Physical and Data Driven Modelling for Control

with Applications from the Paper Industry

Håkan Hjalmarsson and Cristian R. Rojas

Department of Automatic Control, EE School
KTH, Sweden

Outline

- Preliminaries
- Physical modelling
- Data-driven modelling

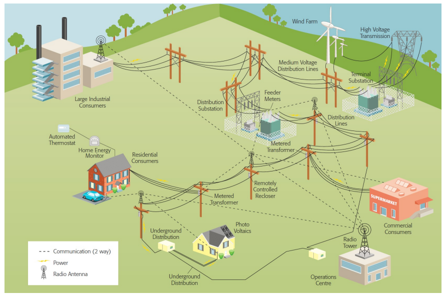
Preliminaries

Why is modelling important?

- *Engineering refers to the practice of organizing the design and construction of any artifice which transforms the physical world around us to meet some recognized need*

  G.F.C. Rogers, *The Nature of Engineering*, MacMillan Press, 1983

- To do engineering, we need to predict how reality behaves



Mathematical models and methods

- Do not require a physical system
  - Can treat new designs/technologies without prototype
  - Do not disturb operation of existing system
- Are easier to work with than real world
  - Easy to evaluate many approaches, parameter values, . . .
  - Flexible to time-scales
  - Can access unmeasurable quantities
- Support safety
  - Experiments may be dangerous
  - Operators need to train for extreme situations
- Help to gain insight and understanding

Systems

What is a system?

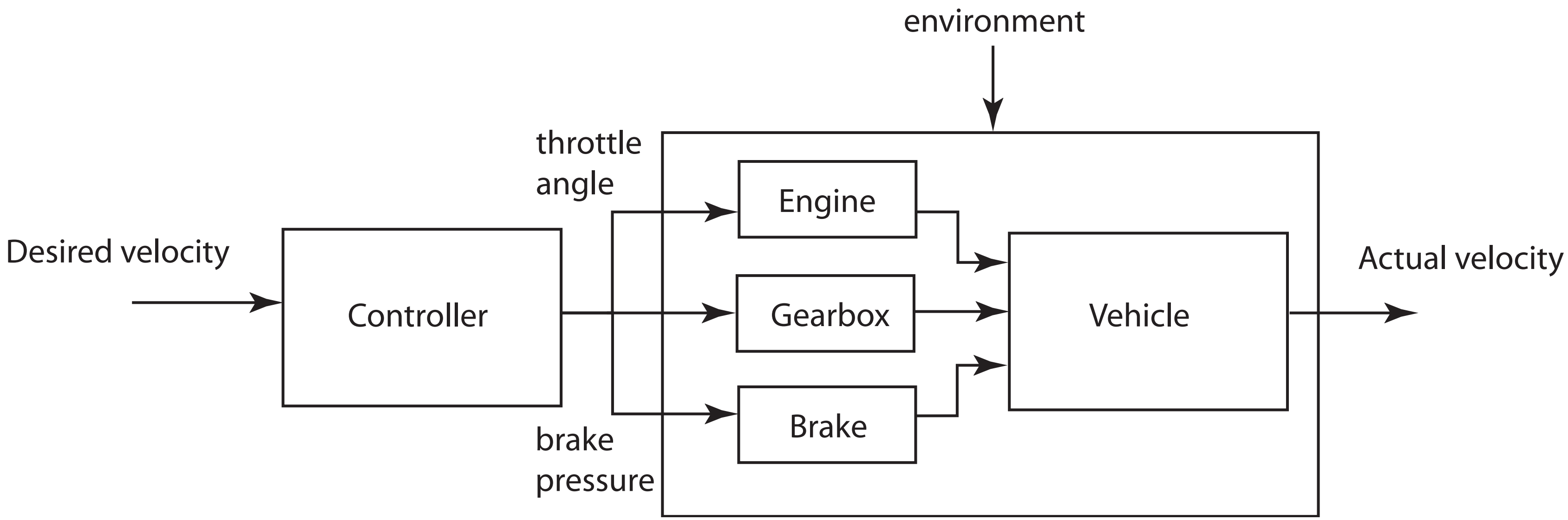
Object or collection of objects whose properties we would like to study

The systems concept

A way of structuring problems (*divide-and-conquer*)

- what belongs to the system, and what does not
- inputs, outputs and internal dynamics

Example Systems view of cruise control in a car

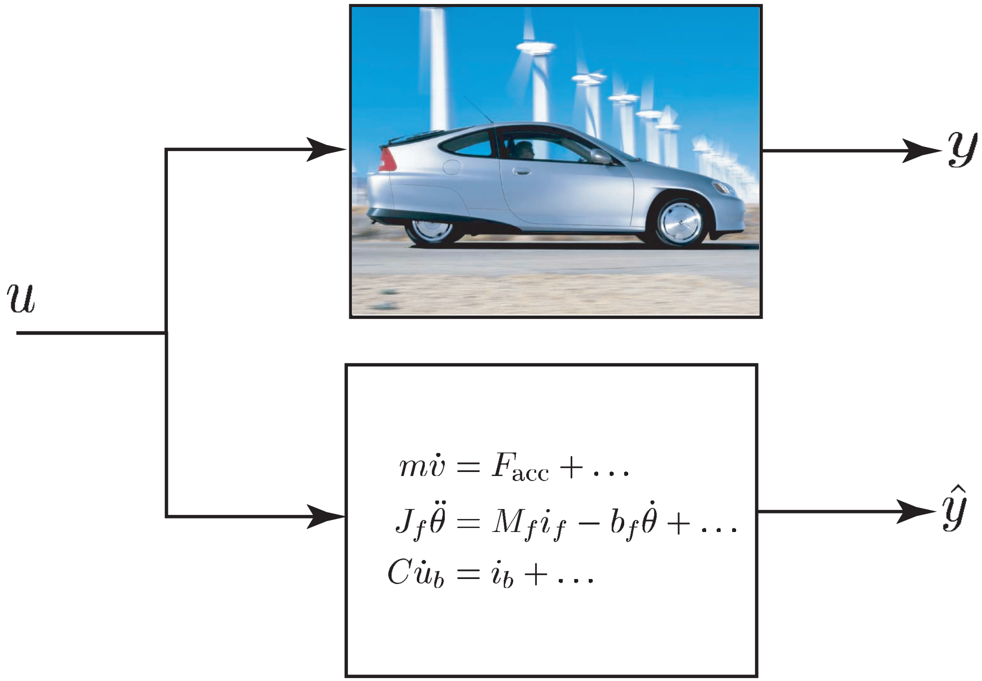


Modeling

A model M *for a system* S *and an experiment* E *is anything to which* E *can be applied in order to answer questions about* S

(Minsky, 1965)

A model is a tool we can use to avoid making real experiments



Classes of models

Many classes of models

- a piece of hardware, mental model, mathematical model, . . .

We will only consider *mathematical models*

- algebraic equations, ODEs, PDEs, finite automata, . . .

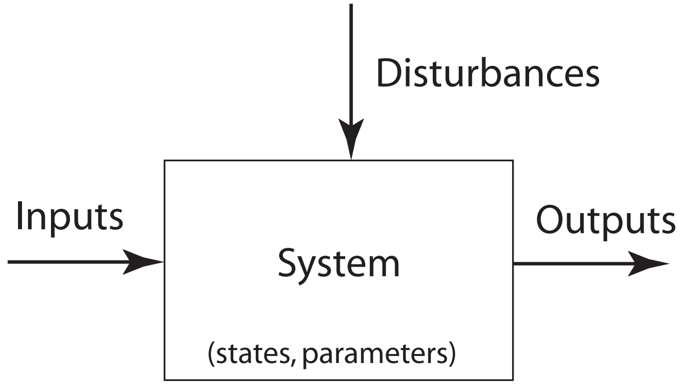
For physical modelling, we focus on differential/algebraic equations:

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), w(t)) \\ 0 &= g(x(t), u(t), w(t)) \end{aligned} \qquad \text{or} \qquad F(\dot{x}(t), x(t), u(t), w(t)) = 0$$

Systems: signals and parameters

Natural to separate model quantities into

- (constant) parameters, and
- (time-varying) signals



System parameters are fixed, *design parameters* adjustable

Signals are external (*inputs, disturbances*) or internal (*states*)

Mathematically, a system is a *mapping* between signals

State-space models

Many systems naturally described by differential equations

$$\begin{aligned}
\dot{x}_1(t) &= f_1\left(x_1(t), \ldots, x_n(t), u_1(t), \ldots, u_m(t)\right) \\
&\vdots \\
\dot{x}_n(t) &= f_n(x_1(t), \ldots, x_n(t), u_1(t), \ldots, u_m(t))
\end{aligned}$$

or, in vector notation

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

States x_i typically describe aggregation of a conservative quantity

– tank levels, momentum of masses, voltages across capacitors, . . .

