# Modal Analysis
with

Saullo G. P. Castro

S.G.P.Castro@tudelft.nl

What is modal analysis? (don`t trust the Wikipedia!)

- Modal analysis is the process of determining the inherent dynamic characteristics of a system in forms of natural frequencies, damping factors and mode shapes, and using them to formulate a mathematical model for its dynamic behavior. The formulated mathematical model is referred to as the modal model of the system and the information for the characteristics are known as its modal data.

- The dynamics of a structure are physically decomposed by frequency and position.

Modal Analysis

He & Fu

Saullo G. P. Castro
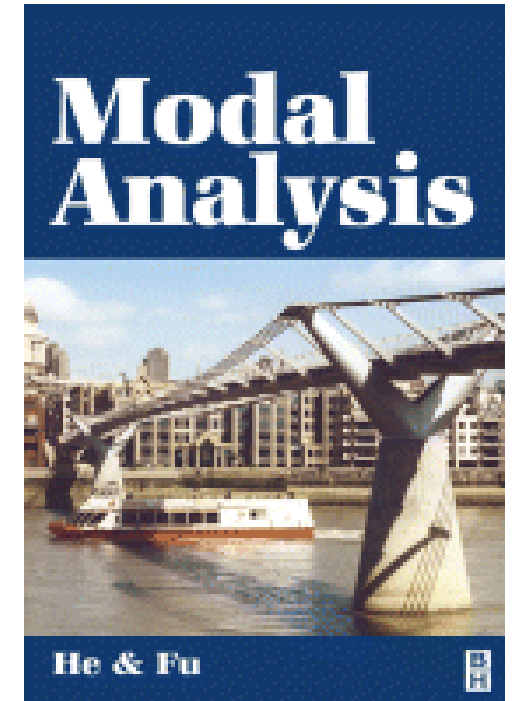
Why Python?

- Most popular and fastest growing programming language

- Highly wanted in industry (employability)

- It is very nice to program in Python

- Highly portable (Linux, Windows, Android etc)

- Scripts can be slightly changed to perform with C-level speed

- Compared to Matlab…

Formulation of a Structural Dynamics Problem

- ## Wave propagation
  - Interest in short-time responses
  - Propagation of disturbances along the structure

Modal Analysis
with
python™

Saullo G. P. Castro

C O B E M
UBERLANDIA
2 0 1 9

TUDelft
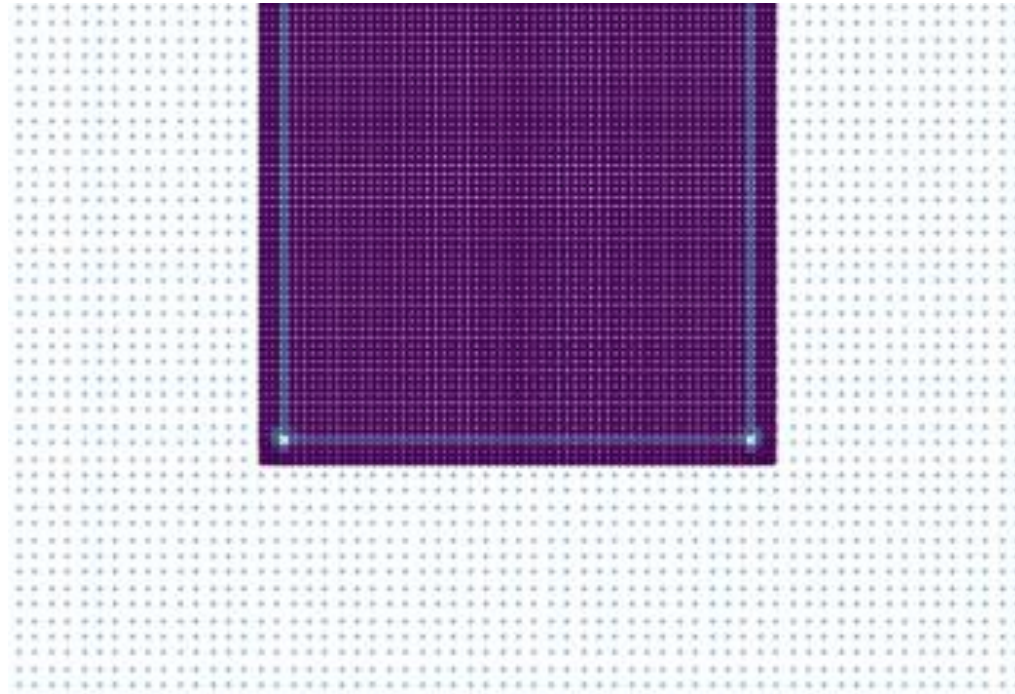
Formulation of a Structural Dynamics Problem

- ## Vibrations
  - Interest in long-time responses (compared to time required for waves to traverse the structure)
  - Structural oscillations in a more global sense

Modal Analysis
with
python™
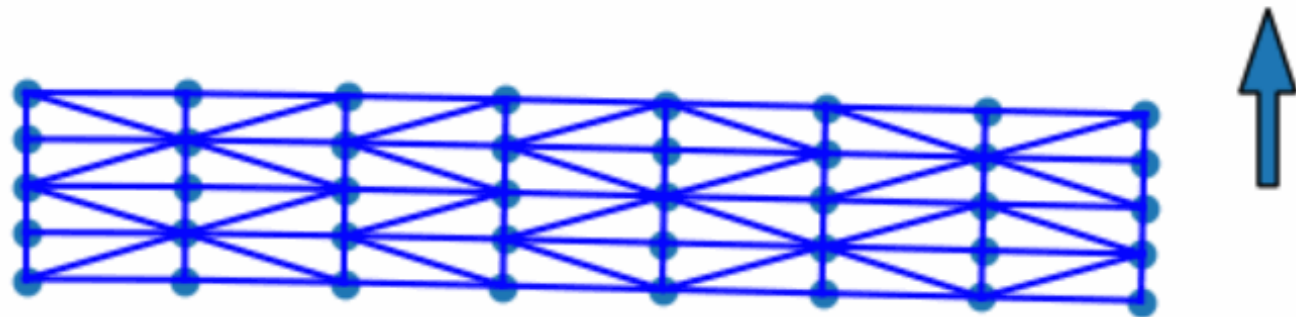
Saullo G. P. Castro

Downloading and citing the material from this course

Saullo G. P. Castro. "Modal Analysis with Python". COBEM 2019, Uberlândia, Brazil. DOI:10.5281/zenodo.3514373.

TUDelft

Saullo G. P. Castro

Course Program

- ## Days 1 and 2: SDOF (single degree-of-freedom) systems

  – Solutions for free and forced vibration, undamped and damped

  – Harmonic and general loads

- ## Days 3: MDOF (multiple degree-of-freedom) systems

  – Generalized eigenvalue problem

  – Symmetric eigenvalue problem

  – Solution for free vibration

  – Frequency domain, experimental estimation of natural frequencies

- ## Days 4 and 5: MDOF systems – solutions

  – Discretization of continuous systems using finite elements

  – Consistent vs. lumped mass matrix

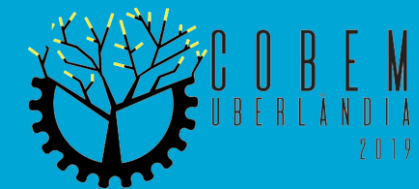# Part 1 – SDOF (single degree-of-freedom) systems

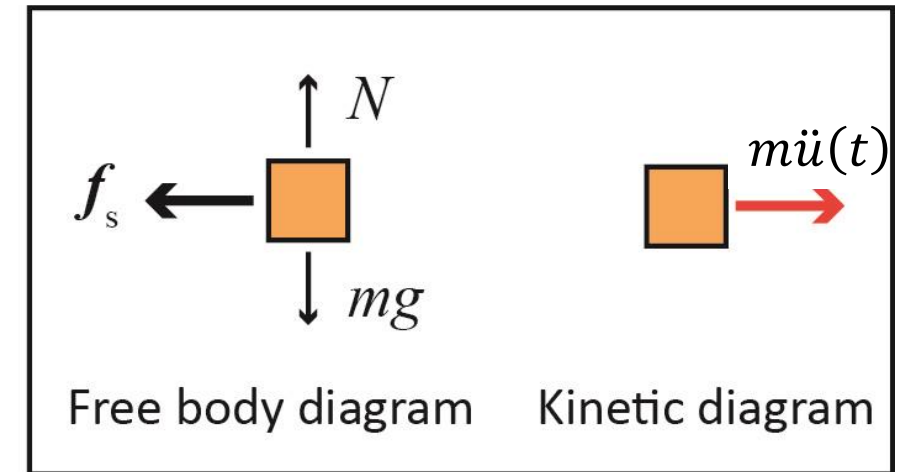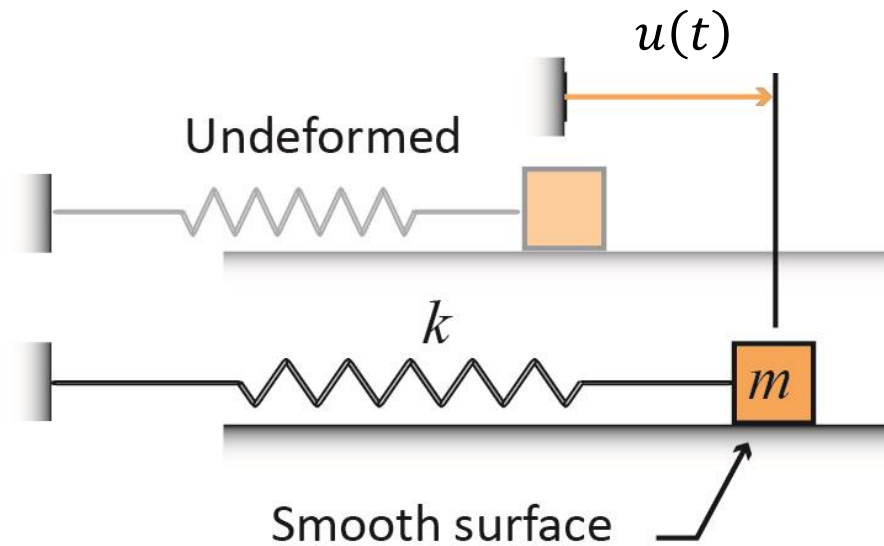Solutions for free/forced vibration of SDOF systems, undamped/damped, harmonic/general loading

Review on vibration of SDOF systems

Free vibration of undamped SDOF

Undeformed

$u(t)$

k

m

Smooth surface

N

f_s

mg

$m\ddot{u}(t)$

Free body diagram   Kinetic diagram

Equation of Motion:

$$k\,u(t) + m\ddot{u}(t) = 0$$

Free vibration of undamped SDOF

$$k\,u(t) + m\ddot{u}(t) = 0$$

Exercise 1: Get general solution

Exercise 1 part b: Plot a particular solution
- $u(0) = 0.4$
- $\dot{u}(0) = 2$
- $k = 150$
- $m = 2$
- $0 \le t \le 1$

