ESS101
Modelling and Simulation, 2025

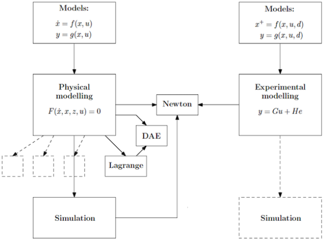
Lecturer and examiner: Yasemin Bekiroğlu
Course assistant: Ahmet Tekden

Systems & Control Division
Department of Electrical Engineering
Chalmers University of Technology

September, 2025

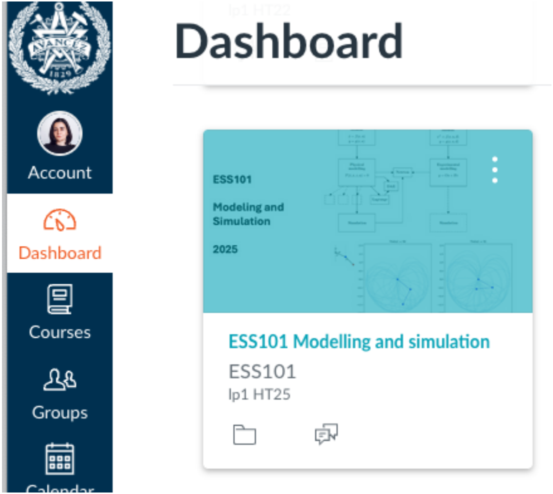
Lecture 1 – Introduction

- Introduction, practical information
- Overview and content of the course
- Prerequisites and learning outcomes
- Why models?
- Example
- State-space models, introduction



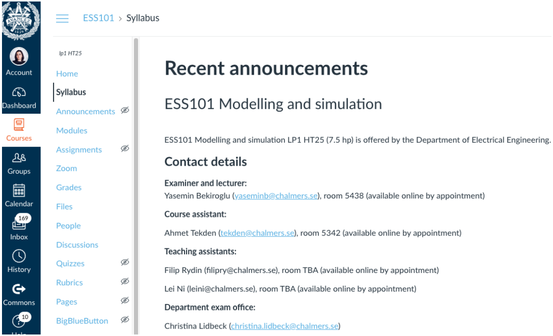
Practical information

- Canvas: syllabus, news, assignments, files, and more...



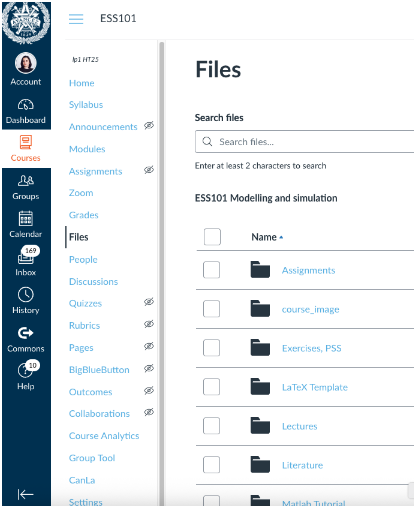
Practical information

- Canvas: syllabus, news, assignments, files, and more...



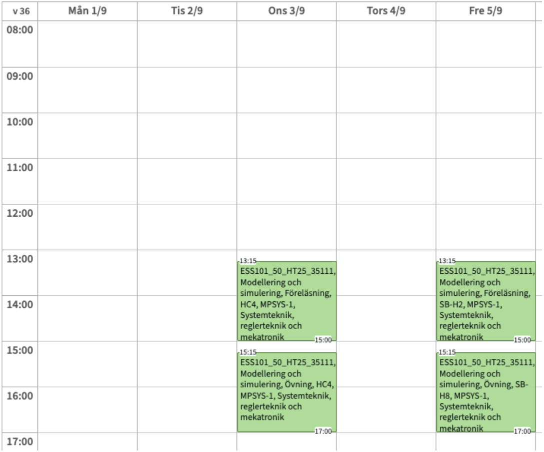
Practical information

- Canvas: syllabus, news, assignments, files, and more...