Output signals depend *algebraically* on states and inputs, *i.e.*,

$$y(t) = g(x(t), u(t))$$

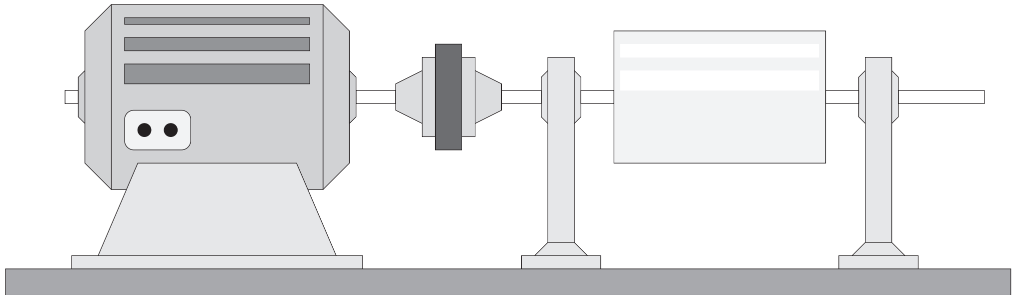
How to build mathematical models?

Two basic approaches

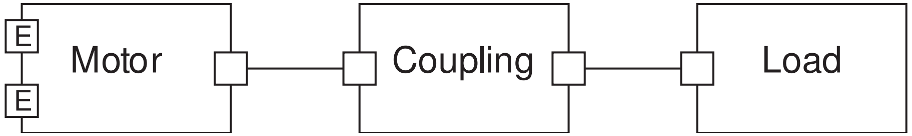
- **Physical modelling**
  - Use first principles, laws of nature, etc. to model components
  - Need to understand system and master relevant facts!
- **System identification**
  - Use experiments and observations to deduce model
  - Need prototype or real system!

Example: physical modeling

DC motor with flexible coupling



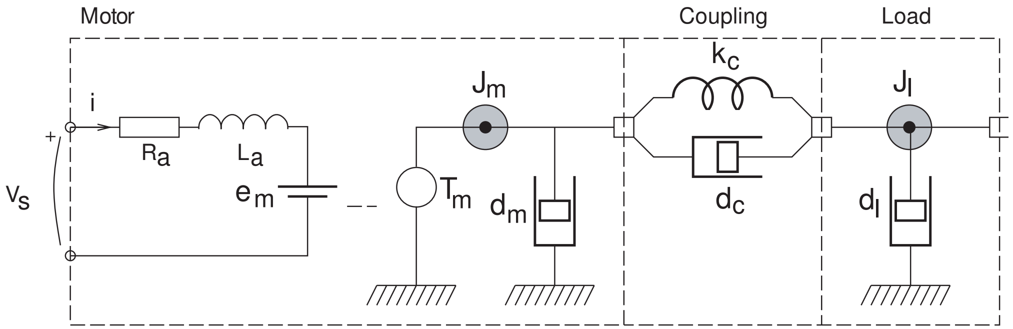
A schematical illustration of the system structure



Example from lecture notes by S-E Mattsson, LTH

Example: physical modeling

More detailed schematic



State equations

$$L_a \frac{di}{dt} = V_s - R_a i - \underbrace{k_m \omega_m}_{e_m}$$

$$\frac{d\theta_m}{dt} = \omega_m \qquad J_m \frac{d\omega_m}{dt} = \underbrace{k_m i}_{T_m} - d_m \omega_m - k_c(\theta_m - \theta_l) - d_c(\omega_m - \omega_l)$$

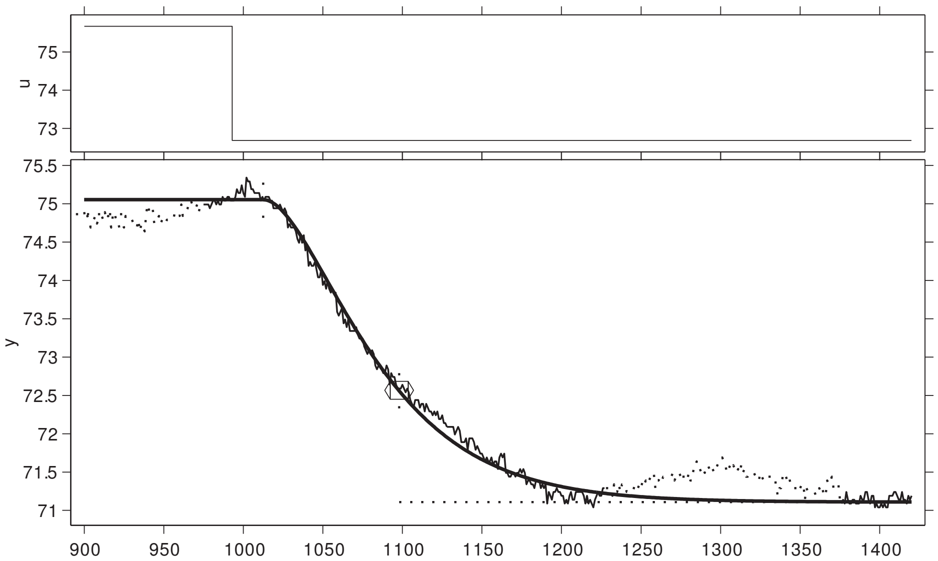
$$\frac{d\theta_l}{dt} = \omega_l \qquad J_l \frac{d\omega_l}{dt} = -d_l \omega_l - k_c(\theta_l - \theta_m) - d_c(\omega_l - \omega_m)$$

Example: system identification

Model of a starch boiler in a Swedish paper mill

– how does supplied steam influence boiler temperature?

– make step change in steam supply, observe temperature:



Transfer function fit

$$Y(s) \approx \frac{1.34e^{-12.9s}}{(43.3s+1)^2}U(s)$$

Model good enough for controller tuning!

Example from Panagopoulos *et al.* (2000)

All models are approximate!

All models are wrong, but some are useful (G.E.P. Box)

A model captures only some aspects of a system

- Important to know which aspects are modelled and which are not
- Make sure that model is valid for intended purpose

All-encompasing models often a bad idea

- Large and complex – hard to gain insight
- Cumbersome and slow to manipulate
- Difficult to estimate from data (too many parameters)

Good models are simple, yet capture the essentials!