Using SymPy's plot

Free vibration of undamped SDOF

$$k\,u(t) + m\ddot{u}(t) = 0$$

General solution

$$u(t) = C_1 \sin(\omega_n t) + C_2 \cos(\omega_n t)$$

$$\omega_n = \sqrt{\frac{k}{m}}$$

Important Question: units of ω_n?
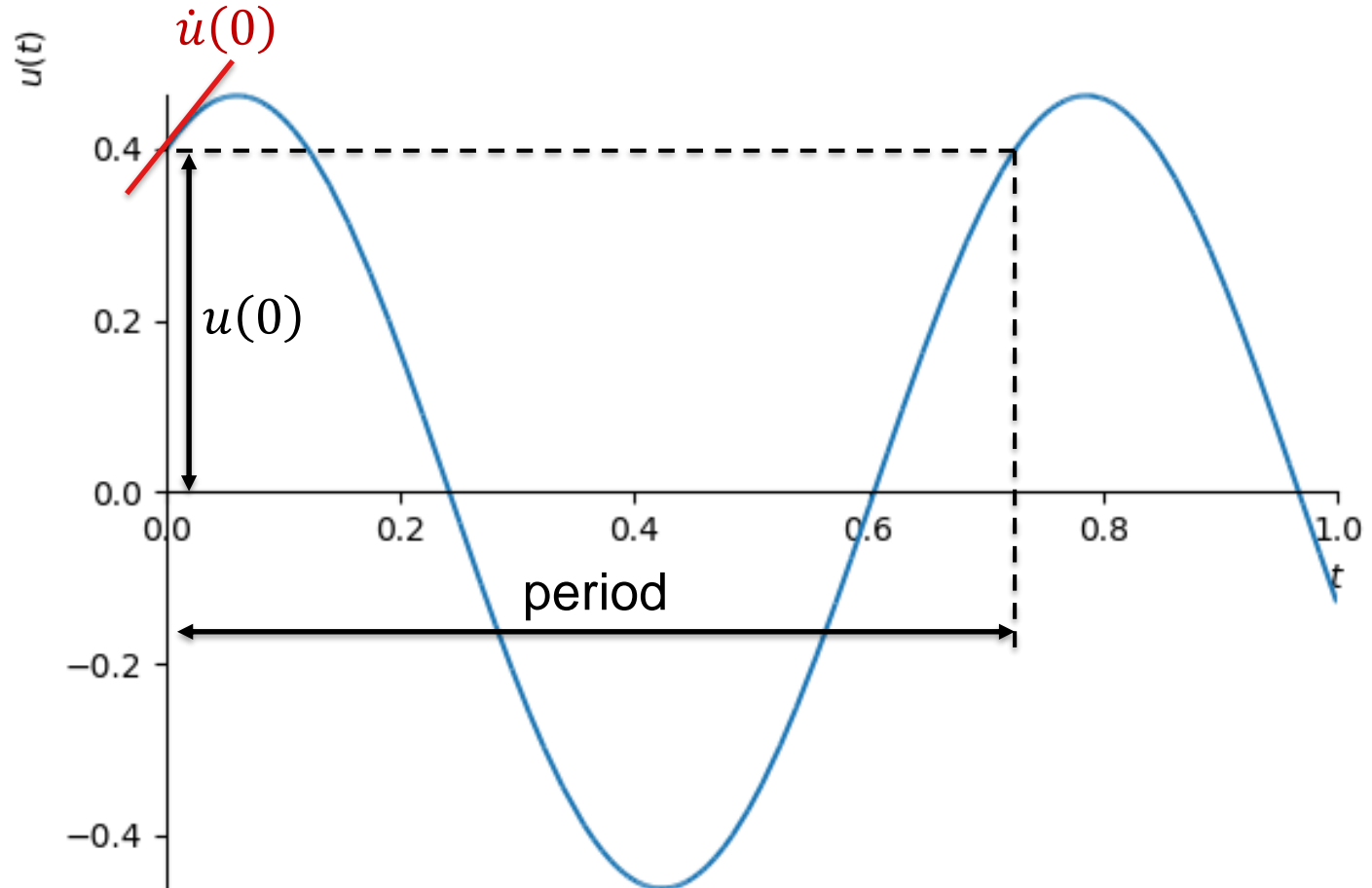
Free vibration of undamped SDOF

$$k\,u(t) + m\ddot{u}(t) = 0$$

Another way to write using $\omega_n = \sqrt{k/m}$

$$\omega_n^2 u(t) + \ddot{u}(t) = 0$$

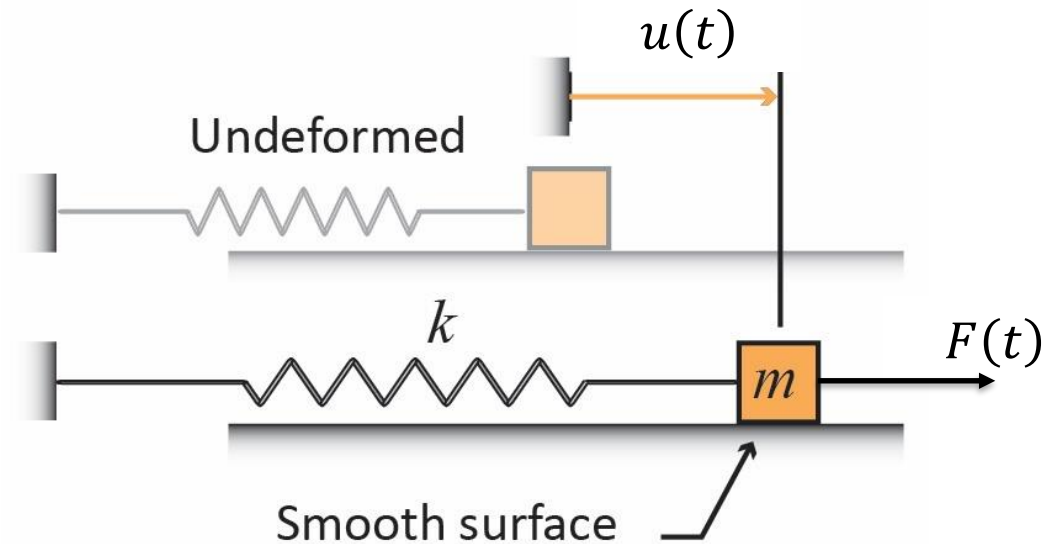**Mass normalized eq. of motion!**

Free vibration of undamped SDOF

Plot

$$period = \frac{2\pi}{\omega_n} \text{ (converting from } s/rad \text{ to } s)$$

Forced vibration of undamped SDOF

Equation of Motion:

$$k\,u(t) + m\ddot{u}(t) = F(t)$$

Forced vibration of undamped SDOF

Equation of Motion:

$$k\,u(t) + m\ddot{u}(t) = F(t)$$

Harmonic force:

$$F(t) = F_0 \sin\!\left(\omega_f t\right)$$

Exercise 2: Get general solution

Forced vibration of undamped SDOF

Equation of Motion:

$$k\, u(t) + m\ddot{u}(t) = F(t)$$

Harmonic force:

$$F(t) = F_0 \sin(\omega_f t)$$

General solution:

$$u(t) = \boxed{\frac{F_0}{m}} \frac{\sin(\omega_f t)}{\omega_n^2 - \omega_f^2} + C_1 \sin(\omega_n t) + C_2 \cos(\omega_n t)$$

Forced vibration of undamped SDOF

Mass-normalized Equation of Motion:

$$\omega_n^2 u(t) + \ddot{u}(t) = f(t)$$

Mass-normalized harmonic force:

$$f(t) = f_0 \sin(\omega_f t)$$
$$f_0 = F_0/m$$

General solution, all mass-normalized:

$$u(t) = f_0 \frac{\sin(\omega_f t)}{\omega_n^2 - \omega_f^2} + c_1 \sin(\omega_n t) + c_2 \cos(\omega_n t)$$

Forced vibration of undamped SDOF

General solution:

$$u(t) = \boxed{\frac{F_0}{m}} \frac{\sin(\omega_f t)}{\omega_n^2 - \omega_f{}^2} + C_1 \sin(\omega_n t) + C_2 \cos(\omega_n t)$$
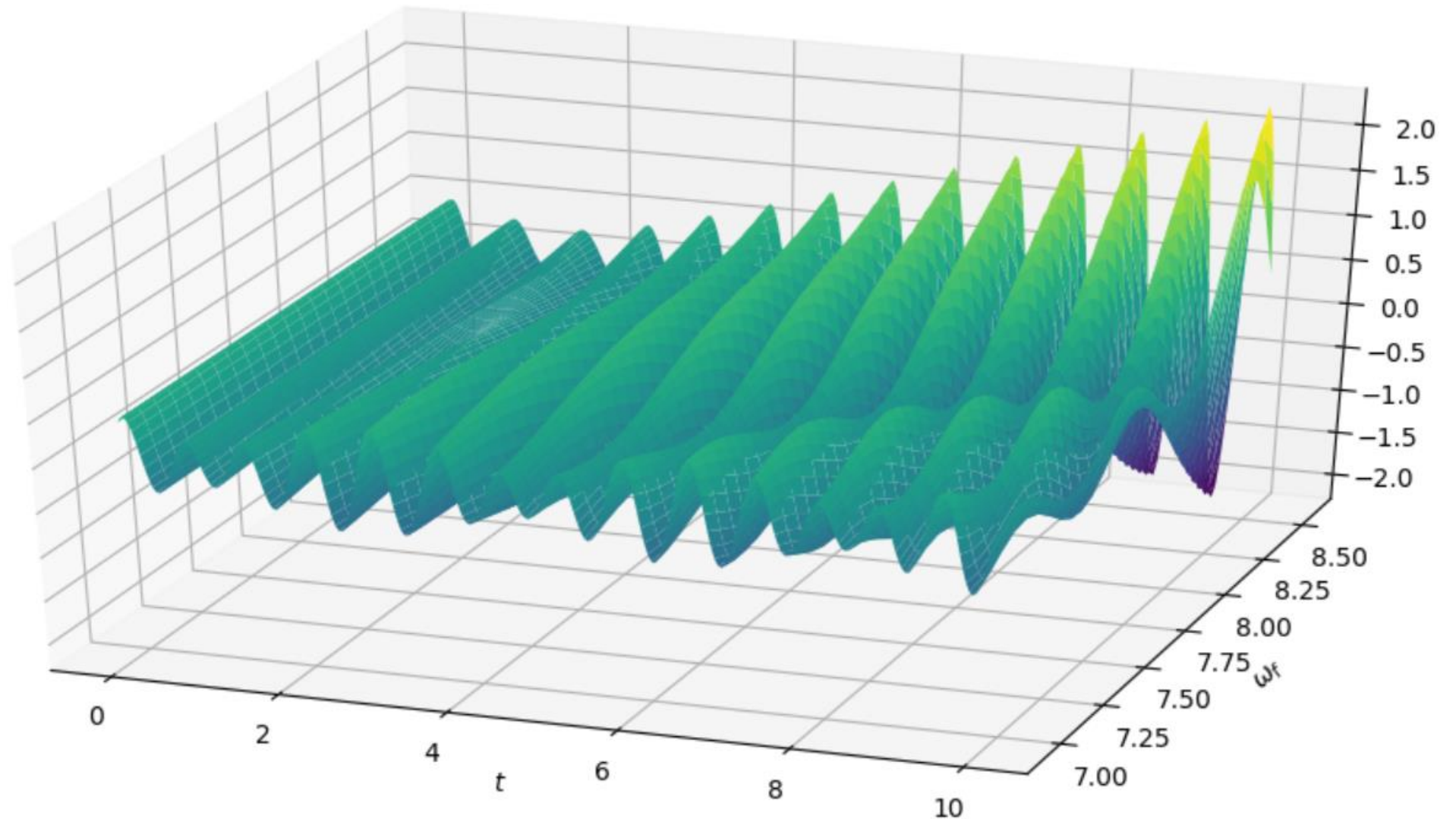
Exercise 2 part b: Plot a particular solution

