Introduction to Signals and Systems
Lecture #2 - Mathematical Representations of Dynamical Systems

Guillaume Drion
Academic year 2017-2018
Outline

- Mathematical modeling: basics and illustrations
- System dynamics: static gain, input integration and state interactions (feedback)
- System output: update function vs output function
A simple modeling example: the wheat and chessboard problem

If you start by putting one grain on the first square and double the number for each consecutive square: (i) how many grains would be put on the nth square? (ii) how many grains do you need to fill the chessboard?
A simple modeling example: the wheat and chessboard problem

If you start by putting one grain on the first square and double the number for each consecutive square: (i) how many grains would be put on the nth square? (ii) how many grains do you need to fill the chessboard?

Mathematical model of the problem

\[
\begin{align*}
x[0] &= 1, \\
x[n + 1] &= 2x[n]
\end{align*}
\]

What is \(x[n]\)? What values can \(n\) take? And \(x[n]\)?
What does the first equation describes?
What does the second equation describes?
A simple modeling example: the wheat and chessboard problem

\[ x[0] = 1, \]
\[ x[n + 1] = 2x[n] \]

What is \( x[n] \)? What values can \( n \) take? And \( x[n] \)?

\( x[n] \) is the number of wheat grains on the \( n \)th square. \( n \) is the square number.

Such function is called **signal**.
A simple modeling example: the wheat and chessboard problem

\[
\begin{align*}
x[0] &= 1, \\
x[n + 1] &= 2x[n]
\end{align*}
\]

What is \( x[n] \)? What values can \( n \) take? And \( x[n] \)?

\( x[n] \) is the number of wheat grains on the \( n \)th square. \( n \) is the square number.

Such function is called **signal**.

\( n \) can take values in the range \([0, 63]\).

This defines the **domain** of the signal \( x[n] \).
A simple modeling example: the wheat and chessboard problem

\[
\begin{align*}
x[0] &= 1, \\
x[n + 1] &= 2x[n]
\end{align*}
\]

What is \( x[n] \)? What values can \( n \) take? And \( x[n] \)?

\- \( x[n] \) is the number of wheat grains on the \( n \)th square. \( n \) is the square number. Such function is called signal.

\- \( n \) can take values in the range \([0, 63]\).

This defines the domain of the signal \( x[n] \).

\- \( x[n] \) can be any positive integer.

This defines the image of the signal \( x[n] \).
Signals: **domain** and **image**

- **Systems treat signals.**

- **Signals** \((x)\) are defined by their **domain** \((X)\) and their **image** \((Y)\):

\[ x : X \rightarrow Y \]

- **Domain** (for us): \(X \subset \mathbb{Z}\) (discrete signal) or \(X \subset \mathbb{R}\) (continuous signal).

- **Image**: set of values that the signal can take (ex: probabilities/binary signals).
Inputs, outputs, variables, etc. are **signals**.

Examples:

“Hi!” on a computer is “01001000 01101001 00100001” (Ascii)

=> Discrete, binary signal $x[n]$. 
Inputs, outputs, variables, etc. are **signals**.

Examples:

“Hi!” on a computer is “01001000 01101001 00100001” (Ascii) => Discrete, binary signal $x[n]$.

Electrical signaling of a neuron:

$V_{Na}$  

$V_K$

=> Continuous signal $x(t)$, larger than $V_K$ and smaller than $V_{Na}$. 
Inputs, outputs, variables, etc. are **signals**.

Examples:

"Hi!" on a computer is "01001000 01101001 00100001" (Ascii)

=> **Discrete, binary signal** \( x[n] \).

Electrical signaling of a **neuron**:

\[ V_{Na} \rightarrow x(t) \rightarrow V_{Na} \]

\[ V_{K} \rightarrow x(t) \rightarrow V_{K} \]

=> **Continuous signal** \( x(t) \), larger than \( V_K \) and smaller than \( V_{Na} \).
Signals: **domain** and **image**

- A continuous signal can have a somehow “discrete image”...
  - **Ex:** current flowing through an ion channel over time:

  ![Image of ion channel](image)

- ... and a discrete signal can have a “continuous image”.
  - **Ex:** many digitalized signals.
Why should we care about the **domain** of a signal?

- Comparing continuous and discrete signals:
  - **Time-shift**
    \[
    x(t) \rightarrow x(t - t_0) \quad x[n] \rightarrow x[n - n_0]
    \]
  - **Time-reversal**
    \[
    x(t) \rightarrow x(-t) \quad x[n] \rightarrow x[-n]
    \]
  - **Time-scaling**
    \[
    x(t) \rightarrow x(\alpha t) \quad x[n] \rightarrow x[\alpha n]
    \]
Why should we care about the **domain** of a signal?