Practical information

- Canvas: syllabus, news, assignments and more...
  - For the schedule, check TimeEdit



v 36	Mån 1/9	Tis 2/9	Ons 3/9	Tors 4/9	Fre 5/9
08:00					
09:00					
10:00					
11:00					
12:00					
13:00			13:15 ESS101_50_HT25_35111, Modellering och simulering, Föreläsning, HC4, MPSYS-1, Systemteknik, reglerteknik och mekatronik 15:00		13:15 ESS101_50_HT25_35111, Modellering och simulering, Föreläsning, SB-H2, MPSYS-1, Systemteknik, reglerteknik och mekatronik 15:00
14:00					
15:00			15:15 ESS101_50_HT25_35111, Modellering och simulering, Övning, HC4, MPSYS-1, Systemteknik, reglerteknik och mekatronik 17:00		15:15 ESS101_50_HT25_35111, Modellering och simulering, Övning, SB-H8, MPSYS-1, Systemteknik, reglerteknik och mekatronik 17:00
16:00					
17:00					

Practical information

- Canvas: syllabus, news, assignments and more...
  - For the schedule, check TimeEdit
- Lectures: slides, notes available @Canvas
- Literature
- Problem solving sessions (PSS): analytical problems
- Assignments and supervision
- Examination
- Communication and interaction

Literature

- S. Gros, B. Egardt: *Modeling and Simulation, Lecture Notes 2024*. Sold at The Store and can be downloaded from Canvas.
- Modelling and Simulation, Exercises. Available @Canvas.
- Supplementary literature suggested at the course homepage.
- Also @Canvas: slides, Matlab files, solutions, old exams, etc.

Assignments

There are three mandatory assignments (Pass/Fail), pursued in groups of two students:

1. Lagrange modelling
2. System identification
3. Explicit and implicit integrators

Important:

- Form groups in Canvas by tomorrow, Sept 5, 4 pm.
- Read the assignment instructions carefully
- Start working early; some of you will find it demanding
- Attend the Matlab tutorial session
- Use Supervision sessions to get help (also Problem solving sessions (PSS))

Examination

To pass the course, you are required to pass the final exam and to have all assignments approved.

Assignment release dates and deadline:

- A1 release: Sept 5, after the lecture
- A1 deadline: Sept 16, midnight
- A2 release: Sept 17, after the lecture
- A2 deadline: Oct 1, noon
- A3 release: Oct 1, after the lecture
- A3 deadline: Oct 22, midnight
- A3 oral exams: Oct 23-24 (time and place to be announced)

Examination

To pass the course, you are required to pass the final exam and to have all assignments approved.

Exam dates during 2025/2026 are:

- 29 October 2025, 8:30, Johanneberg
- 08 Jan 2026 (Re-exam), 14:00, Johanneberg
- 24 Aug 2026 (Re-exam), 8:30, Johanneberg

Do not miss signup deadlines!

Material allowed at the exam:

- Modelling And Simulation, Lecture notes by S. Gros **(no annotations, cannot contain solutions to the exercises or previous exams)**
- Mathematics Handbook, Physics Handbook, and a Chalmers approved calculator. In addition, a Formula sheet will be appended with the exam.

Communication and interaction

We encourage you to use all means for interaction during the course, e.g.:

- Use Discussions @Canvas
- Ask questions during lectures, PSS, and supervision sessions
- Give feedback and suggestions to us and/or to student representatives
- Contact TAs and examiner for individual questions
  - Ahmet Tekden (Course assistant)
  - Filip Rydin
  - Lei Ni

Student representatives

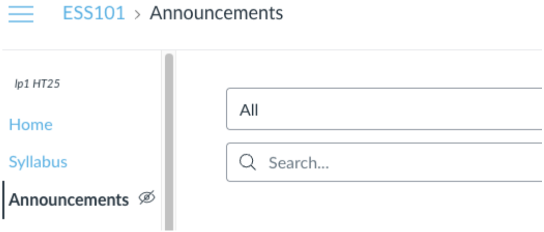
The following student representatives will participate in the course evaluation:

- Linus Fäldt, linusfa@chalmers.se
- Sareena Gouse Vaddinakatti, sareena@chalmers.se
- Massimiliano Migliorini, masmig@chalmers.se
- Xiaojie Ning, xiaojie@chalmers.se

Info about being a representative: `https://www.chalmers.se/en/education/your-studies/plan-and-conduct-your-studies/course-evaluation/`

Announcements

Make sure you follow the announcements.