- $u(0) = 0.4$
- $\dot{u}(0) = 2$
- $k = 150$
- $m = 2$
- $F_0 = 10$
- $0.8\omega_n \leq \omega_f \leq 0.99\omega_n$ (run 1 modal analysis for each)
- $0 < t < 10$

Using SymPy's plot3d
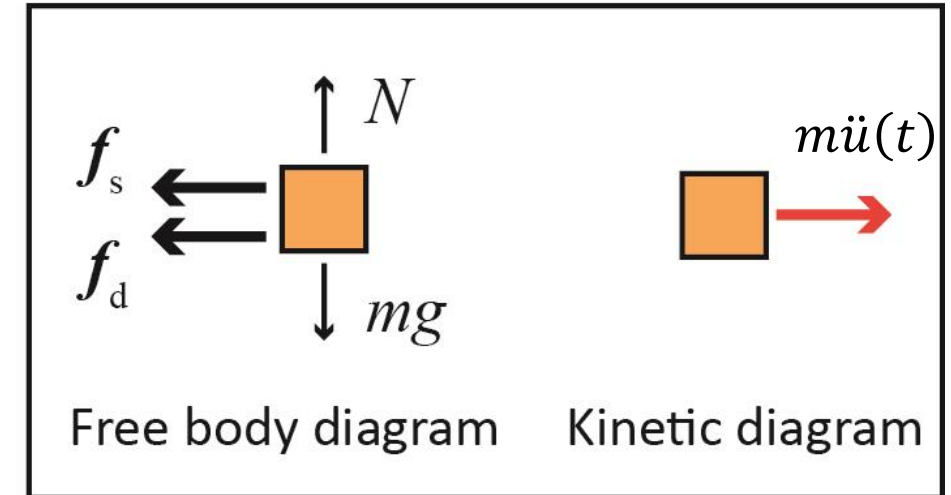
Modal Analysis
with

python™

Saullo G. P. Castro

C O B E M
UBERLANDIA
2019

TUDelft
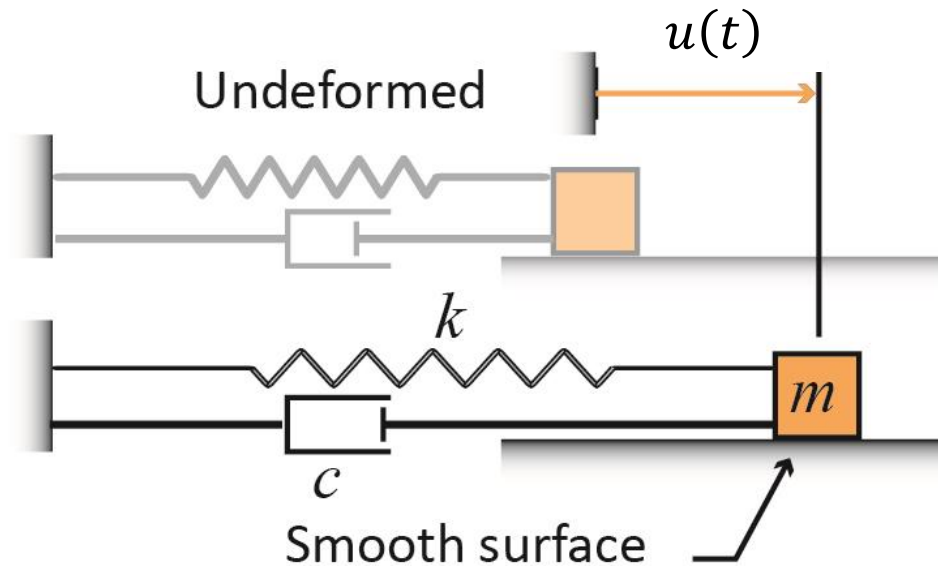
Forced vibration of undamped SDOF

Plot

Free vibration of damped SDOF

Equation of Motion:

$$k\, u(t) + c\dot{u}(t) + m\ddot{u}(t) = 0$$

Free vibration of damped SDOF

Equation of Motion:

$$k\,u(t) + c\dot{u}(t) + m\ddot{u}(t) = 0$$

Example 3: Solve using SymPy

Free vibration of damped SDOF

Equation of Motion:

$$k\,u(t) + c\dot{u}(t) + m\ddot{u}(t) = 0$$

Solution

$$u(t) = C_1 e^{\frac{t\left(-c-\sqrt{c^2-4km}\right)}{2m}} + C_2 e^{\frac{t\left(-c+\sqrt{c^2-4km}\right)}{2m}}$$

What are the conditions to become oscillatory?

Tip:

$$\cos(\theta) = \frac{1}{2}\left(e^{+i\theta} + e^{-i\theta}\right)$$

$$\sin(\theta) = \frac{1}{2i}\left(e^{+i\theta} - e^{-i\theta}\right)$$

Free vibration of damped SDOF

Equation of Motion:

$$k\,u(t) + c\dot{u}(t) + m\ddot{u}(t) = 0$$

Solution

$$u(t) = C_1 e^{\frac{t\left(-c-\sqrt{c^2-4km}\right)}{2m}} + C_2 e^{\frac{t\left(-c+\sqrt{c^2-4km}\right)}{2m}}$$

Critical damping:

$$c_{cr} = 2\sqrt{km}$$

In structures
without dashpots
ζ is usually < 0.05

Damping ratio:

$$\zeta = \frac{c}{c_{cr}}$$

Free vibration of damped SDOF

Equation of Motion in terms of damping ratio:

$$k\,u(t) + 2\zeta\sqrt{km}\,\dot{u}(t) + m\ddot{u}(t) = 0$$

Mass-normalizing:

$$\omega_n^2\,u(t) + 2\zeta\sqrt{k}\,\frac{\sqrt{m}}{m}\dot{u}(t) + \ddot{u}(t) = 0$$

$$\omega_n^2\,u(t) + 2\zeta\,\omega_n\dot{u}(t) + \ddot{u}(t) = 0$$

Example 4: Solve using SymPy

Free vibration of damped SDOF

$$\omega_n^2\, u(t) + 2\zeta\omega_n \dot{u}(t) + \ddot{u}(t) = 0$$

Solution:

$$u(t) = \left(C_1 e^{-\omega_n t \sqrt{\zeta^2-1}} + C_2 e^{\omega_n t \sqrt{\zeta^2-1}} \right) e^{-\omega_n \zeta t}$$

What are the conditions to become oscillatory?

Tip:

$$\cos(\theta) \quad = \quad \frac{1}{2}\left(e^{+i\theta} + e^{-i\theta}\right)$$

$$\sin(\theta) \quad = \quad \frac{1}{2i}\left(e^{+i\theta} - e^{-i\theta}\right)$$

Free vibration of damped SDOF

$$u(t) = \left(C_1 e^{-\omega_n t \sqrt{\zeta^2 - 1}} + C_2 e^{\omega_n t \sqrt{\zeta^2 - 1}} \right) e^{-\omega_n \zeta t}$$

What are the conditions to become oscillatory?

- Underdamped

$$0 < \zeta < 1$$
$$\sqrt{\zeta^2 - 1} \equiv i\sqrt{1 - \zeta^2}$$

$$u(t) = \left(C_1 e^{-i\,\boxed{\omega_n \sqrt{1 - \zeta^2}}\,t} + C_2 e^{i\,\boxed{\omega_n \sqrt{1 - \zeta^2}}\,t} \right) e^{-\omega_n \zeta t}$$

Damped natural frequency
$$\omega_d = \omega_n \sqrt{1 - \zeta^2}$$

27

Saullo G. P. Castro

Free vibration of damped SDOF

Underdamped solution with $u(0) = u_0, \dot{u}(0) = v_0$:

$$u(t) = A_0 e^{-\zeta \omega_n t} \sin(\omega_d t + \varphi)$$

$$A_0 = \sqrt{u_0^2 + \left(\zeta \frac{\omega_n}{\omega_d} u_0 + \frac{v_0}{\omega_d}\right)^2}$$

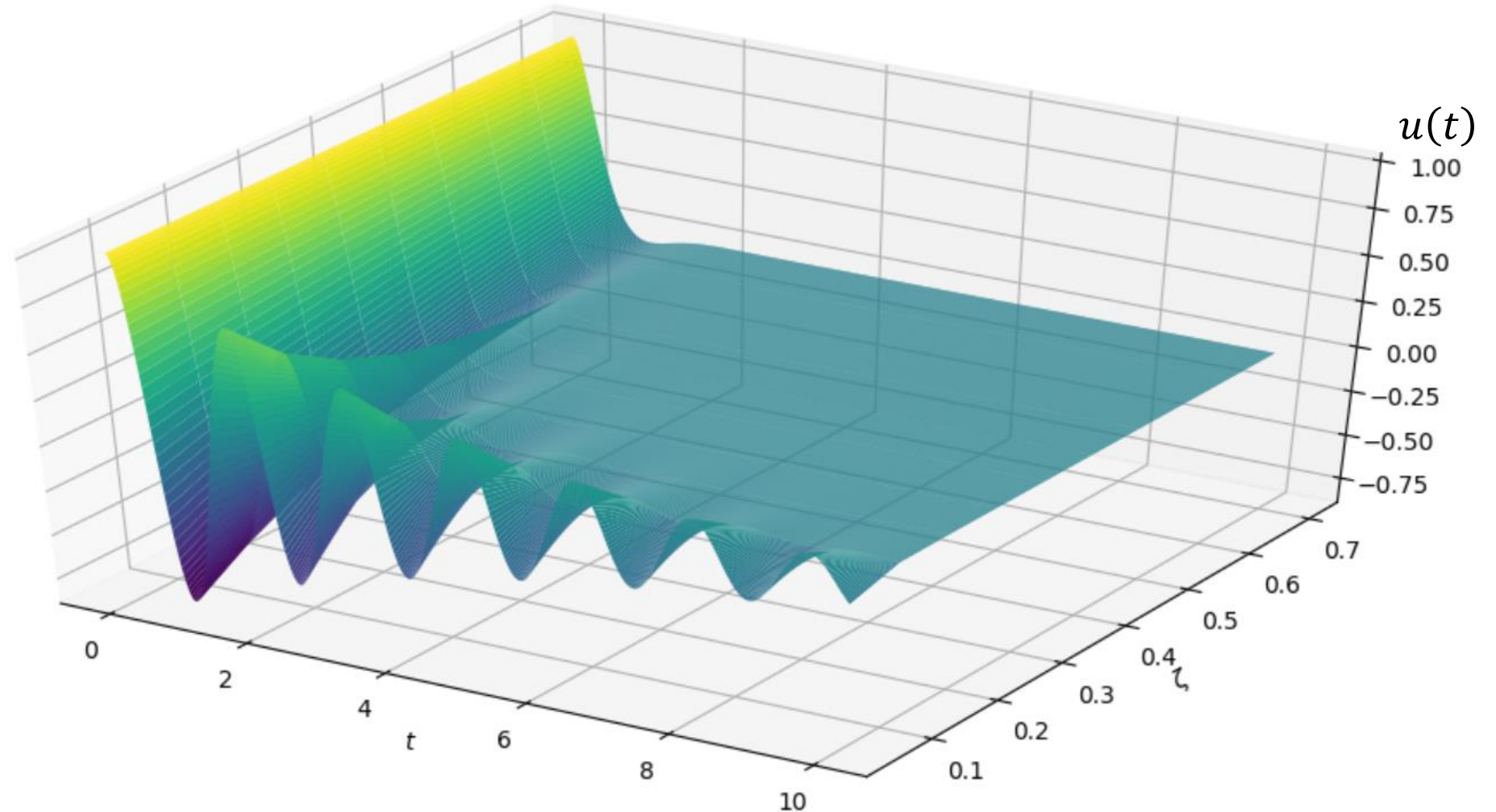$$\varphi = \arctan \frac{\omega_d u_0}{\zeta \omega_n u_0 + v_0}$$

Tedious
algebra alert!