- Comparing continuous and discrete signals:
  - **Time-shift**
    \[ x(t) \rightarrow x(t - t_0) \quad x[n] \rightarrow x[n - n_0] \]
  - **Time-reversal**
    \[ x(t) \rightarrow x(-t) \quad x[n] \rightarrow x[-n] \]
  - **Time-scaling**
    \[ x(t) \rightarrow x(\alpha t) \quad x[n] \rightarrow x[\alpha n] \]
    Some values of \( \alpha \) lead to problems
Why should we care about the **image** of a signal?

- In theory, you can design any system using mathematical modeling.

- Ex: we want to design a system that controls the movement of a robot arm for weight lifting, and proudly come up with this system that works perfectly:

```
<table>
<thead>
<tr>
<th>Input voltage</th>
<th>Lifted weight (relative to arm weight)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.2</td>
</tr>
<tr>
<td>2</td>
<td>2.4</td>
</tr>
<tr>
<td>3</td>
<td>3.6</td>
</tr>
<tr>
<td>4</td>
<td>4.8</td>
</tr>
<tr>
<td>5</td>
<td>6.0</td>
</tr>
<tr>
<td>6</td>
<td>7.2</td>
</tr>
</tbody>
</table>
```

![Graph showing input voltage vs. lifted weight relative to arm weight](image-url)
Why should we care about the **image** of a signal?

- In theory, you can design any system using mathematical modeling.

- Ex: we want to design a system that controls the movement of a robot arm for weight lifting, and proudly come up with this system that works perfectly:

- But in real-life, the values that the signals can take (image) are **constrained**. You have to take this limitation into account in your model!

- Ex: in the robot, the voltage signal cannot exceed some saturation voltage. With our design, the robot can barely lift his arm, let alone any additional weight...
Back to the wheat and chessboard problem

- If you start by putting one grain on the first square and double the number for each consecutive square: (i) how many grains would be put on the nth square? (ii) how many grains do you need to fill the chessboard?

Mathematical model of the problem

\[ x[0] = 1, \]
\[ x[n + 1] = 2x[n] \]

What is \( x[n] \)? What values can \( n \) take? And \( x[n] \)? What does the first equation describes? What does the second equation describes?
A simple modeling example: the wheat and chessboard problem

\[
\begin{align*}
x[0] &= 1, \\
x[n + 1] &= 2x[n]
\end{align*}
\]

What does the first equation describes?

- \( x[0] = 1 \) is the initial condition of my system.

- A system/model can have a very different output for different initial conditions. Example:

\[
\begin{align*}
x[0] &= 1, & x[0] &= 0, \\
x[n + 1] &= 2x[n] & x[n + 1] &= 2x[n]
\end{align*}
\]
A simple modeling example: the wheat and chessboard problem

\[
\begin{align*}
x[0] &= 1, \\
x[n + 1] &= 2x[n]
\end{align*}
\]

What does the second equation describes?

- \( x[n + 1] = 2x[n] \) describes the “future” evolution of my signal \( x[n] \).

- This equation contains the **dynamics** of my system.

- The signal \( x[n] \) is a **variable** of my mathematical model.
A simple modeling example: the wheat and chessboard problem

If you start by putting one grain on the first square and double the number for each consecutive square: (i) how many grains would be put on the nth square?

\[
x[0] = 1, \\
x[n + 1] = 2x[n]
\]

- \(x[0] = 1 = 2^0\),
- \(x[1] = 2 \times 1 = 2 = 2^1\),
- \(x[0] = 2 \times 2 = 4 = 2^2\),
- ...
A simple modeling example: the wheat and chessboard problem

If you start by putting one grain on the first square and double the number for each consecutive square: (i) how many grains would be put on the nth square?

\[ x[0] = 1, \]
\[ x[n + 1] = 2x[n] \]

- \[ x[0] = 1 \]
- \[ x[1] = 2 \times 1 = 2 = 2^1, \]
- \[ x[2] = 2 \times 2 = 4 = 2^2, \]
- \[ \ldots \]

This function describes the state of my system at “time” \( n \).
In “state-space”, dynamical systems are modeled using **difference equations** (discrete domain) or **differential equations** (continuous domain).