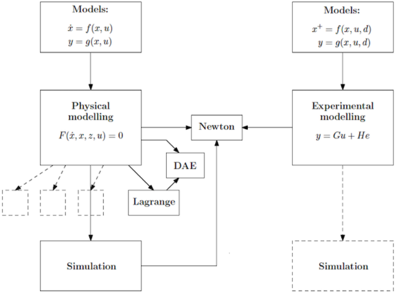
Course aim

The aim of the course is to provide tools for understanding and **constructing different forms of mathematical models** of industrial systems, and to understand **how they are numerically simulated** in the computer.

Combine mathematical principles and analysis with "learn-by-doing" via Matlab coding.

Course aim

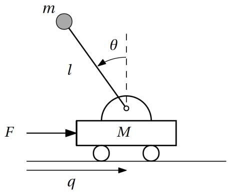
The course includes **modelling methods based on basic physical principles or on measured data from sensors**. Numerical simulation methods are studied with particular emphasis on accuracy and stability.

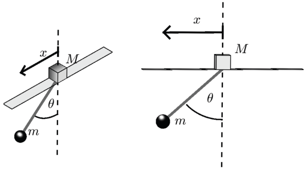


Prerequisites and learning outcomes

After completion of the course, you should be able to:

- Use methods and tools to **develop mathematical models** of dynamical systems by using basic physical laws. The emphasis will be on complex **mechanical systems**.





Learning outcomes

After completion of the course, you should be able to:

- Use methods and tools to **develop mathematical models of dynamical systems by using basic physical laws**.
- Analyze advanced forms of **differential equations** used in modelling.

Learning outcomes

After completion of the course, you should be able to:

- Use methods and tools to develop mathematical models of dynamical systems by using basic physical laws.
- Analyze advanced forms of differential equations used in modelling.
- Use methods and tools to **develop mathematical models of dynamical systems from measurement data**.

Learning outcomes

After completion of the course, you should be able to:

- Use methods and tools to develop mathematical models of dynamical systems by using basic physical laws.
- Analyze advanced forms of differential equations used in modelling.
- Use methods and tools to develop mathematical models of dynamical systems from measurement data.
- Understand basic **principles behind estimating parameters using data**.

Learning outcomes

After completion of the course, you should be able to:

- Use methods and tools to develop mathematical models of dynamical systems by using basic physical laws.
- Analyze advanced forms of differential equations used in modelling.
- Understand basic principles behind estimating parameters using data.
- Use methods and tools to develop mathematical models of dynamical systems from measurement data.
- Understand and **implement some of the numerical methods** used in simulations.

Course outline

The course is structured into four modules:

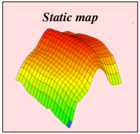
1. Intro - Models for dynamic systems
   - System dynamics and differential equations
   - Refresher of concepts from the basic control course
2. M1 - Building models from physics
   - Physical modelling
   - Equation based modelling
   - Lagrange modelling (assignment 1)
3. M2 - Building models from data
   - Parameter estimation
   - System identification (assignment 2)
4. M3 - Computational tools for modelling
   - Newton method
   - Differential-algebraic equations (DAEs)
5. M4 - Simulating dynamic models
   - Explicit and implicit integration methods (assignment 3)

What is a model?

A *model* is a simplified representation (approximation) of a system (“reality”).

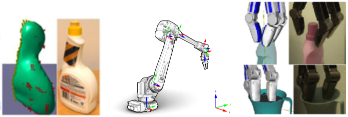
There are many types of models, e.g.