Damped natural frequency
$$\omega_d = \omega_n \sqrt{1 - \zeta^2}$$

C O B E M
U B E R L A N D I A
2 0 1 9

TUDelft

28

Free vibration of damped SDOF

Plotting with $\omega_n = 4$, $u_0 = 1$, $v_0 = 0$, $0.05 \leq \zeta \leq 0.7$, $0 \leq t \leq 10$:
Script: `exercise04_underdamped_free_plot.py`
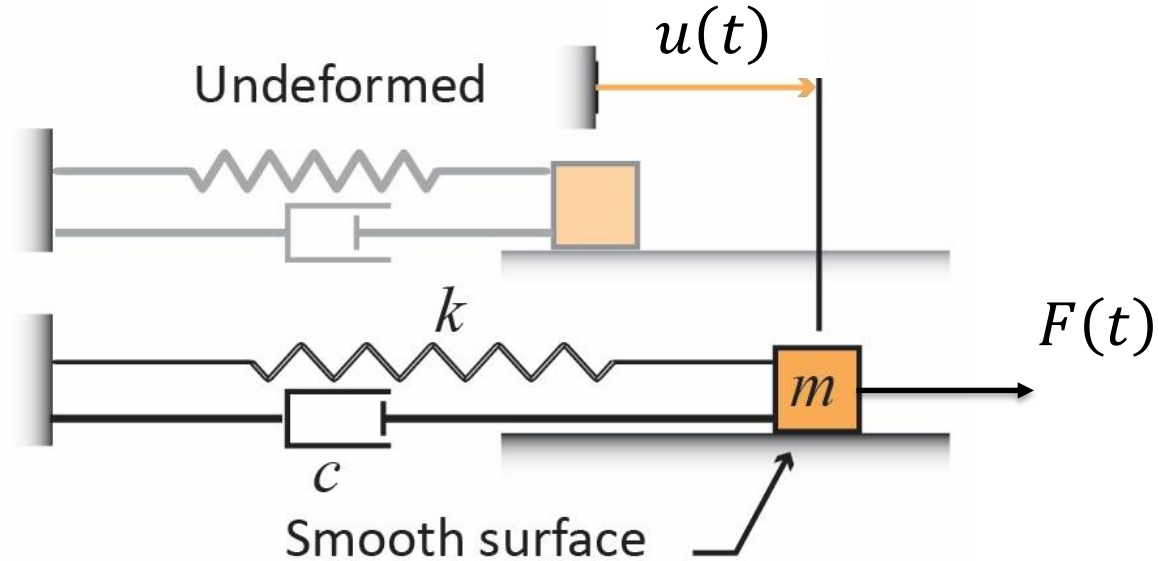
Free vibration of damped SDOF

$$\omega_n^2\, u(t) + 2\zeta\,\omega_n \dot{u}(t) + \ddot{u}(t) = 0$$

Homework for you:

- Critically damped ($\zeta = 1$)

- Overdamped ($\zeta > 1$)

NOTE: These will not be used in the rest of the course

Saullo G. P. Castro

Forced vibration of damped SDOF

$u(t)$

Undeformed

$F(t)$

k

m

c

Smooth surface

Equation of Motion:

$$k\,u(t) + c\dot{u}(t) + m\ddot{u}(t) = F(t)$$

Saullo G. P. Castro

Forced vibration of damped SDOF

Equation of Motion:

$$k\,u(t) + c\dot{u}(t) + m\ddot{u}(t) = F(t)$$

Mass-normalized Equation of Motion:

$$\omega_n^2 u(t) + 2\zeta\omega_n\dot{u}(t) + \ddot{u}(t) = f(t)$$

Solution on next slide

Forced vibration of damped SDOF

Assuming:

- harmonic load: $f(t) = f_0 \cos(\omega_f t)$
- $u(0) = u_0, \dot{u}(0) = v_0$

Solution for underdamped case $(0 < \zeta < 1)$:

$$u(t) = u_h(t) + u_p(t)$$

Tedious
algebra alert!

$$u_h(t) = A_0 e^{-\zeta \omega_n t} \sin(\omega_d t + \varphi)$$
$$u_p(t) = |u_p| \cos(\omega_f t - \theta)$$

$$|u_p| = \frac{f_0}{\sqrt{(\omega_n^2 - \omega_f^2) + (2\zeta\omega_n\omega_f)^2}}$$
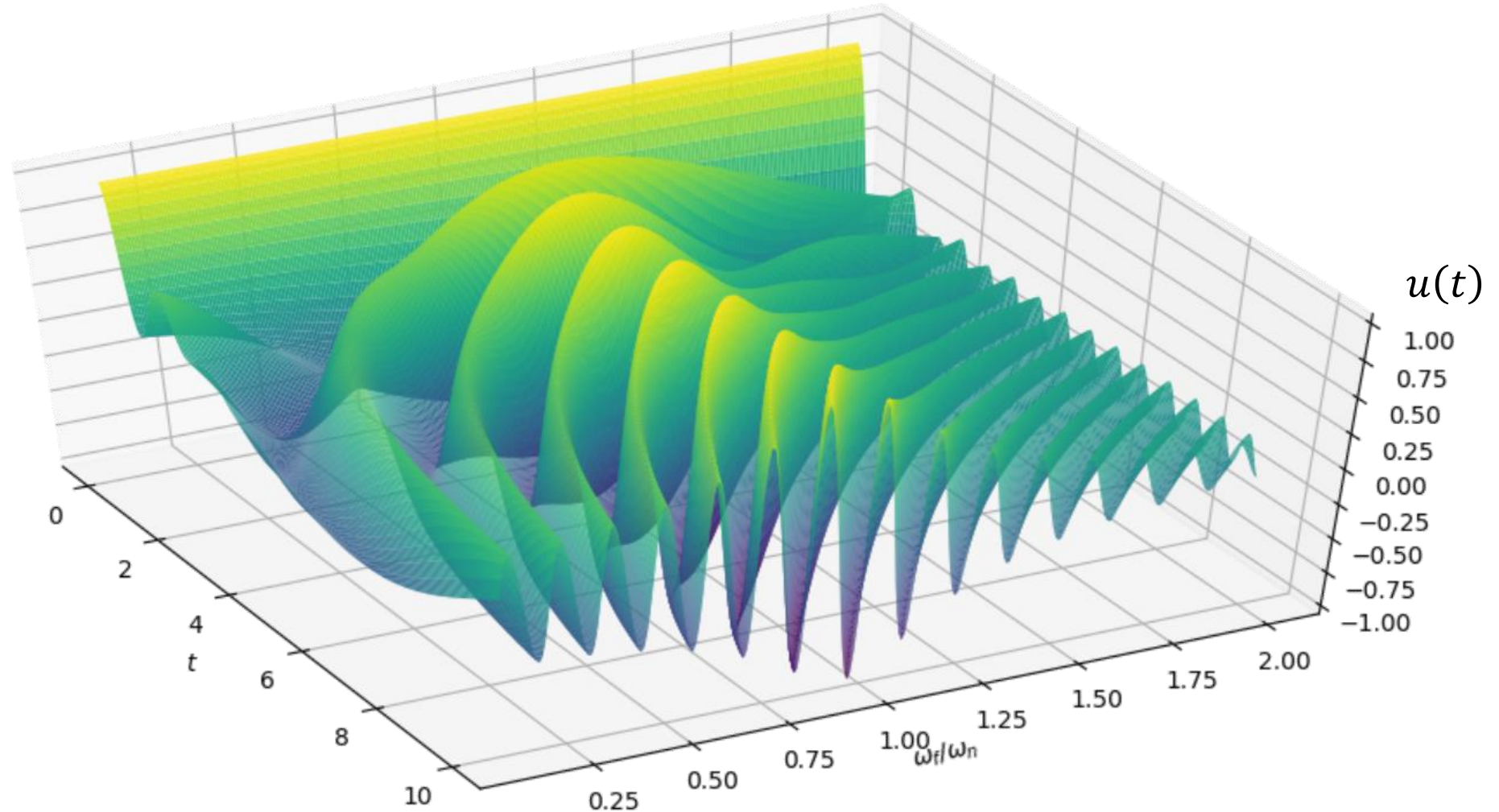
$$A_0 = \frac{u_0 - |u_p| \cos\theta}{\sin\varphi}$$

$$\varphi = \arctan \frac{\omega_d(u_0 - |u_p|\cos\theta)}{v_0 + (u_0 - |u_p|\cos\theta)\zeta\omega_n - \omega_f|u_p|\sin\theta}$$

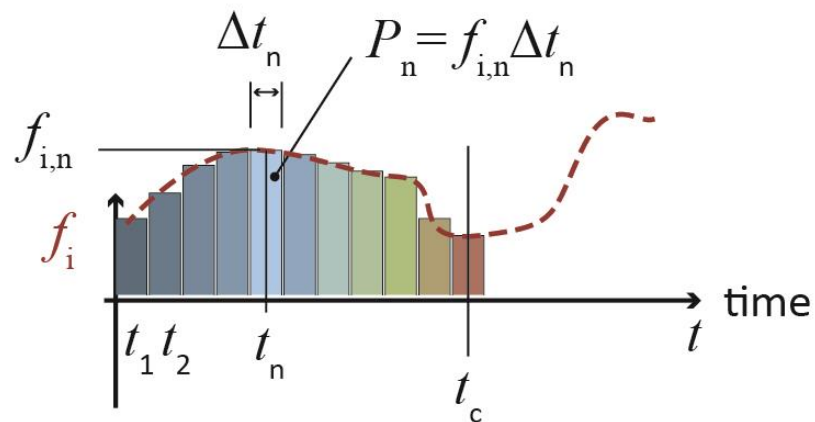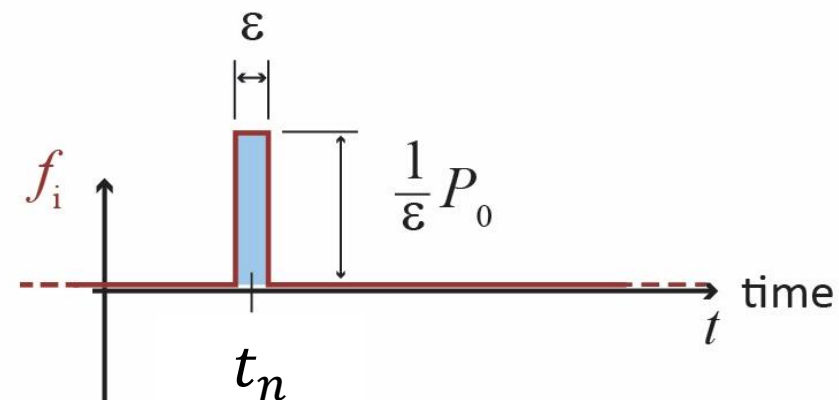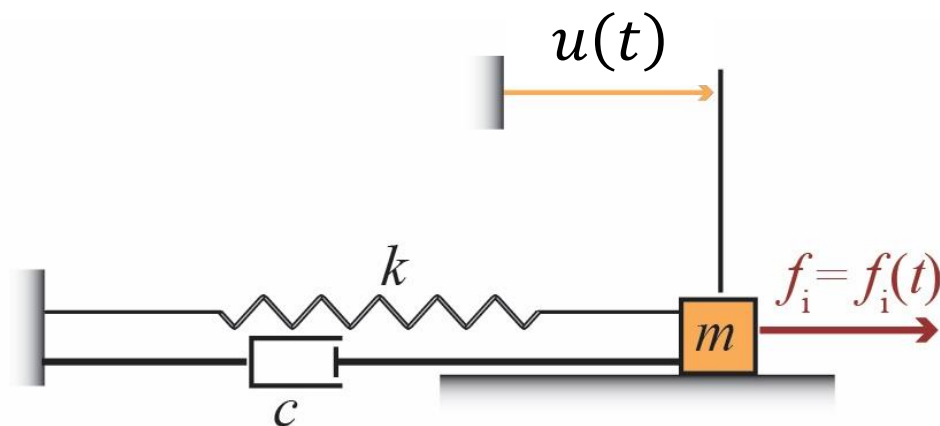$$\theta = \arctan \frac{2\zeta\omega_n\omega_f}{\omega_n^2 - \omega_f^2}$$