<table>
<thead>
<tr>
<th>Discrete domain</th>
<th>Continuous domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ x[n + 1] = ax[n] ]</td>
<td>[ \dot{x} = ax \text{ (where } \dot{x} = \frac{dx}{dt} \text{)} ]</td>
</tr>
<tr>
<td>[ \rightarrow x[n] = a^n ]</td>
<td>[ \rightarrow x(t) = e^{at} ]</td>
</tr>
</tbody>
</table>

\( x[n], x(t) \) are **variables** of the systems (the model describes their evolution). A system can have many variables (dimension).

\( a \) is a **parameter** of the system. A system can have many parameters.
In “state-space”, dynamical systems are modeled using **difference equations** (discrete domain) or **differential equations** (continuous domain).

**Discrete domain**

\[ x[n + 1] = ax[n] \]

\[ \rightarrow x[n] = a^n \]

**Continuous domain**

\[ \dot{x} = ax \quad (\text{where } \dot{x} = \frac{dx}{dt}) \]

\[ \rightarrow x(t) = e^{at} \]

\( a \) is a **parameter** of the system. A system can have many parameters.

\( a < 0 \)

\( a > 0 \)
A simple modeling example: the wheat and chessboard problem

If you start by putting one grain on the first square and double the number for each consecutive square: (ii) how many grains do you need to fill the chessboard?

\[
x[0] = 1, \\
x[n + 1] = 2x[n]
\]

This function describes the output of my system at “time” \(n\).
Modeling example: predator-prey model

\[ H[n + 1] = \]
\[ L[n + 1] = \]
Modeling example: predator-prey model

\[ H[n + 1] = H[n] \]
\[ L[n + 1] = L[n] \]

1. If Hare or Lynx do not die or the food supply does not increase, the populations remain stable.
Modeling example: predator-prey model

\[ H[n + 1] = H[n] + bH[n] \]
\[ L[n + 1] = L[n] + cL[n]H[n] \]

2. If Hares get more food (parameter \( b \)), their population increases. If Lynxes get more Hares to eat, their population increases with a rate \( c \).
Modeling example: predator-prey model

\[
\]
\[
\]

3. If Hares are eaten by lynxes, their population decreases with a rate \( a \). If Lynxes die, their population decreases with a rate \( d \).
Modeling example: predator-prey model

\[
\]

Observations/data

Model simulation
Modeling example: neuron firing

Equivalent scheme
Modeling example: neuron firing

Mathematical model

\[ C_m \dot{V}_m = - \bar{g}_{Na} m_N^3 h_N (V_m - V_{Na}) - \bar{g}_K m_K^4 (V_m - V_K) - g_{leak} (V_m - V_{leak}) + I_{app}, \]

with

\[ \tau_{m_N}(V_m) \dot{m}_N = -(m_N - m_{N,\infty}(V_m)) \]
\[ \tau_{h_N}(V_m) \dot{h}_N = -(h_N - h_{N,\infty}(V_m)) \]
\[ \tau_{m_K}(V_m) \dot{m}_K = -(m_K - m_{K,\infty}(V_m)), \]
Modeling example: neuron firing

Experimental data

Model simulation
Examples of problems requiring mathematical modeling

Design of mass-damper

Design of a robot arm

Design of a DC/DC converter
Outline

- Mathematical modeling: basics and illustrations
- System dynamics: static gain, input integration and state interactions (feedback)
- System output: update function vs output function
System dynamics

- Modeling and analysis of systems: open loop.
  “Observing and analyzing the environment”
System dynamics: **static gain**

- Example #1: Applying a force on a spring.

\[ F = kx \quad \iff \quad x = \frac{1}{k} F \]

![Diagram showing a spring with a force input and displacement output](image)

- **F (Input)**
  - 0
  - 1

- **x (Output)**
  - 0
  - \( \frac{1}{k} \)
System dynamics: **static gain**

Example #1: Applying a force on a spring.

\[ F = kx \iff x = \frac{1}{k}F \]

The output is solely shaped by the input at the present time: **static system**.
System dynamics: static gain

Example #2: A current through a resistive wire.

\[ V = Ri \iff i = \frac{1}{R} V \]
System dynamics: static gain

□ Mathematical representation: functions.

1 input, 1 output: \( y(t) = Ku(t) \) where \( K \) is the static gain.