- Drawing, schematic, blue-print
- Functional description
- Block diagram, flow chart
- Graph
- Mathematical model



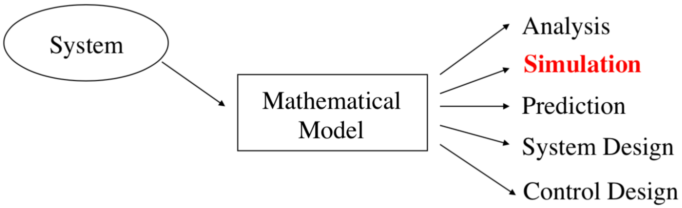
Mathematical models

$$\frac{dx}{dt} = f(x,u)$$
$$y = g(x,u)$$

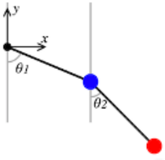


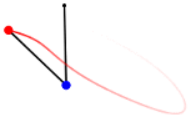
Our focus:
Mathematical models for *dynamical systems*, continuous and discrete time

What is the model used for?



Our focus: Simulation
Models and simulation is used "everywhere" in order to reduce costs, decrease development time, avoid dangerous experiments, and – not the least – because there is no alternative!



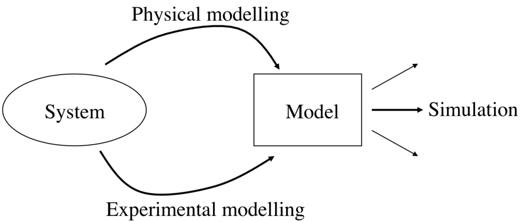


How to build a model?

There are two main approaches to build a model:

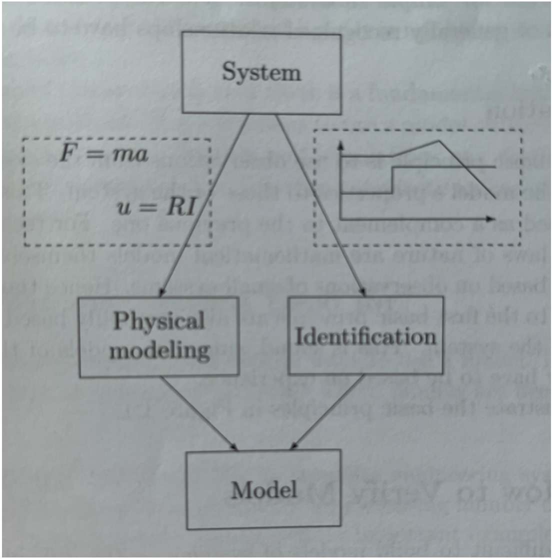
1. **Physical modelling:**
   Use the laws of physics (Newton, Kirchhoff, ...)
2. **Experimental (data-driven) modelling, system identification:**
   Perform experiments on the system, analyze data to deduce a model

Each of these will be pursued in the course (Modules I and II)!



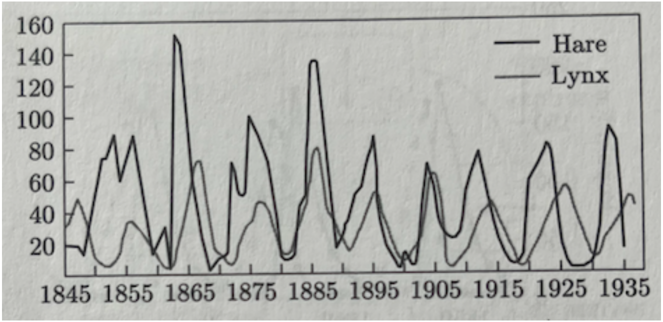
NB: In practice, often a combination of the two techniques is used.

How to build a model?