TUDelft

Forced vibration of damped SDOF
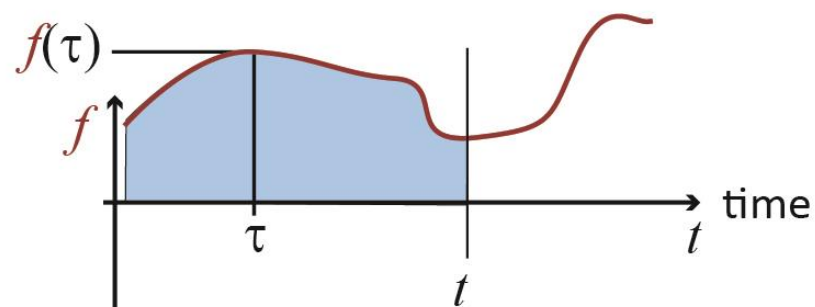
Plotting (script `exercise05_harmonic_f_damped_plot.py`):

- $\omega_n = 5, u_0 = 1, v_0 = 0, f_0 = 10, \zeta = 0.2$

Saullo G. P. Castro

Towards general load cases: Impulsive Load

$u(t)$

k

m

$f_i = f_i(t)$

c

ε

f_i

$\dfrac{1}{\varepsilon} P_0$

time

t

t_n

Δt_n

$P_n = f_{i,n} \Delta t_n$

$f_{i,n}$

f_i

time

$t_1 \, t_2$

t_n

t_c

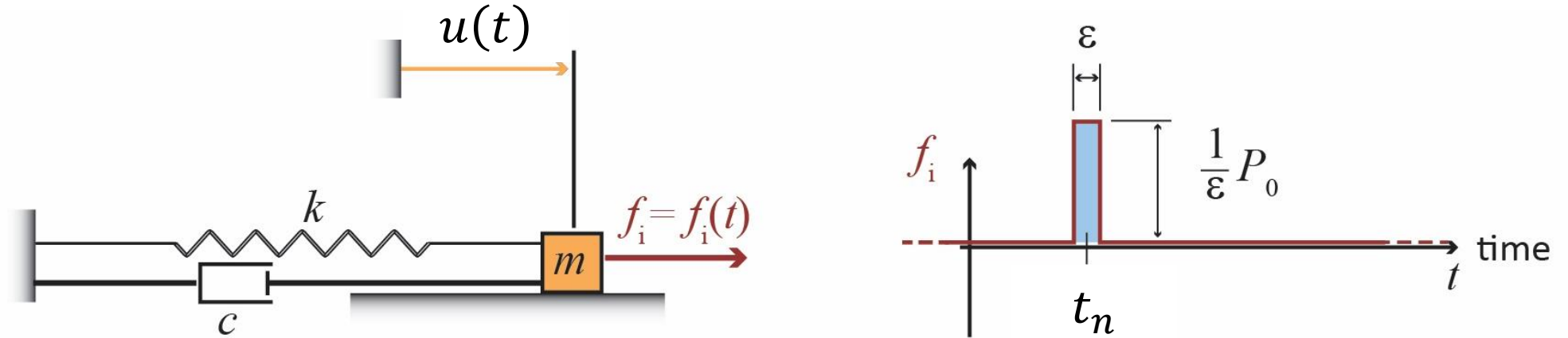t

Discrete Impulsive loadings

$f(\tau)$

f

time

τ

t

t

Continuous loading

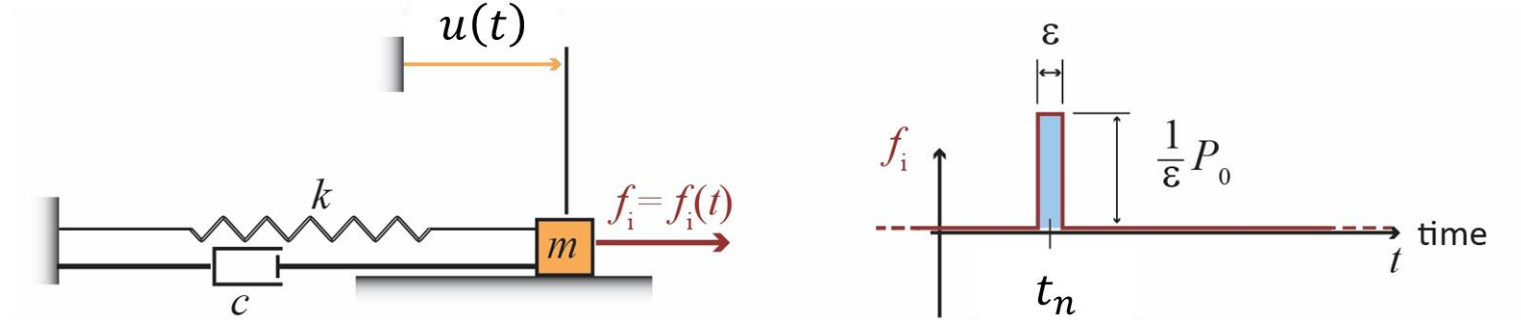# SDOF, Damped Response for an Impulsive Load

$$u(t) = \begin{cases} 0 & t < t_n \\ P_0 h(t - t_n) & t \geq t_n \end{cases}$$

$h(t - t_n) \equiv$ **unit impulse response function**

$$h(t - t_n) = \frac{1}{m\omega_d} e^{-\zeta\omega_n(t-t_n)} \sin\omega_d(t - t_n); t > t_n$$

SDOF, Response for an Impulsive Load

$u(t)$

k

c

$f_i = f_i(t)$

m

ε

f_i

$\frac{1}{\varepsilon} P_0$
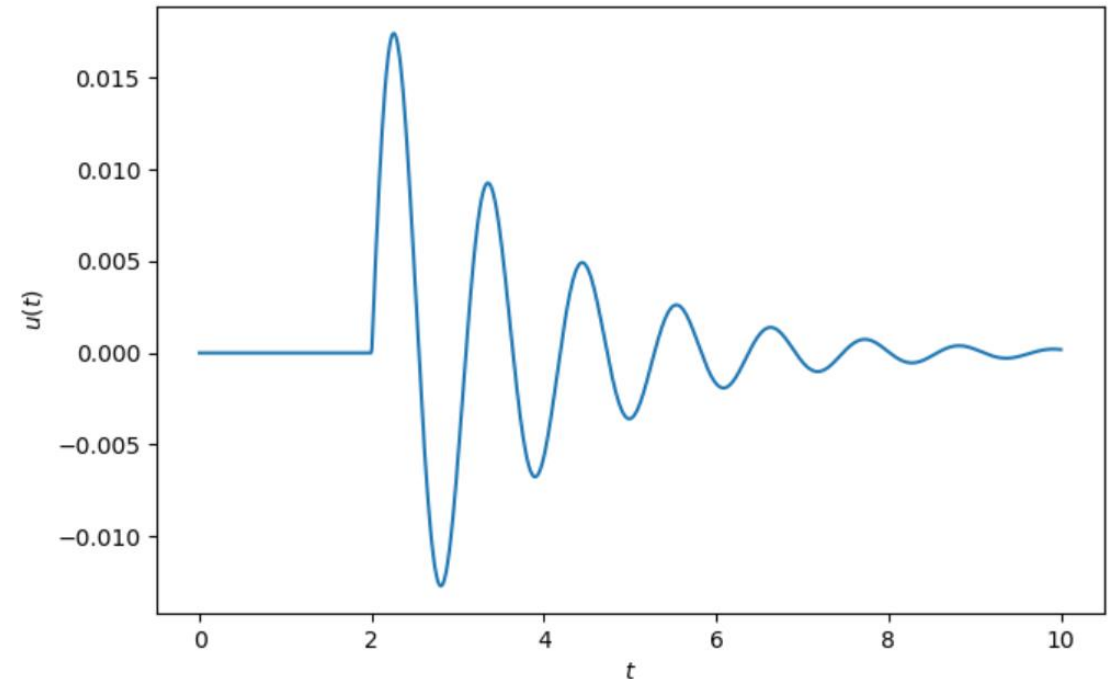
t_n

time

t

Challenge for you:

exercise06_hard_coded_solution.py

- Find problem with scripts, probably related to the Piecewise function in SymPy

```
exercise06_f_impulse_FIXME.py
exercise06_f_impulse_plot_FIXME.py
```

- I tried for a couple of hours without success

Saullo G. P. Castro

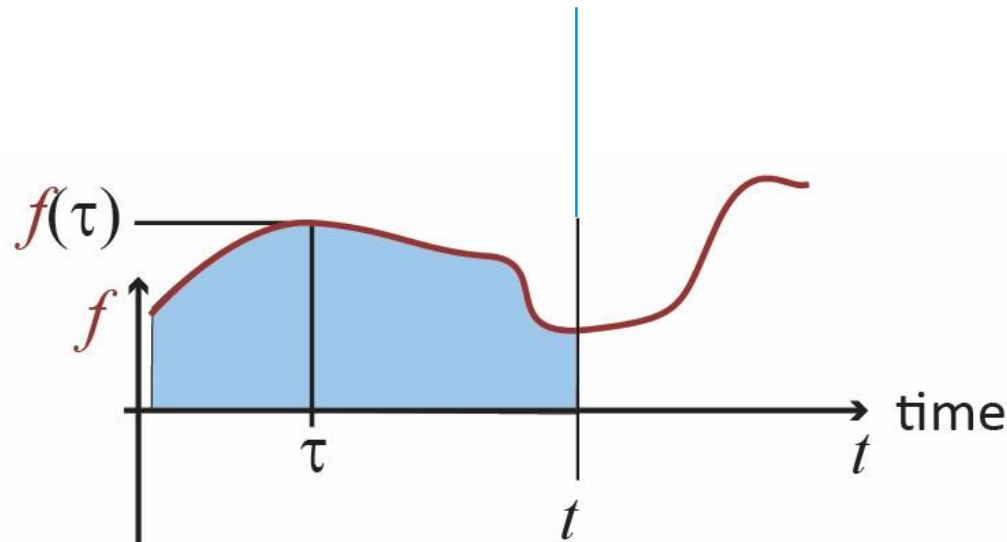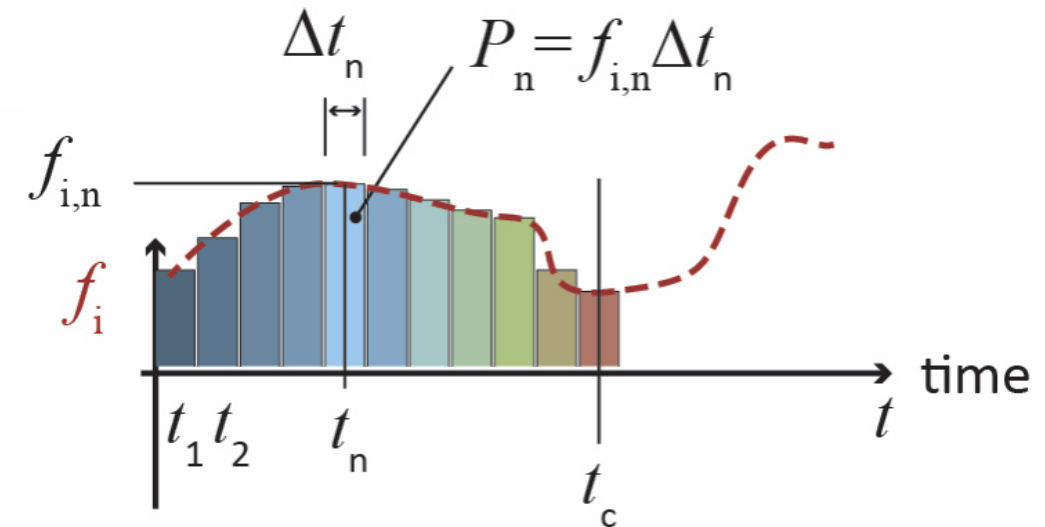# SDOF, Response to General Loads