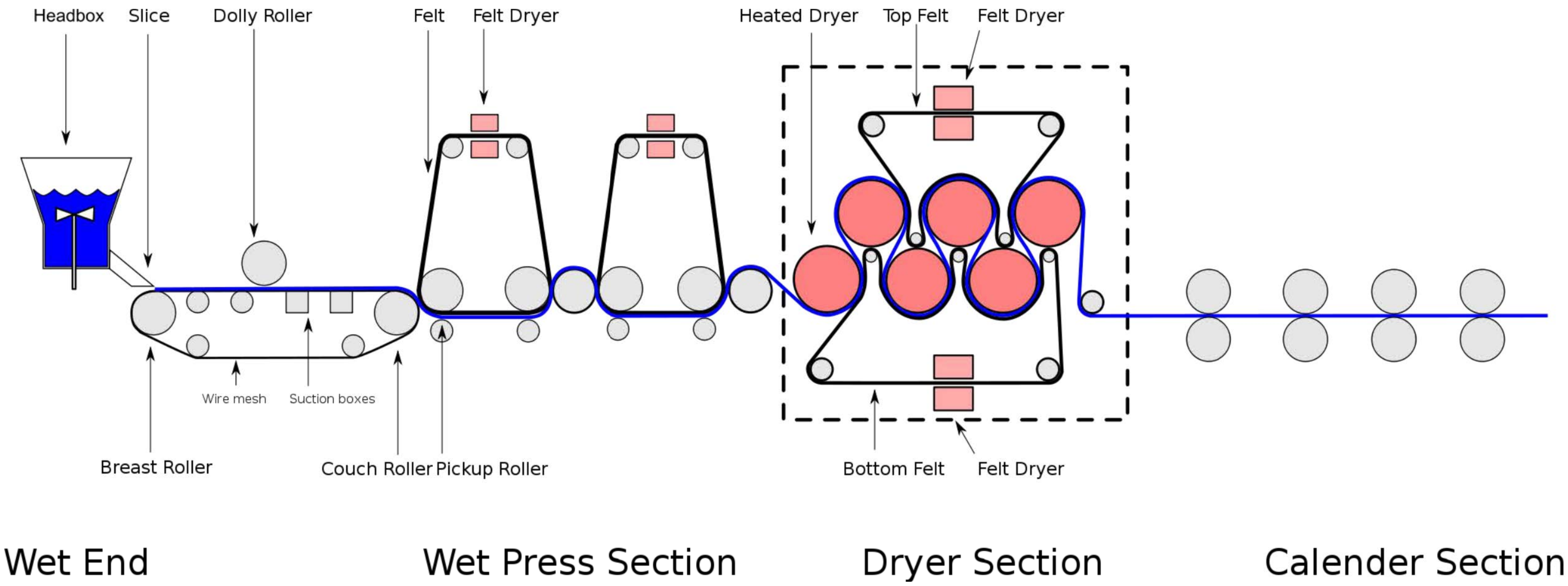
Physical Modelling

Outline of Physical Modelling

- Motivation
- Review of physical domains
- How to build a model
- Modelica

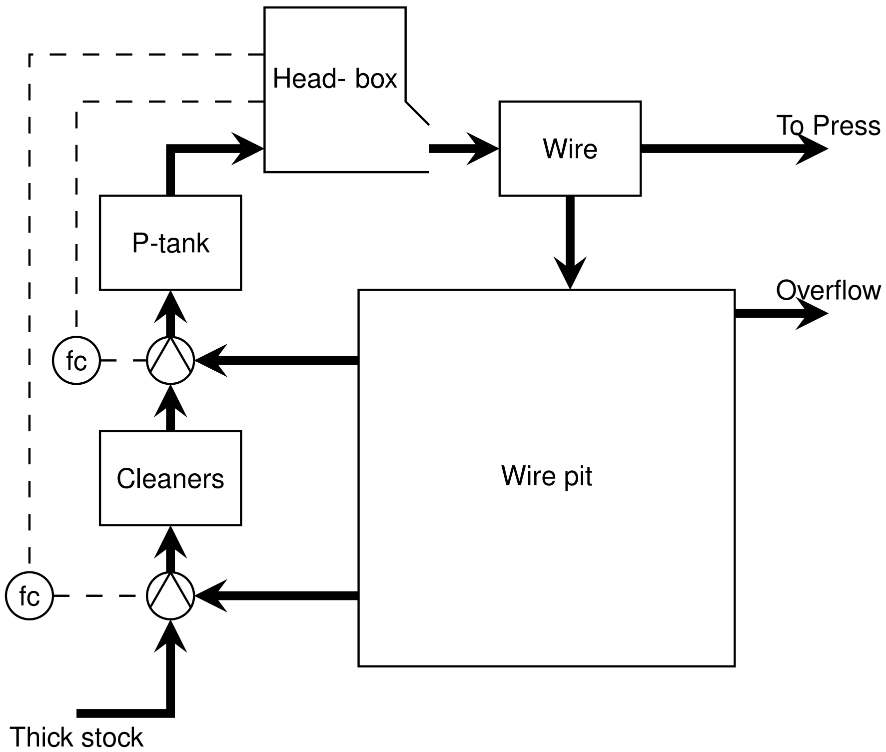
Motivation: paper machine

A complex water removal machine!



Motivation: paper machine (cont.)

Basic structure of the wet end:



Cortesy of Olle Trollberg

We will present several examples inspired by parts of the wet end

Review of physical domains

We will review component models and interconnection rules from

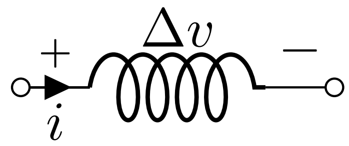
- electronics
- mechanics (translational/linear, rotational)
- hydraulics

Electrical circuits – components

Relations based on conservation of energy and electric charge, where voltage = $v(t)$ and current = $i(t)$

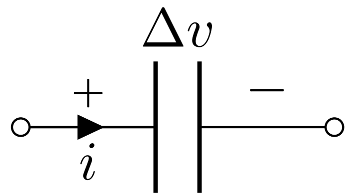
Inductors (accumulation of magnetic energy)

$$\frac{di(t)}{dt} = \frac{1}{L}\Delta v(t)$$



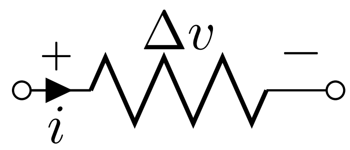
Capacitors (accumulation of charge)

$$\frac{d\Delta v(t)}{dt} = \frac{1}{C}i(t) \;\left(\frac{dQ(t)}{dt} = i(t),\; \Delta v(t) = \frac{1}{C}Q(t)\right)$$



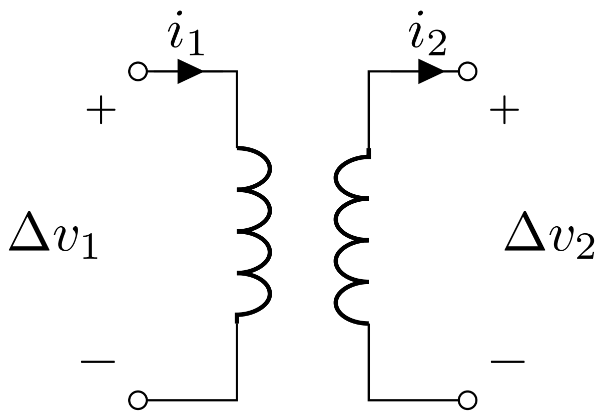
Resistors (dissipation of energy)

$$\Delta v(t) = Ri(t)$$



Electrical circuits – components

Transformers: loss free change of current and voltage



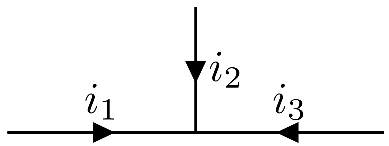
$$
\begin{aligned}
\Delta v_1 i_1 &= \Delta v_2 i_2 \\
\Delta v_1 &= \alpha \Delta v_2 \\
i_1 &= \frac{1}{\alpha} i_2
\end{aligned}
$$

Electrical circuits – interconnections

Interactions between components governed by Kirchoff's laws

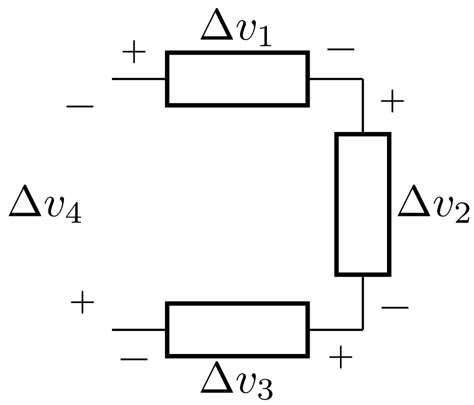
Currents sum to zero at interconnections (no accumulation of charge)

$$\sum_k i_k = 0$$



Sum of voltage drops around closed circuit is zero

$$\sum_k \Delta v_k = 0$$

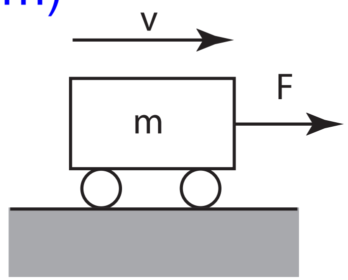


Translational mechanics – components

Relations based on conservation of linear momentum $\boldsymbol{p}(t)$ and energy, where flow of momentum = force = $\boldsymbol{F}(t)$

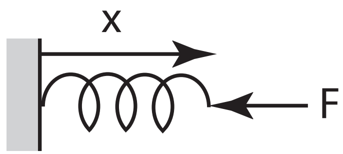
Newton's 2nd law (accumulation of linear momentum)

$$\frac{d\boldsymbol{p}(t)}{dt} = \boldsymbol{F}(t), \quad \frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{v}(t) = \frac{\boldsymbol{p}(t)}{m}$$



Springs (accumulation of potential energy)

$$\boldsymbol{F}(t) = k\boldsymbol{x}(t)$$



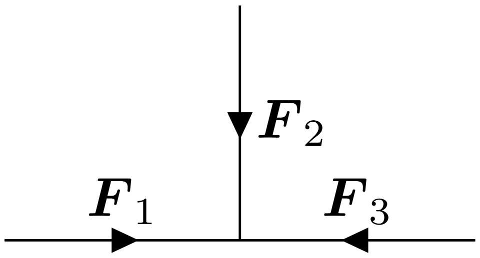
Dampers (dissipation of energy)

$$\boldsymbol{F}(t) = \gamma\boldsymbol{v}(t)$$

Translational mechanics – interconnections

Forces at a node sum to zero

$$\sum_k \boldsymbol{F}_k = 0$$



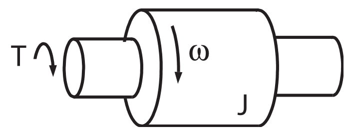
Displacements are equal at interconnection points

Rotational mechanics – components

Relations based on conservation of angular momentum $\boldsymbol{L}(t)$ and energy, where flow of angular momentum = torque = $\boldsymbol{\tau}(t)$

