k+1$. We compute

$$\begin{aligned} x(k+1) = ax(k) + b &= a\Big(a^k x(0) + \frac{1-a^k}{1-a} b\Big) + b \\ &= a^{k+1} x(0) + a\frac{1-a^k}{1-a} b + b \\ &= a^{k+1} x(0) + \Big(a\frac{1-a^k}{1-a} + 1\Big) b \\ &= a^{k+1} x(0) + \frac{a - a^{k+1} + 1 - a}{1-a} b = a^{k+1} x(0) + \frac{1-a^{k+1}}{1-a} b. \end{aligned}$$

This expression matches the form of the solution for $k+1$, so the inductive step holds. ■

E1.5 **Finding typos in Python code.** Each of the following three programs contains precisely one programming mistake. Identify the mistake and the line in which it happens.

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from scipy.integrate import odeint
4
5  # Logistic equation
6  def log_eq(x, t, r, K)
7      return r * x * (1 - x / K)
8
9  # Parameters
10 r, K = 0.1, 100
11 initial_conditions = [1, 10, 50, 100]
12 times = np.linspace(0, 125, 1000)
13
14 # Solve and plot
15 plt.figure()
16 for x0 in initial_conditions:
17     solution = odeint(log_eq, x0, ...
           times, args=(r, K))
18     plt.plot(times, solution)
19
20 plt.xlabel("Time"), plt.ylabel("Population")
21 plt.legend(), plt.show()
```

Listing 1.4:

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from scipy.integrate import odeint
4
5  # Logistic equation
6  def log_eq(x, t, r, K):
7      return r * x * (1 - x / K)
8
9  # Parameters
10 r, K = 0.1, 100
11 initial_conditions = [1, 10, 50, 100]
12 times = np.linspace(0, 125, 1000)
13
14 # Solve and plot
15 plt.figure()
16 for x0 in initial_conditions:
17     solution = odeint(log_eq, x0, ...
           times, args=(r, K))
18     plt.plot(times, soluton)
19
20 plt.xlabel("Time"), plt.ylabel("Population")
21 plt.legend(), plt.show()
```

Listing 1.5:

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from scipy.integrate import odeint
4
5  # Logistic equation
6  def log_eq(x, t, r, K):
7      return r * x * (1 - x / K)
8
9  # Parameters
10 r, K = 0.1, 100
11 initial_conditions = [1, 10, 50, 100]
12 times = np.linspace(0, 125, 1000)
13
14 # Solve and plot
15 plt.figure()
16 for x0 in initial_conditions:
17     solution = odeint(log_eq, times, ...
           args=(r, K))
18     plt.plot(times, solution)
19
20 plt.xlabel("Time"), plt.ylabel("Population")
21 plt.legend(), plt.show()
```

Listing 1.6:

E1.6 **A conserved quantity for the predator-prey system.** With the same notation as in Section 1.3, consider the Lotka-Volterra predator-prey model

$$\begin{aligned}\dot{x}_1 &= \alpha x_1 - \beta x_1 x_2,\\ \dot{x}_2 &= -\delta x_2 + \gamma x_1 x_2.\end{aligned} \tag{E1.4}$$

Show that the following quantity is conserved along the solution of the dynamical system:

$$\mathcal{H}(x_1, x_2) = \gamma x_1 - \delta \ln(x_1) + \beta x_2 - \alpha \ln(x_2). \tag{E1.5}$$

Hint: Note that the quantity $\mathcal{H}(x_1, x_2)$ along the solution of the dynamical system really is a function of time $\mathcal{H}(x_1(t), x_2(t))$, where $x_1(t)$ and $x_2(t)$ are the solutions the system.

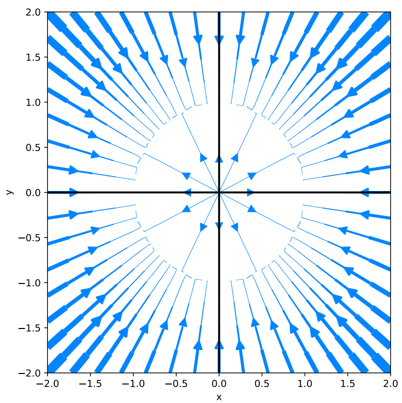
E1.7 **Matching phase portraits to dynamics.** Given a number ω taking values in $\{-1, 0, 1\}$, consider the nonlinear dynamical system

$$\dot{x} = \omega y + \big(1 - (x^2 + y^2)\big)x,$$
$$\dot{y} = -\omega x + \big(1 - (x^2 + y^2)\big)y.$$

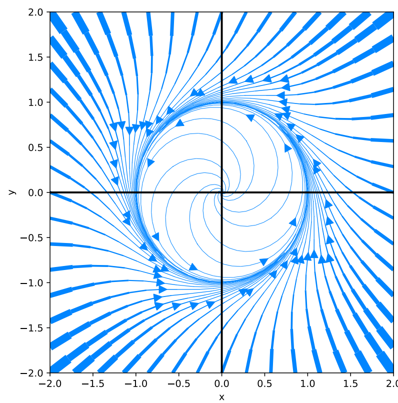
(i) Match the three possible values of ω to each of the three phase portraits below.

(ii) For each value of ω, calculate all the equilibrium points of the resulting system and their stability.

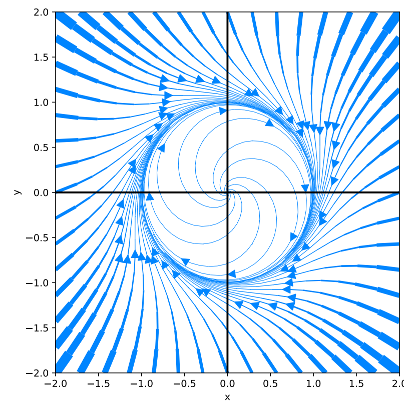
Hint: An equilibrium point x^* is *marginally stable* if nearby trajectories converge to x^* or remain near x^*.



(a) $\omega =$ ☐



(b) $\omega =$ ☐



(c) $\omega =$ ☐

E1.8 **Understanding phase portraits.** Here are some basic questions about phase portraits for 2-dimensional systems. Please provide a short 1-sentence response.

(i) why can there be only one solution curve passing through a given point in the phase portrait?

(ii) what do the arrows on the trajectories in a phase portrait represent?

(iii) why can solution curves never cross each other in a phase portrait?

(iv) what are equilibrium points, and how do you find them on a phase portrait?

(v) how can you tell if an equilibrium point is stable or unstable by looking at the phase portrait?

Note: *Please provide a short 1-sentence response.*

Bibliography

K. J. Åström and R. M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2 edition, 2021. URL http://www.cds.caltech.edu/~murray/books/AM08/pdf/fbs-public_24Jul2020.pdf.

T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, 403(6767):339–342, 2000. doi.

E. P. Odum. *Fundamentals of Ecology*. Saunders Company, 1959.

K. Ogata. *Dynamical Systems*. Pearson, 4 edition, 2003. ISBN 0131424629.

S. H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. Westview Press, 2 edition, 2015. ISBN 9780813350844.