Duhamel integral:

$$u(t) = \int_{-\infty}^{t} f(\tau)h(t - \tau)d\tau$$

Approximation:

$$u(t_c) = \sum_{n=1}^{c} f_{i,n}h(t_c - t_n)\,\Delta t_n$$

Continuous loading

Discrete Impulsive loadings

MDOF Systems

Generalized eigenvalue problem
Orthonormal, from nodal to modal representation
Symmetric eigenvalue problem
Free/forced, undamped/damped vibration

Free vibration of undamped MDOF

General formulation: for a system with N degrees of freedom:

$$M\ddot{u} + Ku = 0$$

DoF vector

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}$$

(Global) mass matrix

$$M = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1N} \\ m_{21} & m_{22} & \cdots & m_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ m_{N1} & m_{N2} & \cdots & m_{NN} \end{bmatrix}$$

Symmetric and positive definite

Null vector

$$0 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

(Global) stiffness matrix

$$K = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1N} \\ k_{21} & k_{22} & \cdots & k_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ k_{N1} & k_{N2} & \cdots & k_{NN} \end{bmatrix}$$

$$K = K^{\top}$$
$$k_{ij} = k_{ji}$$

Free vibration of undamped MDOF

Special case: **lumped** mass matrix

In some cases it is possible to work with a **diagonal** matrix that may be an **approximation** to the (consistent) mass matrix

(Global) mass matrix

$$
M = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1N} \\ m_{21} & m_{22} & \cdots & m_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ m_{N1} & m_{N2} & \cdots & m_{NN} \end{bmatrix}
$$

(Lumped) mass matrix

$$
M_l = \begin{bmatrix} m_1 & 0 & \cdots & 0 \\ 0 & m_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & m_N \end{bmatrix}
$$

TUDelft

Modal Analysis
with

Saullo G. P. Castro

Direct solution to free vibrations of undamped MDOFs systems: The generalized eigenvalue approach

Separation of Variables

$$\boldsymbol{M\ddot{u}} + \boldsymbol{Ku} = \boldsymbol{0}$$

Assume a general solution of the form

$$\boldsymbol{u}(x, t) = \boldsymbol{U}(x)e^{\lambda t} \qquad \boldsymbol{U} = \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_N \end{bmatrix}$$

What does this assumption imply?

Synchronous motion for all degrees of freedom U_i

Separation of Variables

$$M\ddot{u} + Ku = 0$$

Assume a general solution of the form

$$u(x,t) = U(x)T(t) \qquad U = \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_N \end{bmatrix} \qquad T(t) \text{ is a scalar function}$$

Calculating the derivatives:

First derivative

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

Second derivative

$$\ddot{u}(x,t) = U(x)\ddot{T}(t)$$

Separation of Variables

$$M\ddot{u}(x,t) + Ku(x,t) = 0$$

$$\left(M\ddot{T}(t) + KT(t)\right)U(x) = 0$$

Gives N systems of equations, with $I = 1,2 \dots N$

$$-\frac{\ddot{T}(t)}{T(t)} = \frac{\sum_{J=1}^{N} k_{IJ}U_J(x)}{\sum_{J=1}^{N} m_{IJ}U_J(x)} = constant = \omega_n^{(I)2}$$

$$\ddot{T}(t) + \omega_n^{(I)2}T(t) = 0$$

$$\ddot{T}(t) = -\omega_n^{(I)2}T(t)$$

COBEM
UBERLANDIA
2019

TUDelft

Generalized Eigenvalue Problem

$$\left(-\omega_n^{(I)^2} \boldsymbol{M} + \boldsymbol{K}\right) \boldsymbol{U}^{(I)} T(t) = \boldsymbol{0}$$

$$\underbrace{\left(-\omega_n^{(I)^2} \boldsymbol{M} + \boldsymbol{K}\right)}_{\text{Singular}} \boldsymbol{U}^{(I)} = \boldsymbol{0}$$

$$\det\left(-\omega_n^{(I)^2} \boldsymbol{M} + \boldsymbol{K}\right) = 0$$

NOTE: $\boldsymbol{U}^{(I)}$ represents a mode shape corresponding to $\omega_n^{(I)}$ and it is not $\boldsymbol{U}(x)$ in the general solution $\boldsymbol{u}(x,t) = \boldsymbol{U}(x)T(t)$

Example of Generalized Eigenvalue Problem
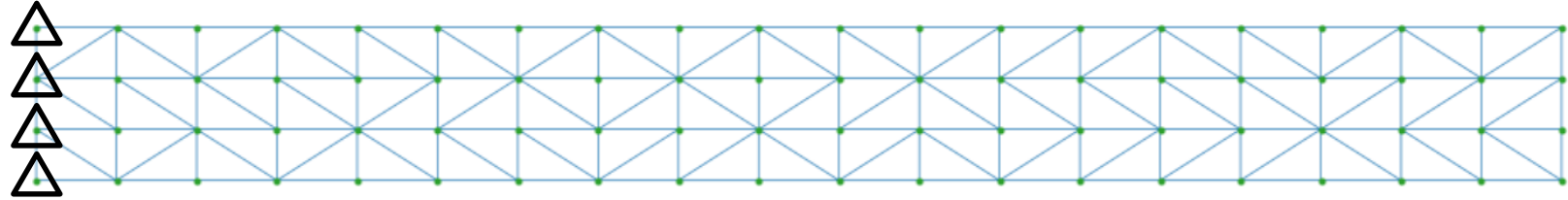
- $\rho = 2.6e3$
- $E = 70e9$
- $A = 0.01^2$
- length= 10
- width=1

- Solve a generalized eigenvalue problem
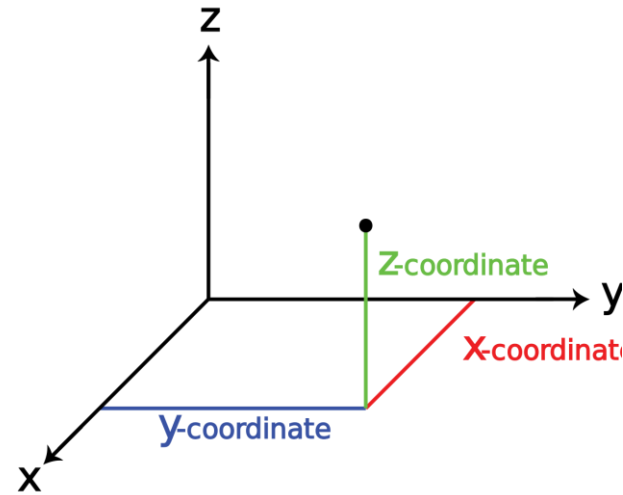- Find natural frequencies
- Plot first 5 mode shapes

Script `exercise07_generalized_eigenvalue_problem.py`

compare lumped=True/False (will be explained later)
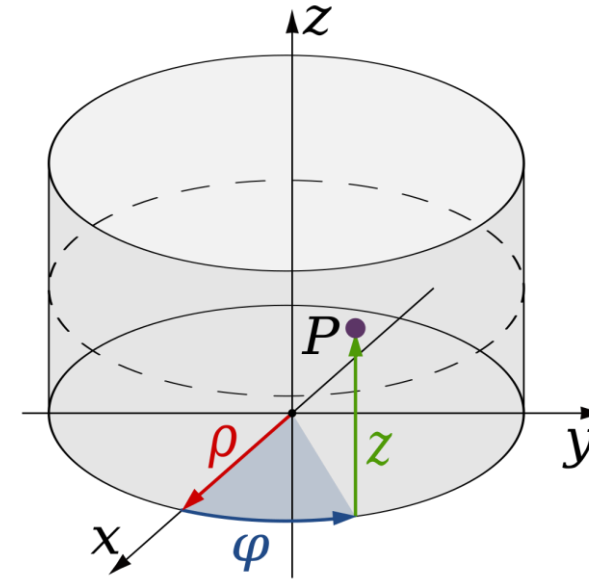
Saullo G. P. Castro

Next Steps…

- Now we know how to find $\omega_n^{(I)}$ and the corresponding mode shapes $\boldsymbol{U}^{(I)}$ for a MDOF system

- Are these modes orthonormal?

NOTE:
Orthonormal is
Orthogonal and Normalized

- Why is this important?

Saullo G. P. Castro

Orthonormal Basis in 3D Space

$$v = v_x\boldsymbol{i} + v_y\boldsymbol{j} + v_z\boldsymbol{k}$$

$$v = v_r\boldsymbol{u_r} + v_\varphi\boldsymbol{u_\varphi} + v_z\boldsymbol{u_z}$$

$$\boldsymbol{i} = \{1, 0, 0\}^T$$
$$\boldsymbol{j} = \{0, 1, 0\}^T$$
$$\boldsymbol{k} = \{0, 0, 1\}^T$$

Orthonormal Basis: $\boldsymbol{i} \cdot \boldsymbol{j} = 0, \boldsymbol{i} \cdot \boldsymbol{k} = 0 \; \boldsymbol{j} \cdot \boldsymbol{k} = 0$

Orthonormal Basis in n-dimensional Space
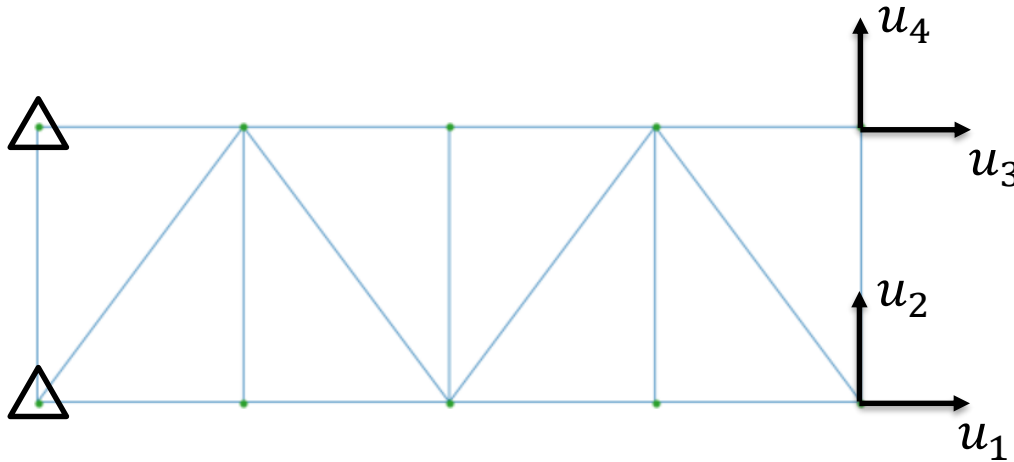
Consider:

$$\boldsymbol{u}^{(1)} = \left\{ u_1^{(1)}, u_2^{(1)}, u_3^{(1)}, u_4^{(1)}, ..., u_N^{(1)} \right\}^T$$

$$\boldsymbol{u}^{(2)} = \left\{ u_1^{(2)}, u_2^{(2)}, u_3^{(2)}, u_4^{(2)}, ..., u_N^{(2)} \right\}^T$$

$$\vdots$$

$$\boldsymbol{u}^{(N)} = \left\{ u_1^{(N)}, u_2^{(N)}, u_3^{(N)}, u_4^{(N)}, ..., u_N^{(N)} \right\}^T$$

Assuming they are orthonormal:

$$\boldsymbol{u}^{(I)} \cdot \boldsymbol{u}^{(J)} = 0$$
$$\text{for } I \neq J$$

Any vector in this MDOF space can be represented as a linear combination of different orthonormal $\boldsymbol{u}^{(I)}$:

$$\boldsymbol{u} = \sum_I^N c_I \boldsymbol{u}^{(I)}$$

Expansion Theorem

Orthogonal vectors can form a new basis in the n-dimensional space of the MDOF system.