Example - Predators and prey: Explain the periodic variations in populations

Variations in populations of **predator-lynx and prey-hares**:

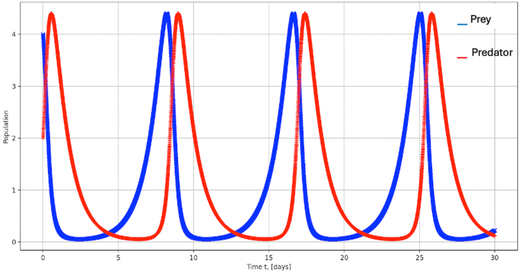


Example: Predators and prey

Describe the variation in populations of two species which interact via predation. This is a classical model to represent the dynamic of two populations.
Let $\alpha > 0, \beta > 0, \delta > 0$, and $\gamma > 0$. The system is given by

$$\begin{cases} \dot{x}_1 = x_1(\alpha - \beta x_2) \\ \dot{x}_2 = x_2(-\delta + \gamma x_1) \end{cases} \tag{1}$$

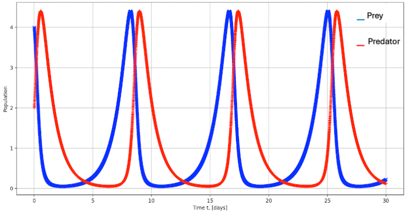
where x_1 represents prey (hare) population and x_2 predators (lynx) population.



Example: Predators and prey

$$\begin{cases} \dot{x}_1 = x_1(\alpha - \beta x_2) \\ \dot{x}_2 = x_2(-\delta + \gamma x_1) \end{cases} \tag{2}$$

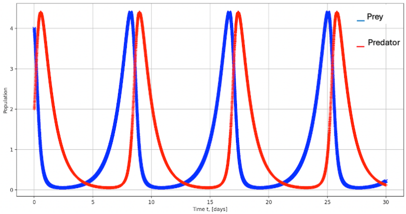
- We have **2 state variables** (called x_1 for prey population and x_2 for predator population).



Example: Predators and prey

$$\begin{cases} \dot{x}_1 = x_1(\alpha - \beta x_2) \\ \dot{x}_2 = x_2(-\delta + \gamma x_1) \end{cases} \tag{3}$$

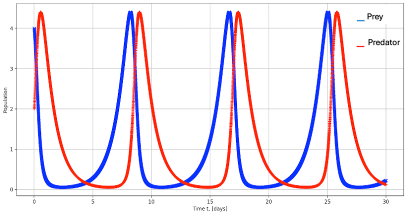
- We have 2 state variables (called x_1 for prey population and x_2 for predator population).
- This is **an autonomous system, it has no inputs**.



Example: Predators and prey

$$\begin{cases} \dot{x}_1 = x_1(\alpha - \beta x_2) \\ \dot{x}_2 = x_2(-\delta + \gamma x_1) \end{cases} \tag{4}$$

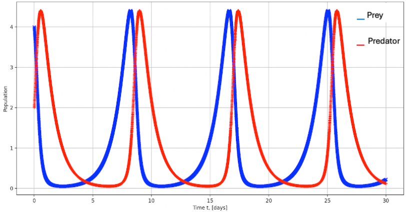
- We have 2 state variables (called x_1 for prey population and x_2 for predator population).
- This is an autonomous system, it has no inputs.
- The above **model captures the main dynamics in this system** - cyclic pattern: e.g.: The death rate of the predator decreases if there is large number of preys, death rate of prey increases if there are many predators.



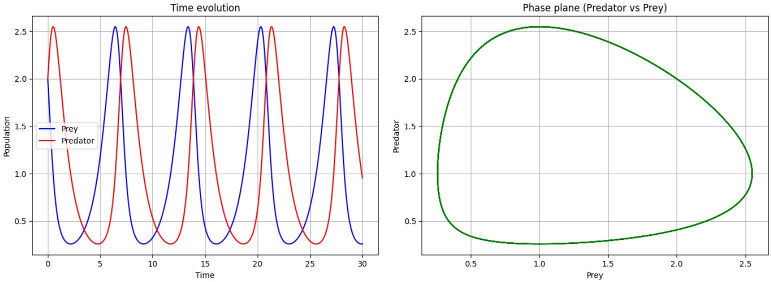
Example: Predators and prey

$$\begin{cases} \dot{x_1} = x_1(\alpha - \beta x_2) \\ \dot{x_2} = x_2(-\delta + \gamma x_1) \end{cases} \tag{5}$$