□ More than one input and/or output: matrix representation:

\[
y(t) = k_1 u_1(t) + k_2 u_2(t) + \cdots + k_n u_n(t) \]

\[
= \begin{pmatrix} k_1 & k_2 & \cdots & k_n \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \cdots \\ u_n \end{pmatrix} = Ku(t)
\]
Example #1bis: Applying a force on a damper.

\[ F = cv = c\dot{x} \quad \text{where} \quad \dot{x} = \frac{dx}{dt} \]

\[ \implies \dot{x} = \frac{1}{c}F \]
Example #1bis: Applying a force on a damper.

\[ F = cv = c \dot{x} \quad \text{where} \quad \dot{x} = \frac{dx}{dt} \]

\[ \Rightarrow \dot{x} = \frac{1}{c} F \]

The output depends on past values of input: \textit{energy storage (memory)}.

The \textit{energy} from input F is \textit{stored} in position x. x is a \textit{state} of the system.
Example #2bis: A current “through” a capacitor.

\[ i = CV \iff \dot{V} = \frac{1}{C}i \]

The energy from input i is stored in voltage V. V is a state of the system.
System dynamics: **input integration**

- **Integrators** store past values of the input into **states**.

\[
\dot{x} = Bu \quad \text{or} \quad x[n + 1] = Bu[n]
\]

- Mathematical representation: **ordinary differential equations** (continuous). 
  or **difference equations** (discrete).
System dynamics: **state interactions**

Example #1ter: Applying a force on a spring and a damper in parallel.

\[ F = kx + cx \]

Behavior?
System dynamics: state interactions

Example #1ter: Applying a force on a spring and a damper in parallel.

\[ F = kx + cx \]

\[ \Rightarrow \dot{x} = \frac{1}{c} [F - kx] \]

(a) \[ \dot{x} = \frac{1}{c} F \equiv \frac{1}{c} u : \text{integration of the input} \]
System dynamics: **state interactions**

Example #1ter: Applying a force on a spring and a damper in parallel.

\[ F = kx + c\dot{x} \]

\[ \Longrightarrow \dot{x} = \frac{1}{c} \left[ F - kx \right] \]

(a) \[ \dot{x} = \frac{1}{c} F \equiv \frac{1}{c} u \] : integration of the input

(b) \[ \dot{x} = -\frac{k}{c} x \] : does not depend on input - **Internal dynamics**!
System dynamics: **state interactions**

Example #1ter: Applying a force on a spring and a damper in parallel.

\[ F = kx + c\dot{x} \]

\[ \implies \dot{x} = \frac{1}{c} [F - kx] \]

(b) \( \dot{x} = -\frac{k}{c} x \): **Internal dynamics.**

An increase in the position \( x \) induces a decrease in the speed \( \dot{x} \). Adding the spring adds **negative feedback** to the system.
Example #1: Applying a force on a spring and a damper in parallel.

\[ F = kx + c\dot{x} \]

\[ \implies \dot{x} = \frac{1}{c} [F - kx] \]
Example #2ter: Applying an input voltage on an RC circuit.

\[ i = i_R + i_C = \frac{V}{R} + CV \]

\[ \implies \dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right] \]

(a) \( \dot{V} = \frac{1}{C} i \): integration of the input

(b) \( \dot{V} = -\frac{1}{RC} V \): Internal dynamics.
Example #2ter: Applying an input voltage on an RC circuit.

\[ i = i_R + i_C = \frac{V}{R} + CV \dot{V} \]

\[ \Rightarrow \dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right] \]
System dynamics: state interactions

- Mechanical system vs electrical system.

**Mechanical system**

\[ \dot{x} = \frac{1}{c} [F - kx] \]

**Electrical system**

\[ \dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right] \]
System dynamics: state interactions

- Mechanical system vs electrical system.

\[ \ddot{x} = \frac{1}{c} [F - kx] \quad \implies \quad \dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right] \]

\begin{align*}
\text{Mechanical system} & : F & \rightarrow & \frac{1}{c} & \rightarrow & x & \rightarrow & y \\
 & & \uparrow & \text{-}k/c & \downarrow & \text{-}k/c & \rightarrow & \frac{1}{c} \\
\text{Electrical system} & : i & \rightarrow & \frac{1}{C} & \rightarrow & V & \rightarrow & y \\
 & & \uparrow & -1/RC & \downarrow & -1/RC & \rightarrow & -1/RC
\end{align*}

Block diagram representation
System dynamics: state interactions

Static gain of the mechanical and electrical systems?

\[ \dot{x} = \frac{1}{c} \left[ F - kx \right] \]

\[ \dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right] \]
System dynamics: state interactions

Static gain of the mechanical and electrical systems?

