Welcome to

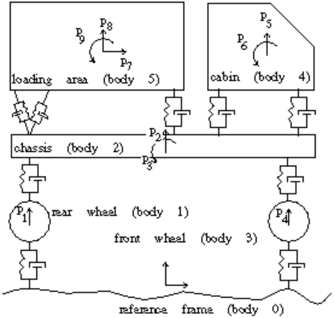
TMA4215 Numerical Mathematics

https://wiki.math.ntnu.no/tma4215/2013h/start

Lecturer:	Anne Kværnø
Course assistants:	Eirik Hoel Høiseth
	Asif Mushtaq (projects).

Case: A Truck Model

Mathematical model:



Physics:
Newton's second law $F = ma$.

Rear wheel:

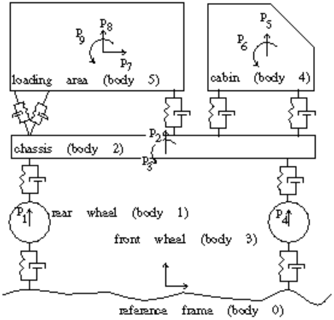
$$m_1 p_1'' = f_{10} + f_{12} - m_1 g$$

where e.g.:

$$f_{10} = k_{10}(p_1 - u(t)) + d_{01}(p_1' - u'(t)) + f_{10}^0$$

p_1	Vertical position
m_1	Mass of the
g	Gravitational constant
k_{01}, d_{01}	Stiffness and damping constants.
$u(t)$	From the road

Similar for $p_2, p_3, \ldots, p_9$.

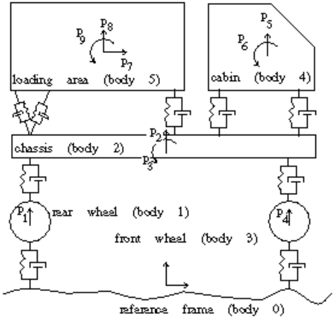


Putting everything together:

- We get a system of 9 second order differential equations

$$Mp'' = f(t, p, p'),$$

- which you will hardly solve by hand.
- In December, you will not only be able to solve this problem numerically
- but you have learned how to construct your own algorithms for doing so.



Objective of the course

Given a class of problems (linear or nonlinear equations, integrals, differential equations), you will learn how to

- develop a numerical algorithm
- implement it (computer program)
- do convergence/stability analysis
- do verification (testing, testing, testing)

The focus is on ideas and principles.

Your background

- Mathematics 1-4
- Some programming experience (MATLAB or Python)
- TMA4145 Linear methods this fall is not required, but will be an advantage.

Taylor's theorem

Theorem (A.4 and A.5)

If f is a $n+1$ times continuous function, then

$$f(x) = f(a) + (x-a)f'(a) + \cdots + \frac{(x-a)^n}{n!} f^{(n)}(a) + R_{n+1}(x)$$

where

$$R_{n+1}(x) = \frac{(x-a)^{n+1}}{(n+1)!} f^{(n+1)}(\xi(x)) = \int_a^x \frac{(x-t)^n}{n!} f^{(n+1)}(t)\,dt$$

and $\xi(x)$ is somewhere between a and x.

Definition

$$f(x) = \mathcal{O}(g(x)) \quad \text{as} \quad x \to a$$

if and only there exist positive numbers δ and C such that

$$|f(x)| \leq C|g(x)| \quad \text{for} \quad |x - a| < \delta$$

Common use:

$$f(x) = \mathcal{O}((x-a)^p)$$

for some fixed point a and some integer p.

Floating point model

Computers have finite memory, hence not every number can represented exactly on a computer.
Examples: $\sqrt{2}$, π have a infinite number of digits.
To fit in a computer real numbers are approximated by the *floating point model*:

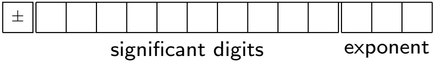
- Binary system is used: If $r \in \mathbb{R}$ then

$$
\begin{aligned}
r &= \pm(\alpha_k 2^k + \alpha_{k-1} 2^{k-1} + \alpha_1 2 + \alpha_0 + \alpha_{-1} 2^{-1} + \cdots) \\
&= \pm(0.\alpha_k\, \alpha_{k-1}\, \ldots\, \alpha_1\, \alpha_0\, \alpha_{-1}\, \ldots) \cdot 2^k
\end{aligned}
$$

  where $\alpha_i \in \{0, 1\}$, $\alpha_k \neq 0$.

- in a computer, a fixed amount of memory is allocated to represent a number:

$$
fl(r) = \pm(0.\alpha_k\, \alpha_{k-1}\, \ldots\, \alpha_{k-m-1}\, \tilde{\alpha}_{k-m}) \cdot 2^e
$$



Floating point numbers

Double precision IEEE 745 (Matlab)

1 bit	52 bits	11 bits
	significant digits	exponent

- **Rounding:** $r \to fl(r)$.
- **Machine accuracy:** The smallest number ϵ such that

$$fl(1+\epsilon) \neq 1$$

- **Relative rounding error** δ_r:

$$fl(r) = r\,(1+\delta_r), \qquad |\delta_r| \leq \varepsilon.$$

- **Loss of significant digits:** loss of precision due subtraction of two almost equal numbers.