Newton's 2nd law for rotation (accumulation of angular momentum)

$$\frac{d\boldsymbol{L}(t)}{dt} = \boldsymbol{\tau}(t), \quad \frac{d\boldsymbol{\theta}(t)}{dt} = \boldsymbol{\omega}(t) = \boldsymbol{J}^{-1}\boldsymbol{L}(t)$$



Torsional spring (accumulation of potential energy)

$$\boldsymbol{\tau}(t) = k\boldsymbol{\theta}(t)$$

Torsional friction (dissipation of energy)

$$\boldsymbol{\tau}(t) = \gamma\boldsymbol{\omega}(t)$$

Rotational mechanics – interconnections

Interconnection rules analogous to translational mechanics

Torques at a point sum to zero

$$\sum_k \boldsymbol{\tau}_k = 0$$

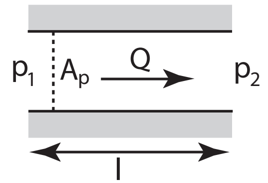
Angular displacements equal at interconnection points

Hydraulic systems – components

Models relation based on conservation of mass and linear momentum, where pressure = $P(t)$ and mass flow = $Q(t)$

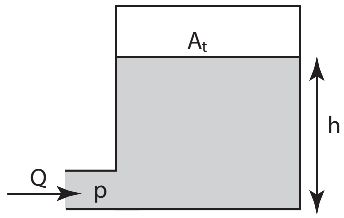
Pipes: pressure drop drives flow (accumulation of linear momentum)

$$\frac{dQ(t)}{dt} = \frac{1}{L_f}\Delta P(t), \quad \Delta P(t) = P_1(t) - P_2(t)$$



Tanks (accumulation of mass)

$$\frac{dP(t)}{dt} = \frac{1}{C_f}Q(t) \left(\frac{dm(t)}{dt} = Q(t),\ P(t) = \frac{1}{C_f}m(t)\right)$$



Flow resistance (dissipation of energy)

$$\Delta P(t) = R_f Q(t)$$

Note: $L_f = \rho l/A_p \qquad C_f = A_t/(\rho g)$

Hydraulic systems – interconnections

Flows sum to zero at interconnections (no accumulation of mass)

$$\sum_k Q_k = 0$$

Pressure is equal at interconnection points

How to build a model?

- Bond-graphs
  - H. Paynter (*Analysis and design of engineering systems*, 1961)
  - Based on energy transfer and similarities between domains
- Variational approach
  - Tools: calculus of variations
  - Started with Lagrangian/Hamiltonian mechanics:
    (see C. Lanczos, *The Variational Principles of Mechanics*)
  - Based on the concept of *equilibrium* (used even in disciplines such as economics and finance)
- $\cdots$

We will develop a strategy inspired by, but more general than, bond graphs

Basic principles

- *Causality*

  Each equation determines one variable as a function of others

- *Conservation laws / balance equations*

  Most differential eqns come from the conservation of something

- *Objectivity*

  Balance equations have to be specified for given enclosures, reference frames, etc.

- *Aggregation*

  Practical models are approximations, built by aggregating spatially microscopic and/or temporally fast effects

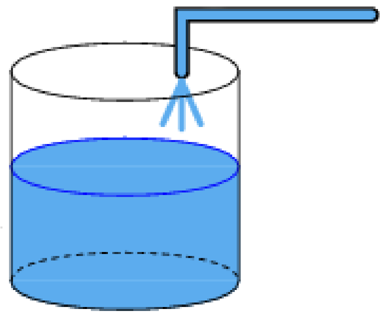
Causality

We want to derive models of the (ODE) form

$$\frac{dx(t)}{dt} = f(t), \qquad x(t) = \int_{t_0}^{t} f(\tau)d\tau + x(t_0)$$

Discretizing these equations gives

$$x(t + dt) \approx x(t) + f(t)dt$$



x : mass in the tank

f : mass inflow

These equations can be interpreted as: f *is the cause of* x

Causality is not always *physically real*, but it is of great computational and explanatory value

Note: Not everyone agrees; see J.C. Willems, "The behavioral approach to open and interconnected systems". *IEEE Control Syst. Mag.*, 27(6): 46-99, 2007.

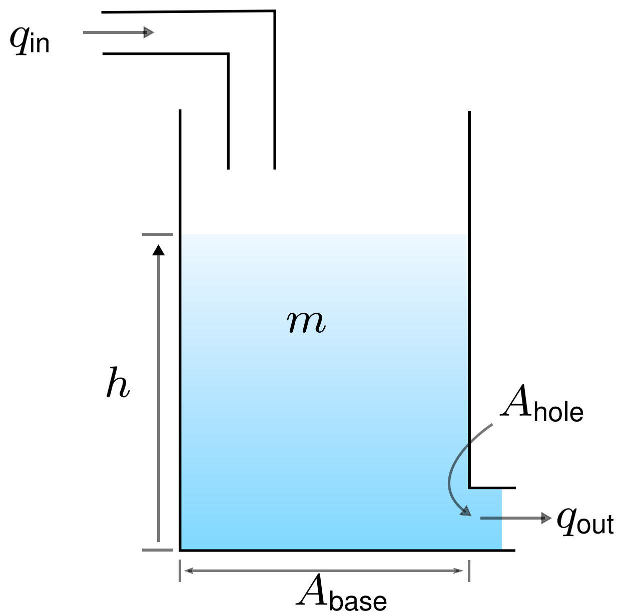
Conservation laws & constitutive relations

Most differential equations in a model are *balance/conservation equations*, of the form

$$\begin{aligned} \text{volume change per time unit} &= \text{inflow} - \text{outflow} \\ \text{accumulated energy per time unit} &= \text{power in} - \text{power out} \\ 0 &= \text{sum of all currents entering a node} \end{aligned}$$

The remaining (typically static/algebraic) equations are called *constitutive relations*

Example: water tank



$$\frac{dm(t)}{dt} = q_{\text{in}}(t) - q_{\text{out}}(t)$$ balance equation

$$q_{\text{out}}(t) = A_{\text{hole}}\rho\sqrt{2g\max\{h(t), 0\}}$$ constitutive relation

$$h(t) = m(t)/(\rho A_{\text{base}})$$ constitutive relation

Conservative quantities

The identification of conservative quantities in a model is crucial!

There are 5 types of conservative quantities in (classical) physics:

- Energy ($dE/dt =$ power)
- Mass ($dm/dt =$ mass flow)
- Electric charge ($dQ/dt =$ electric current)
- Linear momentum ($d\boldsymbol{p}/dt =$ force)
- Angular momentum ($d\boldsymbol{L}/dt =$ torque)

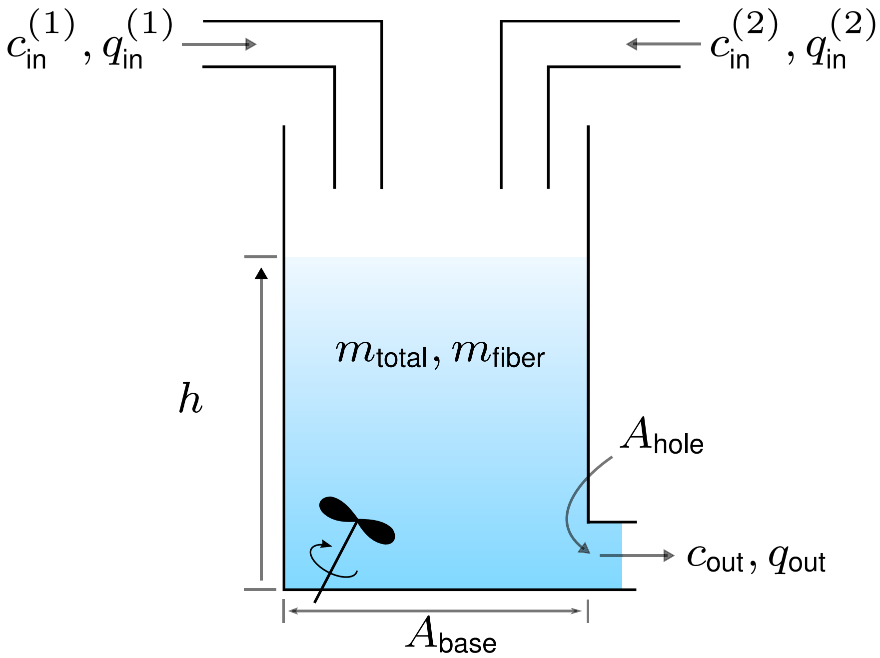
Chain rule (convective flows)

To compute the flows for the balance equations, it is important to notice that, often, the conservative quantity flows “on top” of another one; the flow can thus be determine via a *chain rule*