**Mechanical system**
\[
\dot{x} = \frac{1}{c} \left[ F - kx \right]
\]

Steady-state: \( \dot{x} = 0 \)
\[
\frac{1}{c} \left[ F - kx \right] = 0
\]
\[
\implies x = \frac{F}{k}
\]

**Electrical system**
\[
\dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right]
\]
System dynamics: state interactions

- Static gain of the mechanical and electrical systems?

**Mechanical system**

\[
\implies \dot{x} = \frac{1}{c} [F - kx]
\]

Steady-state: \( \dot{x} = 0 \)

\[
\frac{1}{c} [F - kx] = 0
\]

\[
\implies x = \frac{F}{k}
\]

**Electrical system**

\[
\implies \dot{V} = \frac{1}{C} \left[ i - \frac{V}{R} \right]
\]

Steady-state: \( \dot{V} = 0 \)

\[
\frac{1}{C} \left[ i - \frac{V}{R} \right] = 0
\]

\[
\implies V = Ri
\]

- The damper (resp. capacitor) only plays a **dynamical role** (no static effect).
System dynamics: mathematical representation

- The dynamics of a system are represented by **ordinary differential equations** (continuous) or **difference equation** (discrete) of the form

\[ \dot{x}(t) = f(x(t), u(t)) \quad x[n + 1] = f(x[n], u[n]) \]

- For **linear systems**, it writes

\[ \dot{x} = Ax + Bu \quad x[n + 1] = Ax[n] + Bu[n] \]

where \( x \) and \( u \) are vectors, and \( A \) and \( B \) are matrices (dimensionality of the system).
System dynamics: state interactions

☐ Classical example: Newton’s second law of motion.

\[ F = m\ddot{x} \]

State #1: position \((x_1 = x)\)
State #2: speed \((x_2 = \dot{x})\)

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix}
0 & 1 \\
0 & 0
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} + \begin{bmatrix}
0 \\
1/m
\end{bmatrix} F
\]
System dynamics: state interactions

☐ Classical example: Newton’s second law of motion.

\[ F = m\ddot{x} \]

State #1: position \((x_1 = x)\)
State #2: speed \((x_2 = \dot{x})\)

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
1/m
\end{bmatrix} F
\]

Speed affects position  
(state interactions)

Force affects speed  
(Input integration)
Outline

- Mathematical modeling: basics and illustrations
- System dynamics: static gain, input integration and state interactions (feedback)
- System output: update function vs output function
**States** are not always outputs of the system

Example: a series RC circuit in a black box

![Diagram of a series RC circuit](image)

\[ \dot{V}_c = \frac{1}{RC} (V_{in} - V_c) \]

but we cannot measure \( V_c \)!

\[ \dot{V}_c = \frac{1}{RC} (V_{in} - V_c) \]

is the update function: it describes systems dynamics
States are not always outputs of the system

Example: a series RC circuit in a black box

\[ \dot{V}_c = \frac{1}{RC}(V_{in} - V_c) \]

but we cannot measure \( V_c \)!

Systems output? In terms of input and state: \( y = V_R = V_{in} - V_c \)

\[ y = V_{in} - V_c \text{ is the output function.} \]
States are not always outputs of the system

Example: a series RC circuit in a black box

\[ V_{out} = V_{in} - V_c \]

\[ \dot{V}_c = \frac{1}{RC} (V_{in} - V_c) \]

but we cannot measure \( V_c \)!

update function

output function
General form of dynamical systems: state-space representation

A dynamical system is represented by its update function and its output function:

\[
\dot{x} = f(x, u) \quad x[n + 1] = f(x[n], u[n])
\]
\[
y = g(x, u) \quad y[n] = g(x[n], u[n])
\]
General form of dynamical systems: state-space representation

- A dynamical system is represented by its update function and its output function

\[ \dot{x} = f(x, u) \quad \quad x[n + 1] = f(x[n], u[n]) \]
\[ y = g(x, u) \quad \quad y[n] = g(x[n], u[n]) \]

- For **linear systems**, it writes

\[ \dot{x} = Ax + Bu \quad \quad x[n + 1] = Ax[n] + Bu[n] \]
\[ y = Cx + Du \quad \quad y[n] = Cx[n] + Du[n] \]

where \( x, u \) and \( y \) are vectors,

\( A, B, C \) and \( D \) are matrices (dimensionality of the system).
Linear systems: $A,B,C,D$ representation

- Linear, time-invariant (LTI) dynamical systems can be represented in the form

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx + Du
\end{align*}
\]

where $A$ describes how the dynamics of the system evolve (dynamics matrix)

$B$ describes how the input influences the states (input matrix)

$C$ describes how the states “are seen” in the output (output matrix).