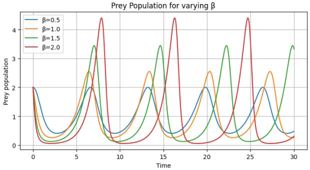
- We have 2 state variables (called x_1 for prey population and x_2 for predator population).
- This is an autonomous system, it has no inputs.
- The above model captures the main dynamics in this system - cyclic pattern: e.g.: The death rate of the predator decreases if there is large number of preys, death rate of prey increases if there are many predators.
- The resulting simulation mimics the behaviour of the real system. But it is a **deterministic approach (without any stochastic noise)**.

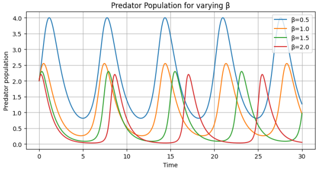


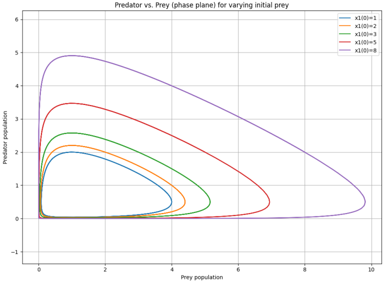
Predator-prey model



Analyze model behavior





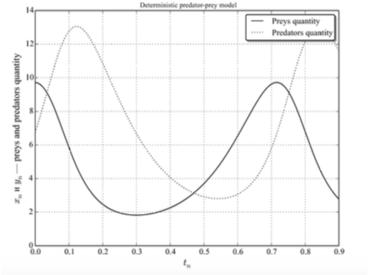


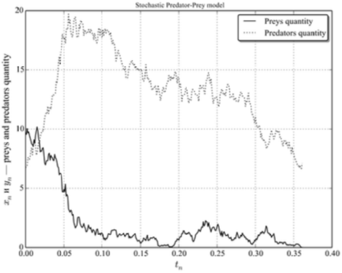
Predator-prey model

```python
alpha = 1.
beta = 1.
delta = 1.
gamma = 1.
x1_0 = 2. # initial x_1 value
x2_0 = 2. # initial x_2 value

def derivative(X, t, alpha, beta, delta, gamma):
    x1, x2 = X
    dotx1 = x1 * (alpha - beta * x2)
    dotx2 = x2 * (-delta + gamma * x1)
    return np.array([dotx1, dotx2])

def Euler(func, X0, t, alpha, beta, delta, gamma):
    """
    Euler solver.
    """
    dt = t[1] - t[0]
    nt = len(t)
    X  = np.zeros([nt, len(X0)])
    X[0] = X0
    for i in range(nt-1):
        X[i+1] = X[i] + func(X[i], t[i], alpha,  beta, delta, gamma) * dt
    return X
```

```python
Xe = Euler(derivative, X0, t, alpha, beta, delta, gamma)
```

Deterministic vs Stochastic





Why state space models?

There are several reasons why state space models are widely used:

- Natural result from physical modelling, captures internal dynamics
- Compact representation, works naturally in time domain
- Linear or nonlinear
- Easy to treat multiple inputs and outputs
- Well suited for computations, e.g. simulations

State space models

A nonlinear state-space model:

$$\dot{x}(t) = f(x(t), u(t))$$
$$y(t) = h(x(t), u(t))$$

A linear state-space model:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

- $x(t) = [x_1(t) \dots x_n(t)]^T$ is the *state vector* and n is the *order* of the model
- $u(t)$ is the *input* (possibly a vector)
- $y(t)$ is the *output* (possibly a vector)

A state-space model is an *internal* model – the state variables give more information about the system than just the input-output relation.

The state contains precisely the information that is needed ("memory") to predict the system's future trajectory, given the input signal.

Higher order differential equations can often be transformed into a state-space model. A more general form of model is the *implicit differential equation* $F(\dot{x}, x, u) = 0$.