Example (Energy)

$$
\begin{aligned}
\frac{dE}{dt} &= \frac{dE}{dq}\frac{dq}{dt} = v \cdot i && \text{(electrical components)} \\
&= \frac{dE}{dm}\frac{dm}{dt} = P \cdot Q && \text{(hydraulic systems)} \\
&= \frac{dE}{d\boldsymbol{p}} \cdot \frac{d\boldsymbol{p}}{dt} = \boldsymbol{v} \cdot \boldsymbol{F} && \text{(translational mechanics)} \\
&= \frac{dE}{d\boldsymbol{L}} \cdot \frac{d\boldsymbol{L}}{dt} = \boldsymbol{\omega} \cdot \boldsymbol{\tau} && \text{(rotational mechanics)}
\end{aligned}
$$

Example: Concentration of fiber in a tank

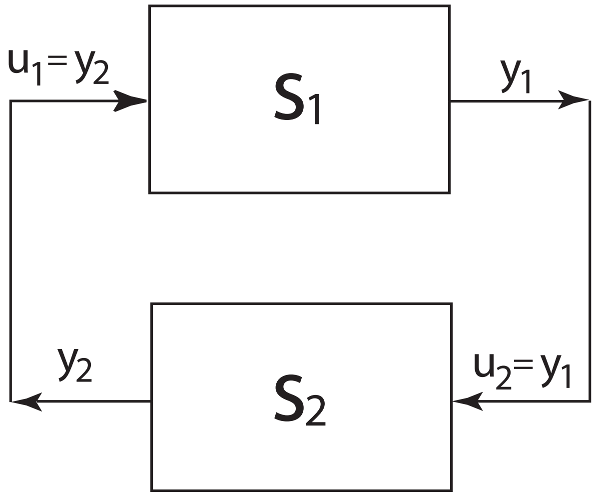


A tank is being filled by a mixture of fiber and water

Goal Determine the concentration in the tank

Causality conflicts: *algebraic loops*

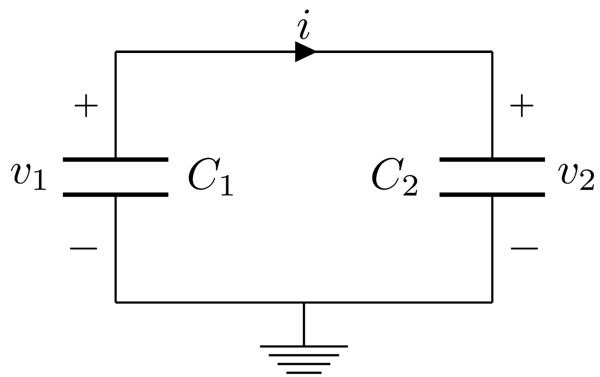
Interconnection of state-space models not always state-space model



$$S_1 : \begin{cases} \dot{x}_1 = f_1(x_1, u_1) \\ y_1 = h_1(x_1, u_1) \end{cases} \qquad S_2 : \begin{cases} \dot{x}_2 = f_2(x_2, u_2) \\ y_2 = h_2(x_2, u_2) \end{cases}$$

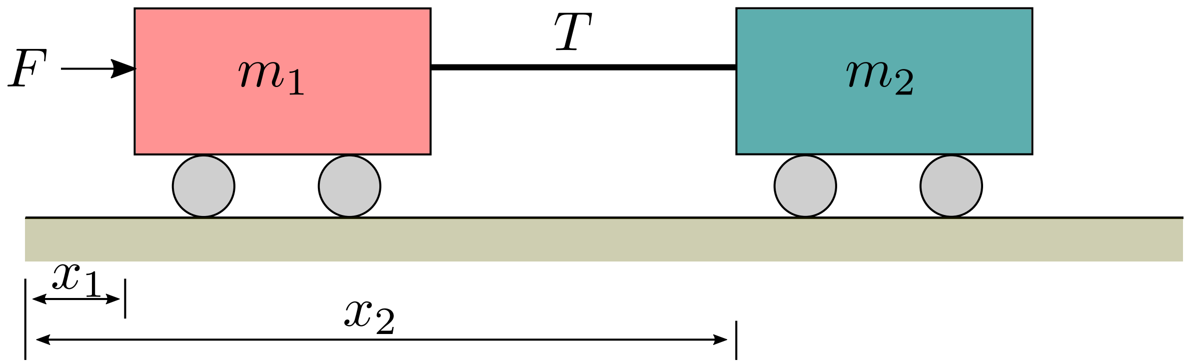
Need to solve $u_1 = h_2(x_2, h_1(x_1, u_1))$ to find consistent states

Example: Coupled capacitors



Can we compute the voltage drops v_1, v_2 across the capacitors?

Example: Coupled masses



Length of each cart = l

Distance between the carts = l_0

Goal Compute the tension T on the bar connecting the masses

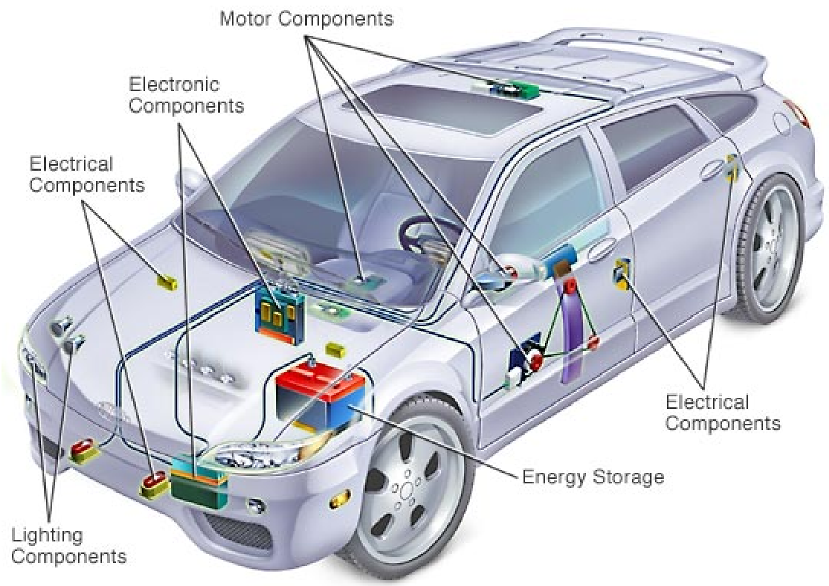
Origin of algebraic loops

Separation of time constants: good models focus on dynamics whose time constants are relevant for the intended purpose of the model

- fast dynamics approximated as algebraic relations
- variables that vary slowly are approximated by constants

Give models that are easier to manipulate and simulate

Example Electrical and mechanical dynamics of a car have very different time scales!



How to solve algebraic loops

Motto: Nature does not solve algebraic equations!

- *Singular perturbations ("stiff compliance")*
  Since algebraic loops may come from neglected fast dynamics, we can re-enter them into our model as singular perturbations:

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

$$\varepsilon \frac{dy}{dt} = g(x, y, \varepsilon), \qquad 0 < \varepsilon \ll 1$$

Potential (stiff) numerical problems!

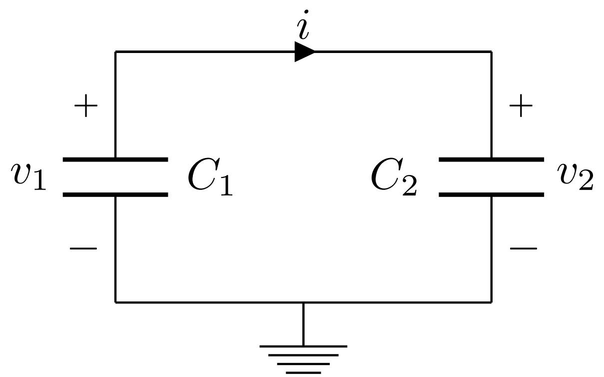
- *Differential-algebraic equations (DAEs) and hybrid systems*
  Letting $\varepsilon \to 0$ in the singularly perturbed system gives

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

$$0 = g(x, y, \varepsilon)$$

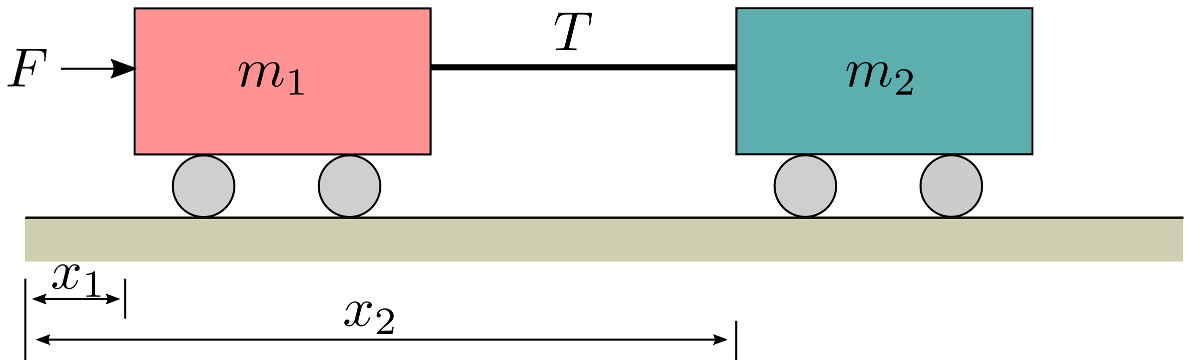
Can be solved using Modelica (more later!)