$D$ describes how the input directly influences the output (feedthrough matrix).

- The output is therefore a function of the input and the states.
  Simple but recurrent case: the output is one of the states.
Linear systems: block diagram representation
In this course, we will develop tools to analyze the behavior of dynamical systems.

In particular: **stability**, **static behavior**, **dynamical behavior**.
Why do we need general tools to analyze system dynamics? Illustration.

- In 1838, Pierre-François Verhulst proposed a dynamical model for the growth of a population (\( N \)) depending on the intrinsic growth rate (\( r \)) and the maximum number of individuals the environment can support (\( K \)).

\[
\frac{dN}{dt} = rN(1 - \frac{N}{K})
\]

- This equation is called the **logistic equation**.

- Simple behavior: If \( r >> \) and \( N << K \): the population grows fast.
  
  - If \( N = K \): the population does not grow anymore.

- Mathematical modeling is widely used in ecology.
The logistic equation

- Simulation of the logistic equation for different growth rates and $K=1$. 

![Logistic equation, $r=2$](image1)

![Logistic equation, $r=4$](image2)
A discrete equivalent of the logistic equation: the logistic map

In 1976, Robert May proposed a “discrete equivalent”

\[ x[n + 1] = \alpha x[n](1 - x[n]) \quad \text{vs} \quad \frac{dN}{dt} = rN(1 - \frac{N}{K}) \]
Dynamical behavior of the logistic map

Logistic map, $a=2$

Logistic map, $a=3.3$

Logistic map, $a=4$
Dynamical behavior of the logistic map

Logistic map, $a=2$

Logistic map, $a=3.3$
Dynamical behavior of the logistic map: chaos.

Logistic map, $a=2$

Logistic map, $a=3.3$

Logistic map, $a=4$
Dynamical behavior of the logistic map: chaos.

□ In 1976, Robert May proposed a “discrete equivalent”

\[ x[n + 1] = \alpha x[n] (1 - x[n]) \quad \text{vs} \quad \frac{dN}{dt} = rN\left(1 - \frac{N}{K}\right) \]

□ As opposed to the continuous system, the dynamics of the discrete system are extremely rich, and can be “chaotic” for certain values of \( \alpha \).

□ This dynamical richness comes from the nonlinearity of the system. But it highlights the fact that continuous and discrete systems are not always “equivalent”.
How can general tools help us analyze the behavior of dynamical systems?

- Illustration: systems **stability**.

Back to the spring-damper system $\Rightarrow \dot{x} = \frac{1}{c} [F - kx]$. 

How can general tools help us analyze the behavior of dynamical systems?

- Illustration: systems **stability**.

  Back to the spring-damper system \( \dot{x} = \frac{1}{c} [F - kx] \).

- If \( k > 0 \) (classical spring):
  the spring acts against the movement, giving a stable steady-state to the position. -> **negative feedback**
How can general tools help us analyze the behavior of dynamical systems?

- Illustration: systems \textit{stability}.

Back to the spring-damper system \[ \dot{x} = \frac{1}{c} \left[ F - kx \right]. \]

- If \( k > 0 \) (classical spring):
  the spring acts against the movement, giving a stable steady-state to the position.
  \[ \rightarrow \text{negative feedback} \]

- If \( k < 0 \):
  the further the system is, the faster is goes away.
  the system will go away until the “negative spring” saturates.
  \[ \rightarrow \text{positive feedback} \]
How can general tools help us analyze the behavior of dynamical systems?

- In a linear system: no saturation  
  \[ \text{positive feedback brings instability!} \]

- How to assess stability of a general linear system (1D): \( \dot{x} = ax + bu \)  
  System is stable if \( a < 0 \).
How can general tools help us analyze the behavior of dynamical systems?

- In a linear system: no saturation
  → Any positive feedback brings instability!

- How to assess stability of a general linear system (1D): \( \dot{x} = ax + bu \)
  System is stable if \( a < 0 \).

- How to assess stability of a general linear system (N-D): \( \dot{x} = Ax + Bu \)
  Multiple state-interactions: what are the feedbacks?
  N-feedback directions (eigenvectors of matrix \( A \)), each having its own amplitude (eigenvalues of \( A \)).
  → The system is stable if \( R(eig(A)) < 0 \) (all feedbacks negative).