Therefore, any vector in the n-dimensional space can be represented as a linear combination of the n linearly independent eigenvectors.

$$\boldsymbol{u} = \sum_{I}^{N} c_I \, \boldsymbol{u}^{(I)}$$

Note that **we are changing from nodal coordinates in our MDOF to modal coordinates**, using the new n-dimensional basis formed by the orthonormal eigenmodes.

From nodal coordinates to modal coordinates

Nodal coordinates:

$$\boldsymbol{u} = \{u_1, u_2, u_3, u_4, \dots, u_N\}^T$$
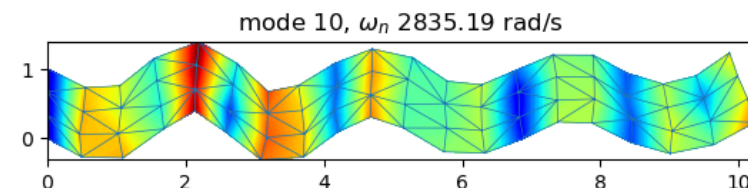
N unknowns $u_1, u_2, \dots$

Modal coordinates:

$$\boldsymbol{u} = \sum_{I}^{N} c_I \boldsymbol{u}^{(I)}$$

N unknowns $c_1, c_2, \dots$

so... What is the advantage?

From nodal coordinates to modal coordinates

What is the advantage?
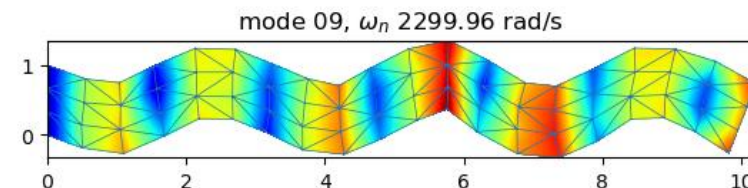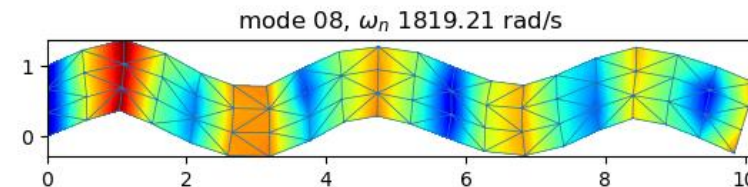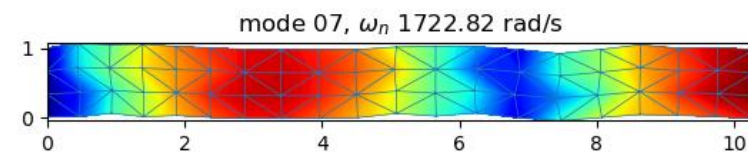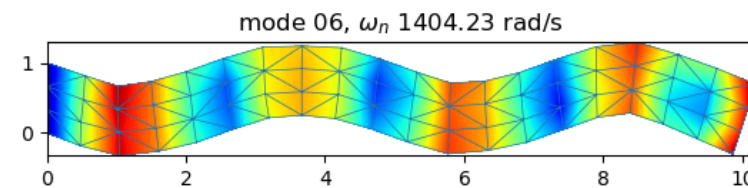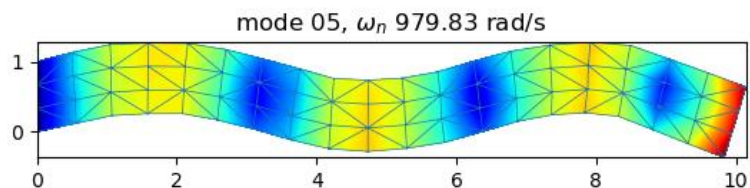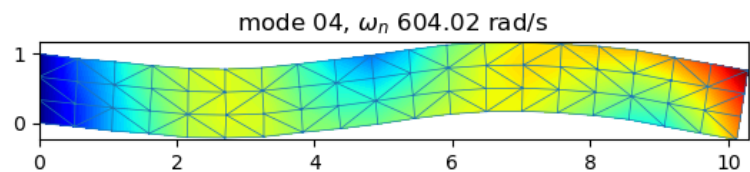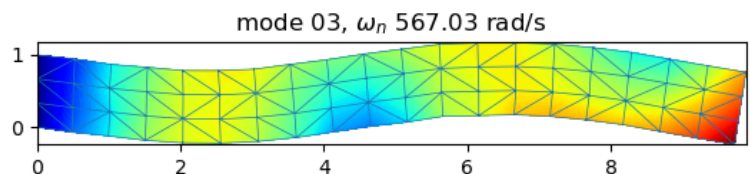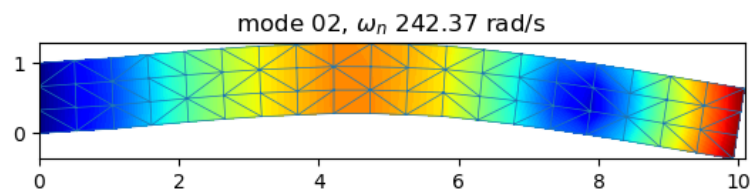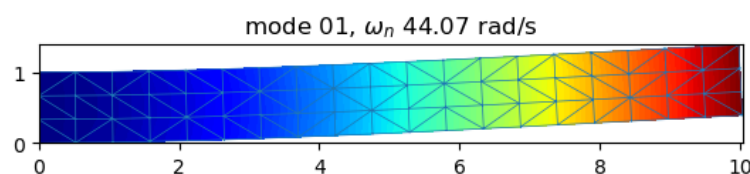
Next Steps…

- Make our mode shapes become a valid orthonormal basis

$$\boldsymbol{u}^{(I)} \cdot \boldsymbol{u}^{(J)} = 0$$
$$\text{for } I \neq J$$

Orthonormalization of Mode Shapes

$$\left(-\omega_n^{(I)^2} \boldsymbol{M} + \boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$

Rewriting as:

$$\omega_n^{(I)^2} \boldsymbol{M}\boldsymbol{U}^{(I)} = \boldsymbol{K}\boldsymbol{U}^{(I)}$$

Taking another mode:

$$\omega_n^{(J)^2} \boldsymbol{M}\boldsymbol{U}^{(J)} = \boldsymbol{K}\boldsymbol{U}^{(J)}$$

Orthonormalization of Eigenmodes

Left-multiplying by $U^{(J)^T}$

$$U^{(J)^T} \omega_n^{(I)^2} M U^{(I)} = U^{(J)^T} K U^{(I)}$$

Left-multiplying by $U^{(I)^T}$

$$U^{(I)^T} \omega_n^{(J)^2} M U^{(J)} = U^{(I)^T} K U^{(J)}$$

Note that:

$$U^{(J)^T} K U^{(I)} = U^{(I)^T} K U^{(J)}$$
$$U^{(J)^T} M U^{(I)} = U^{(I)^T} M U^{(J)}$$

Subtracting:

$$\left(\omega_n^{(I)^2} - \omega_n^{(J)^2} \right) U^{(J)^T} M U^{(I)} = 0$$

Orthonormalization of Eigenmodes

Thus, for $I \neq J$:

$$\boldsymbol{U}^{(J)^T} \boldsymbol{M} \boldsymbol{U}^{(I)} = 0$$

For $I = J$:

$$\boldsymbol{U}^{(I)^T} \boldsymbol{M} \boldsymbol{U}^{(I)} = 1$$

Thus:

$$\boldsymbol{U}^{(J)^T} \boldsymbol{M} \boldsymbol{U}^{(I)} = \delta_{IJ}$$

and:

$$\boldsymbol{U}^{(J)^T} \boldsymbol{K} \boldsymbol{U}^{(I)} = \delta_{IJ} \omega_n^{(I)^2}$$

Script `exercise08_orthonormality_checks.py`

Do we already have a orthonormal basis with $\boldsymbol{U}^{(I)}$?

Orthonormalization of Eigenmodes

We can conclude that Scipy's eigh already gives the mass-normalized eigenvectors:

https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.eigh.html#scipy.linalg.eigh

Do we already have a orthonormal basis with $U^{(I)}$?

Quick overview …

- We did separation of variables assuming that all DOFs are synchronous:

$$\boldsymbol{u}(x,t) = \boldsymbol{U}(x)T(t)$$

- Then replaced this into $\boldsymbol{Ku} + \boldsymbol{M\ddot{u}} = \boldsymbol{0}$ to get:

$$\ddot{T}(t) + {\omega_n^{(I)}}^2 T(t) = 0$$

- Which has the general solution:

$$T^{(I)}(t) = C_1^{(I)} \cos \omega_n^{(I)} t + C_2^{(I)} \sin \omega_n^{(I)} t$$

- We are seeking an orthonormal basis that will allow us to do:

$$\boldsymbol{U} = \sum_I^N c_I \boldsymbol{U}^{(I)}$$

- Such that the final solution will have the format:

$$\boldsymbol{u}(x,t) = \sum_I^N \left(c_1^{(I)} \cos \omega_n^{(I)} t + c_2^{(I)} \sin \omega_n^{(I)} t \right) \boldsymbol{U}^{(I)}$$

TUDelft

Symmetric Eigenvalue Problem

Expected format:
$$\left(\lambda^{(I)}\boldsymbol{I} + \boldsymbol{A}\right)\boldsymbol{V}^{(I)} = \boldsymbol{0}, \qquad with \; A_{ij} = A_{ji}$$

Our generalized eigenvalue problem:
$$\left(-\omega_n^{(I)^2}\boldsymbol{M} + \boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$

What can we do?

Symmetric Eigenvalue Problem

Expected format:
$$\left(\lambda^{(I)}\boldsymbol{I} + \boldsymbol{A}\right)\boldsymbol{V}^{(I)} = \boldsymbol{0}, \qquad with\ A_{ij} = A_{ji}$$

Our generalized eigenvalue problem:
$$\left(-\omega_n^{(I)^2}\boldsymbol{M} + \boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$

Left-multiplying by $\boldsymbol{M}^{-1}$:
$$\left(-\omega_n^{(I)^2}\boldsymbol{M}^{-1}\boldsymbol{M} + \boldsymbol{M}^{-1}\boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$
$$\left(-\omega_n^{(I)^2}\boldsymbol{I} + \boldsymbol{M}^{-1}\boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$

What else can we do?