Example: Coupled capacitors (cont.)



Can we compute the current i flowing through the capacitors?

Example: Coupled masses (cont.)

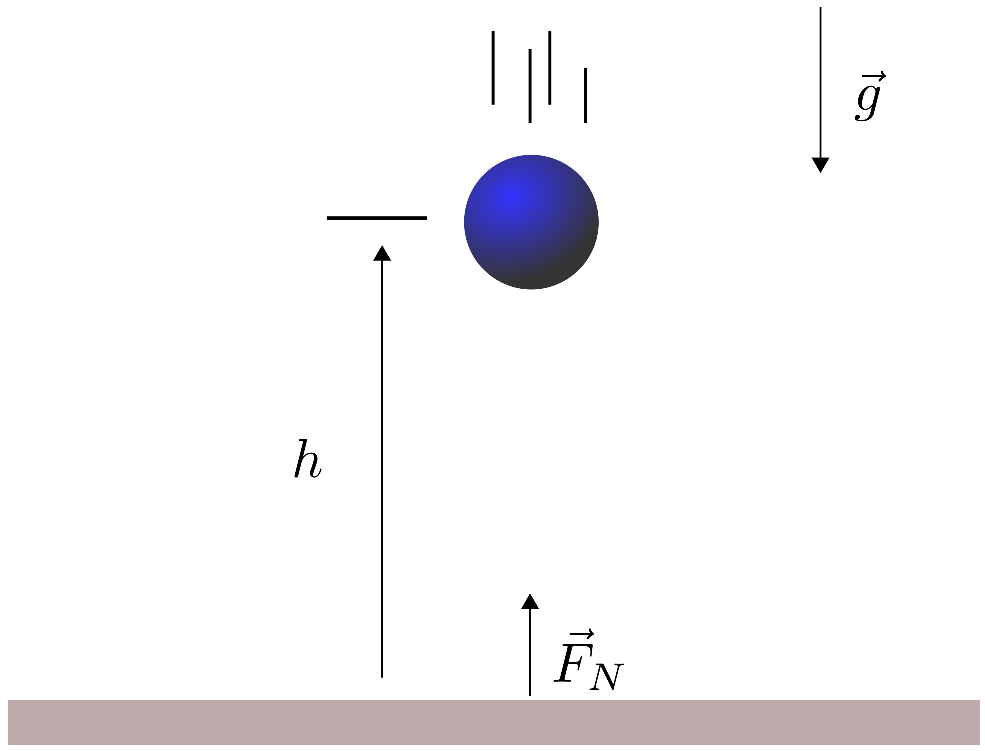


Length of each cart = l

Distance between the carts = l_0

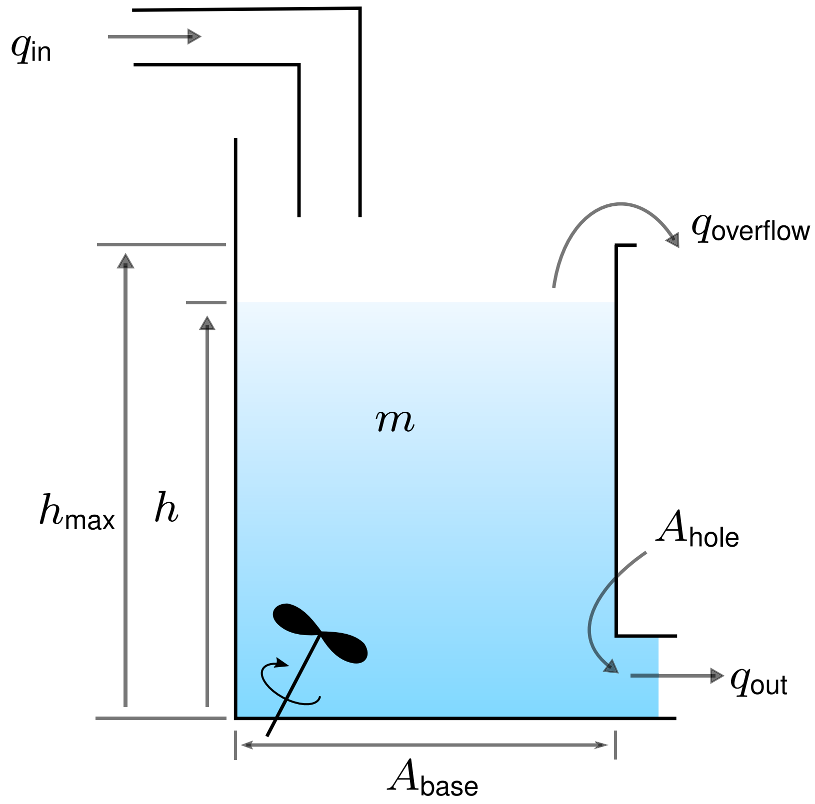
Goal Compute the tension T on the bar connecting the masses

Example: A bouncing ball



Goal Determine the height of the ball

Example: Tank with overflow

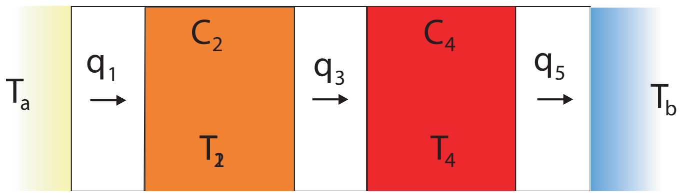


Goal Determine the water level h inside the tank

Aggregation

In modelling, one often has to aggregate fast dynamics (treating them as instantaneous/algebraic) and spatially distributed phenomena

Example *Compartment model of heat transfer*



$$\frac{\partial}{\partial t}T(x,t) = a(x)\frac{\partial^2}{\partial x^2}T(x,t)$$

Partition rod into compartments, with (assumed) constant temperature

$$\frac{d}{dt}T_i = \frac{1}{C}(P_i^{\text{in}} - P_i^{\text{out}})$$

For non-boundary compartments, we have

$$P_i^{\text{in}} = K(T_{i-1} - T_i) \qquad P_i^{\text{out}} = K(T_i - T_{i+1})$$

Infinite-dimensional description (PDE) modelled by system of ODEs

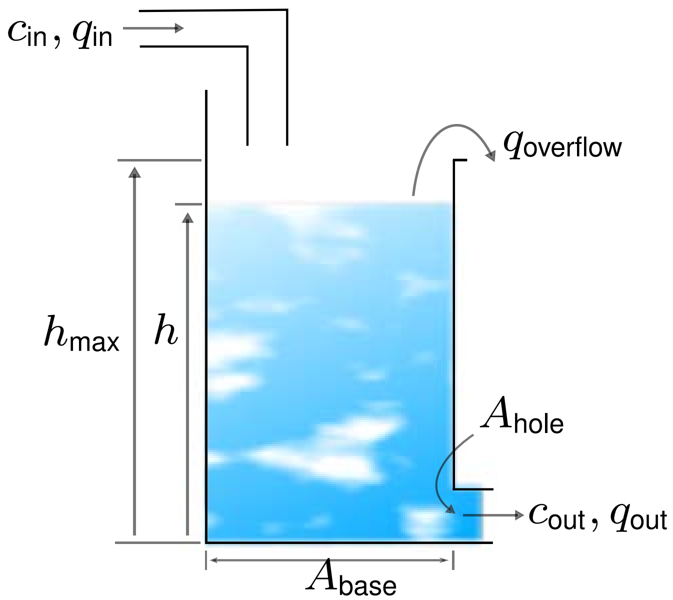
Example: Conveyor belt

Length of conveyor belt = l

Radius of wheels= r

Goal Determine out flow q_{out}

Example: Plug flow tank



The concentration in the tank is not uniform!

Goal Determine the outflow q_{out} and its concentration c_{out}

Modelica

Modelica basics

Approach

- Equation-based component descriptions
- Object-oriented model libraries
  - well-defined interfaces
  - hierarchical modelling, inheritance
- Software tools for
  - analysis and consistency checking
  - numerical simulation (of differential-algebraic equations)

Equation-based Modeling

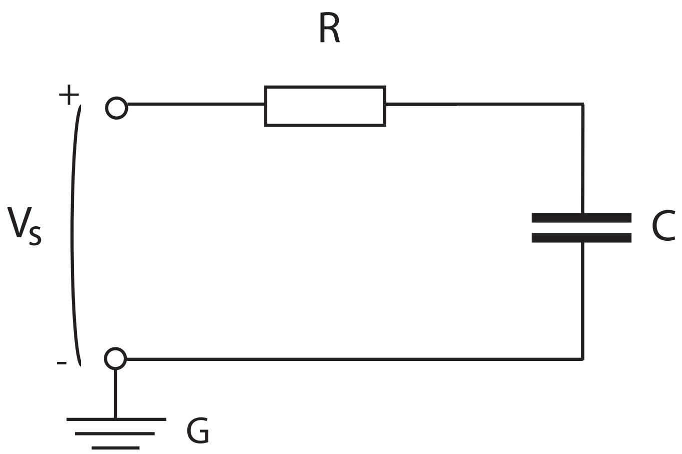
Nonlinear RC-circuit

$$u_S = A\sin(t)$$
$$C\frac{d}{dt}u_C = i$$
$$u_R = R(i + i^5)$$
$$u_S - u_C - u_R = 0$$

Modelica code

```
model nonlinear_RC
  Real uS, uR, uC, i;
  parameter Real C=1, R=1, A=1;
equation
  uS = A*sin(time);
  C*der(uC) = i;
  uR = R*(i + i^5);
  uS - uC - uR = 0;
end nonlinear_RC;
```

Object-oriented modeling of electrical circuit



To develop reusable models in Modelica, we need to specify

- interconnection rules for electrical systems
- component relations
- how components are interconnected to form full system

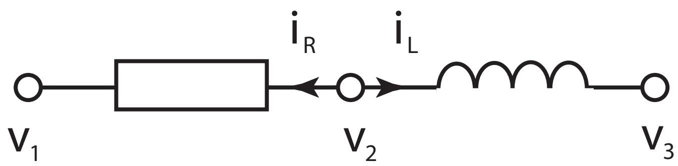
Interconnection rules in Modelica

For each interconnection, we specify flow and effort variables

Flow variables: sum to zero at every node

Effort variables: are equal at interconnection points

Example Interconnections for electrical components



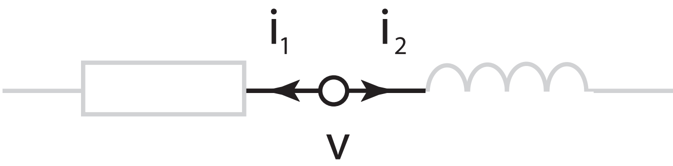
Flow variable: current(i.e., $-i_L - i_R = 0$), effort variable: potential

Connectors

Connection rules defined using “connector” construct

```
connector Pin "Pin of electrical component"
       Modelica.SIunits.Voltage v "Potential at pin";
  flow Modelica.SIunits.Current i "Current flowing into pin";
end Pin;
```

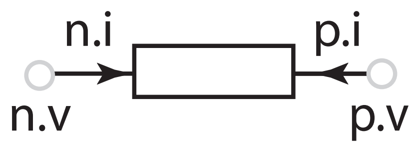
Flow variables sum to zero, effort variables are equal



Generic one-port component

Common behaviour of components specified in “partial model”

```
partial model OnePort
  "Component with two electrical pins (p and n)
   and a current i flowing from p to n"

  Modelica.SIunits.Voltage v "Voltage drop between the two pins";
  Modelica.SIunits.Current i "Current from pin p to pin n";

  Pin p;
  Pin n;
equation
  v = p.v - n.v    "Voltage drop across component";
  0 = p.i + n.i;
  i = p.i        "Current through component";
end OnePort;
```



Component code

Resistor

```
model Resistor "Ideal linear electrical resistor"
  extends   OnePort;
  parameter Modelica.SIunits.Resistance R=1 "Resistance";

equation
  R*i=v;
end Resistor;
```

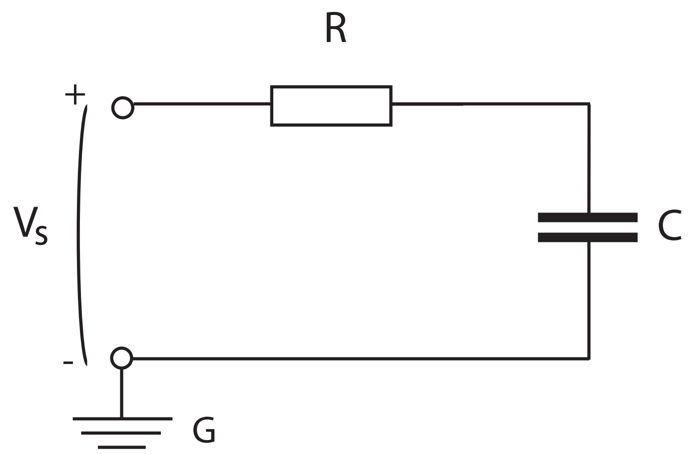
Capacitor

```
model Capacitor "Ideal linear electrical capacitor"
  extends   OnePort;
  parameter Modelica.SIunits.Capacitance C=1 "Capacitance";

equation
  i = C*der(v);
end Capacitor;
```

Interconnecting components

Models define: components used, parameter values, interconnections

```
model RCcircuit
  Vsource   Vs(u=1);
  Capacitor C(C=0.01);
  Resistor  R(R=1);
  Ground    G;
equation
  connect(Vs.p, R.p);
  connect(R.n, C.p);
  connect(C.n, Vs.n);
  connect(Vs.n. G.p);
end RCcircuit;
```



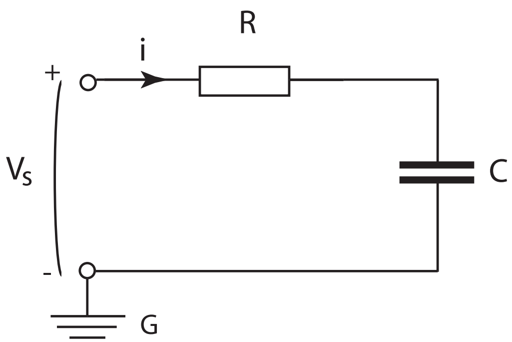
Resulting equations

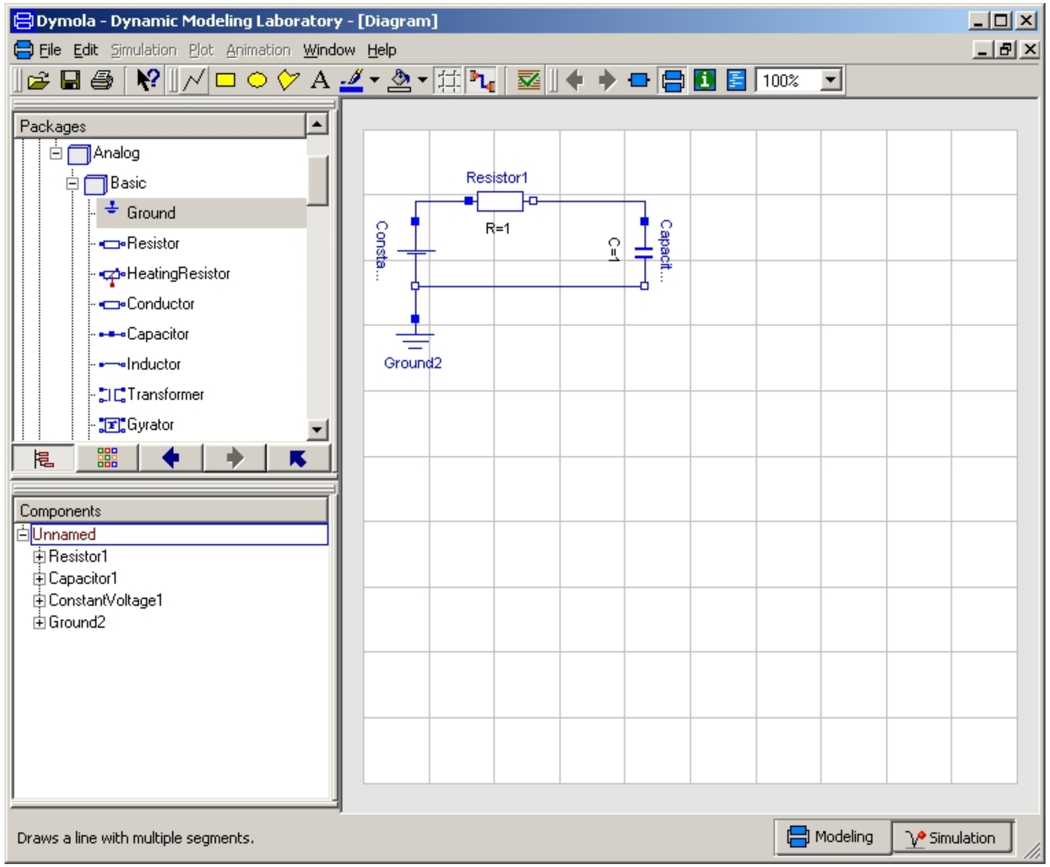
Resulting equations

```
Vs.p.v-R.p.v=0
Vs.p.i+R.n.i=0
R.v=R.R*R.i

R.p.v-C.p.v=0
R.p.i+C.p.i=0
C.C*der(C.v)=C.i

C.p.v-Vs.n.v=0
C.p.i+Vs.n.i=0

Vs.n.v=0
Vs.p.v=u
```

Easily simplified to standard relations

Electrical circuit example





We can also build a model using the Modelica standard library

Modelica standard library

Modelica (top level)

Blocks, Constants, Electrical, Icons, Math, Mechanics, SIunits

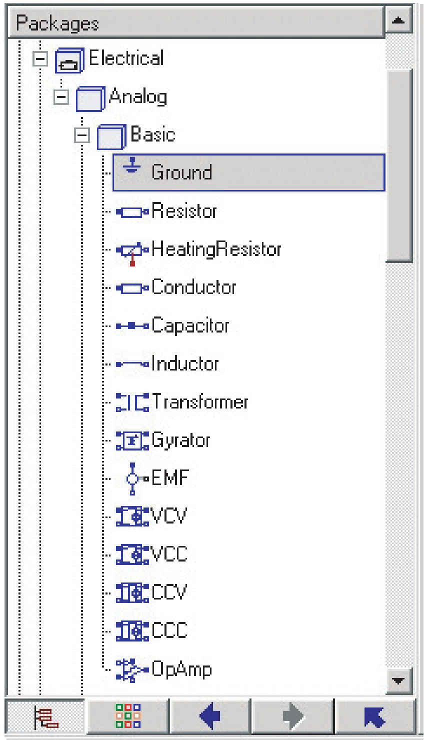
Modelica.Electrical.Analog

Basic, Ideal, Interfaces, Lines, Semiconductors, Sensors, Sources

Modelica.Electrical.Analog.Basic

Ground, Resistor, Capacitor, Inductor, Transformer, Gyrator,
EMF, controlled sources (VCV, VCC, CCV, CCC)

Basic analog circuits library



Re-use models!

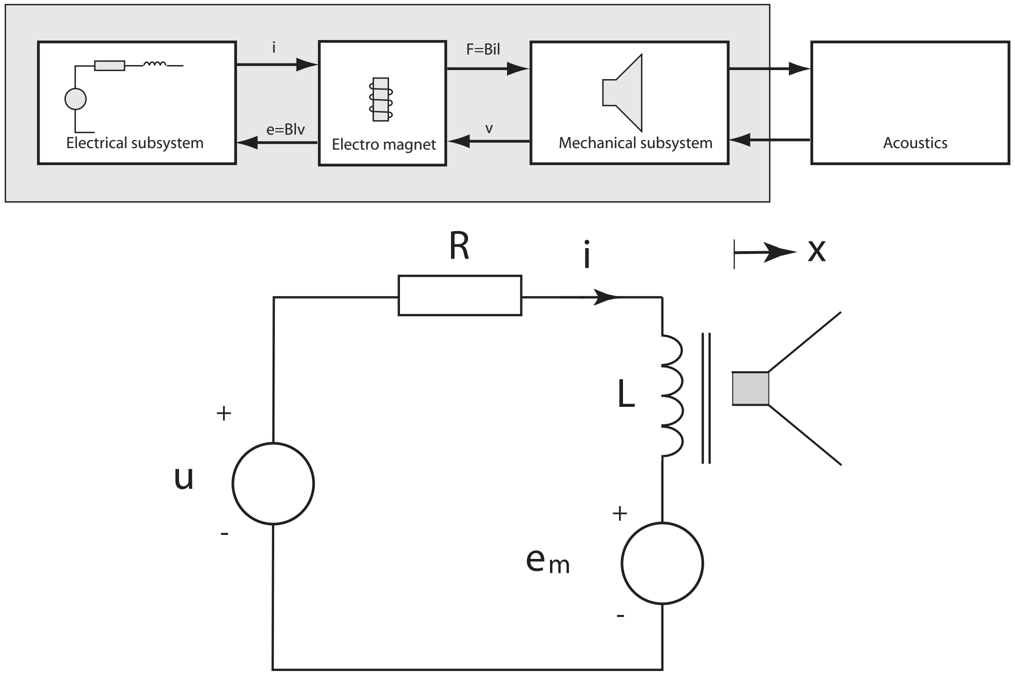
Try first to

- use existing library components, or
- modify library components

Develop new model classes only when this is not feasible

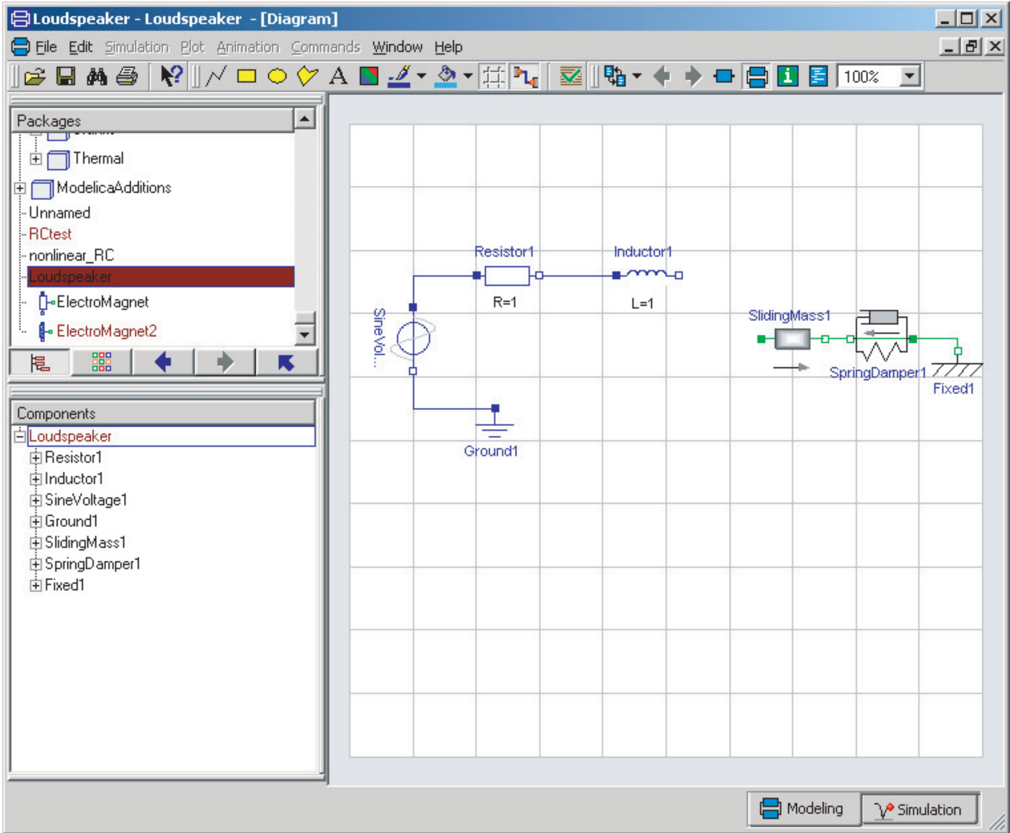
Example: loudspeaker modeling in Modelica

Consider the loudspeaker model from Lecture 2



Example: loudspeaker modeling in Modelica

Most of the model can be constructed using standard components



We only need to construct a component for the electro magnet

Modelica model of electromagnet

Transforms from electrical to translational mechanics domain

- need to understand connectors for translational mechanics!

Flange

```
connector Flange_b "right 1D translational flange";
                   "(flange axis directed OUT OF cut plane)";
  Modelica.SIunits.Position s "Absolute position of flange";
  flow Modelica.SIunits.Force f "cut force directed into flange";
end Flange_b;
```

Modelica code for electromagnet

```
model ElectroMagnet
  parameter Real B(final unit="Tesla") = 1;
  parameter Real l(final unit="m") = 1;

  Modelica.SIunits.Voltage e;
  Modelica.SIunits.Current i;
  Modelica.SIunits.Velocity v;

  Modelica.Mechanics.Translational.Interfaces.Flange_b flange_b;
  Modelica.Electrical.Analog.Interfaces.PositivePin p;
  Modelica.Electrical.Analog.Interfaces.NegativePin n;
equation
  e = p.v - n.v;
  0 = p.i + n.i;
  i = p.i;

  v = der(flange_b.s);
  e = B*l*v;
  flange_b.f = B*i*l;
end ElectroMagnet;
```

Complete Loudspeaker Model