Symmetric Eigenvalue Problem

Expected format:

$$\left(\lambda^{(I)}\boldsymbol{I} + \boldsymbol{A}\right)\boldsymbol{V}^{(I)} = \boldsymbol{0}, \qquad with\ A_{ij} = A_{ji}$$

Our generalized eigenvalue problem:

$$\left(-\omega_n^{(I)^2}\boldsymbol{M} + \boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$

Represent $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^*$ (Cholesky decomposition), which is $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^{\boldsymbol{T}}$ for symmetric matrices

$$\left(-\omega_n^{(I)^2}\boldsymbol{L}\boldsymbol{L}^{\boldsymbol{T}} + \boldsymbol{K}\right)\boldsymbol{U}^{(I)} = \boldsymbol{0}$$

COBEM
UBERLANDIA
2019

TUDelft

63

Symmetric Eigenvalue Problem

$$\left(-\omega_n^{(I)^2} LL^T + K\right) U^{(I)} = 0$$

Assuming $U = L^{-T}V$ and left-multiplying by L^{-1}:

$$\left(-\omega_n^{(I)^2} L^{-1}LL^T L^{-T} + L^{-1}KL^{-T}\right) V^{(I)} = 0$$

$$\left(-\omega_n^{(I)^2} I + \widetilde{K}\right) V^{(I)} = 0$$

with:
$$\widetilde{K} = L^{-1}KL^{-T}$$

Mass-normalized stiffness matrix

Exercise with Symmetric Eigenvalue Problem

Using script `exercise09_symmetric_eigenvalue_problem.py`
Let's solve:

$$\left(-{\omega_n^{(I)}}^2 I + \widetilde{K}\right) V^{(I)} = 0$$

with:
$$\widetilde{K} = L^{-1} K L^{-T}$$

And see the properties of $V^{(I)}$

NOTE: $V^{(I)}$ calculated with `np.linalg.eigh` are already normalized to $\left|V^{(I)}\right| = 1$

Solving our free undamped vibration problem

From:

$$\boldsymbol{u}(x,t) = \sum_{I}^{N} \left(c_1^{(I)} \cos \omega_n^{(I)} t + c_2^{(I)} \sin \omega_n^{(I)} t \right) \boldsymbol{U}^{(I)}$$

Using $\boldsymbol{U} = \boldsymbol{L}^{-T}\boldsymbol{V}$:

$$\boldsymbol{u}(x,t) = \sum_{I}^{N} \left(c_1^{(I)} \cos \omega_n^{(I)} t + c_2^{(I)} \sin \omega_n^{(I)} t \right) \boldsymbol{L}^{-T}\boldsymbol{V}^{(I)}$$

The problem now consists on finding constants $c_1^{(I)}$ and $c_2^{(I)}$ …

Solving our free undamped vibration problem

$$\boldsymbol{u}(x,t) = \sum_{I}^{N} \left(c_1^{(I)} \cos \omega_n^{(I)} t + c_2^{(I)} \sin \omega_n^{(I)} t \right) \boldsymbol{L}^{-T} \boldsymbol{V}^{(I)}$$

$$\dot{\boldsymbol{u}}(x,t) = \sum_{I}^{N} \left(-\omega_n^{(I)} c_1^{(I)} \sin \omega_n^{(I)} t + \omega_n^{(I)} c_2^{(I)} \cos \omega_n^{(I)} t \right) \boldsymbol{L}^{-T} \boldsymbol{V}^{(I)}$$

Initial conditions: $\boldsymbol{u}(x,0) = \boldsymbol{u_0}$:

$$\boldsymbol{u_0} = \sum_{I} c_1^{(I)} \boldsymbol{L}^{-T} \boldsymbol{V}^{(I)}$$

Left-multiplying by $\boldsymbol{L}^T$ and $\boldsymbol{V}^{(J)}$

$$\boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{u_0} = \sum_{I} c_1^{(I)} \boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{L}^{-T} \boldsymbol{V}^{(I)}$$

$$\boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{u_0} = \sum_{I} c_1^{(I)} \boldsymbol{V}^{(J)} \boldsymbol{I} \boldsymbol{V}^{(I)}$$

$$c_1^{(J)} = \boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{u_0}$$

Initial conditions: $\dot{\boldsymbol{u}}(x,0) = \boldsymbol{v_0}$:

$$\boldsymbol{v_0} = \sum_{I} c_2^{(I)} \omega_n^{(I)} \boldsymbol{L}^{-T} \boldsymbol{V}^{(I)}$$

Left-multiplying by $\boldsymbol{L}^T$ and $\boldsymbol{V}^{(J)}$

$$\boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{v_0} = \sum_{I} c_2^{(I)} \omega_n^{(I)} \boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{L}^{-T} \boldsymbol{V}^{(I)}$$

$$\boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{v_0} = \sum_{I} c_2^{(I)} \omega_n^{(I)} \boldsymbol{V}^{(J)} \boldsymbol{I} \boldsymbol{V}^{(I)}$$

$$c_2^{(J)} = \frac{\boldsymbol{V}^{(J)} \boldsymbol{L}^T \boldsymbol{v_0}}{\omega_n^{(J)}}$$

Example: Free vibration of a truss
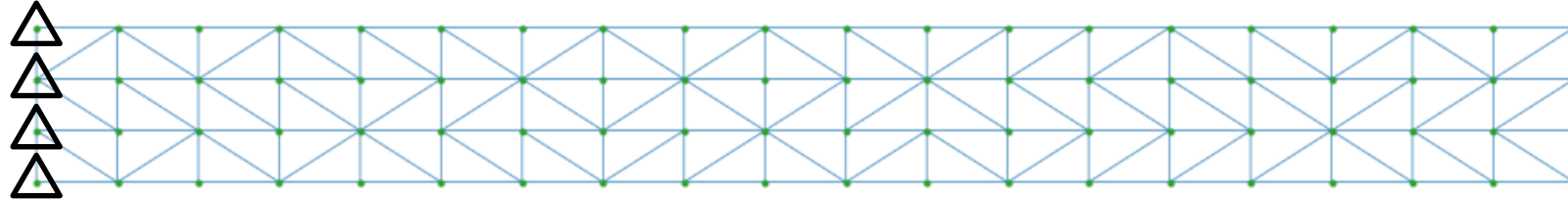
Using script
`exercise10_free_undamped_vibration_truss.py`

- Check the two initial conditions for u_0

- Check how good is our approximation for different number of modes

- Does the number of modes for a good approximation changes with the initial condition?

Saullo G. P. Castro

Transforming a MDOF system into uncoupled SDOF systems

Objective:

Use the relations for SDOF that we already know for forced vibration, damped/undamped

python™

Saullo G. P. Castro

MDOF system into uncoupled SDOF systems

Equation of motion for free undamped vibration:

$$M\ddot{u} + Ku = 0$$

From the previous definition: $U(x) = L^{-T}V(x)$; multiplying both sides by $T(t)$ gives $u(x,t) = L^{-T}v(x,t)$, thus:

$$I\ddot{v} + \widetilde{K}v = 0$$

With $\widetilde{K} = L^{-1}KL^{-T}$. This is a set of N coupled equations. Using our orthonormal eigenvectors of $\widetilde{K}$, previously named $V^{(I)}$, let's define:

$$P(x) = [V^{(1)} \quad V^{(2)} \quad ... \quad V^{(p)}]$$

Where p is the number of modes wanted in the approximation. Defining $v(x,t) = P(x)r(t)$ and left-multiplying by P^T gives:

$$P^T P\ddot{r} + P^T \widetilde{K} P r = 0$$

Let`s check the properties of this system
script `exercise11_check_uncoupling.py`

COBEM
UBERLANDIA
2019

TUDelft

MDOF system into uncoupled SDOF systems

$$P^T P \ddot{r} + P^T \widetilde{K} P r = 0$$

Gives us an uncoupled system with p equations, $p \leq N$:

$$I \ddot{r}(t) + \Lambda r(t) = 0$$

$$\Lambda = P^T \widetilde{K} P$$

Note that Λ is simple (it does not have to be calculated):

$$\Lambda = \begin{bmatrix} \left(\omega_n^{(1)}\right)^2 & & & 0 \\ & \left(\omega_n^{(2)}\right)^2 & & \\ & & \ddots & \\ 0 & & & \left(\omega_n^{(p)}\right)^2 \end{bmatrix}$$

MDOF system into uncoupled SDOF systems

The uncoupled system has the form:

$$\ddot{r}_I + \left(\omega_n^{(I)}\right)^2 r_I = 0 \qquad I = 1,2,\ldots,p$$

Noting that $r_I = r_I(t)$, the general solution is, for each DOF:

$$r_I(t) = C_1^{(I)} \cos \omega_n^{(I)} t + C_2^{(I)} \sin \omega_n^{(I)} t$$

Where $r(0)$ and $\dot{r}(0)$ are needed, which can be calculated from $\boldsymbol{u_0}(x,t)$ and $\boldsymbol{v_0}(x,t)$. Given that:

$$\boldsymbol{v} = \boldsymbol{L}^T \boldsymbol{u}$$
$$\boldsymbol{v} = \boldsymbol{P}\boldsymbol{r}, \qquad \boldsymbol{r} = \boldsymbol{P}^T \boldsymbol{v}$$

Thus:

$$\boldsymbol{r} = \boldsymbol{P}^T \boldsymbol{L}^T \boldsymbol{u}$$
$$\boldsymbol{u} = \boldsymbol{L}^{-T} \boldsymbol{P}\, \boldsymbol{r}$$

Damped free vibration

The SDOF system has the form:

$$\ddot{r}_I + 2\textcolor{red}{\zeta_I}\omega_{\mathsf{n}}^{(I)}\dot{r}_I + \left(\omega_{\mathsf{n}}^{(I)}\right)^2 r_I = 0 \qquad I = 1,2,\dots,p$$

Note that the concept of damping ratio can be adopted for each mode (modal damping).

Undamped forced vibration

The SDOF system has the form:

$$\ddot{r}_I + \left(\omega_n^{(I)}\right)^2 r_I = f_I \qquad I = 1, 2, \ldots, p$$

Here f_I is called modal force, which is the Ith row of $\boldsymbol{f = P^T L^{-1} F}$

$$\boldsymbol{M\ddot{u} + Ku = F} \qquad\qquad \boldsymbol{u = L^{-T}v}$$
$$\boldsymbol{ML^{-T}\ddot{v} + KL^{-T}v = F}$$
$$\boldsymbol{LL^T L^{-T}\ddot{v} + KL^{-T}v = F}$$
$$\boldsymbol{L^{-1}LL^T L^{-T}\ddot{v} + L^{-1}KL^{-T}v = L^{-1}F}$$
$$\boldsymbol{I\ddot{v} + \widetilde{K}v = L^{-1}F}$$
$$\boldsymbol{P\ddot{r} + \widetilde{K}Pr = L^{-1}F} \qquad\qquad \boldsymbol{v = Pr}$$
$$\boldsymbol{P^T P\ddot{r} + P^T \widetilde{K}Pr = P^T L^{-1}F}$$
$$\boldsymbol{I\ddot{r} + \Lambda r = f}$$

Damped forced vibration

The SDOF system has the form:

$$\ddot{r}_I + 2\zeta_I \omega_{\mathrm{n}}^{(I)} \dot{r}_I + \left(\omega_{\mathrm{n}}^{(I)}\right)^2 r_I = f_I \qquad I = 1,2,\ldots,p$$

TODO provide a general case forced damped impulse, convolution

Modal Analysis
with

python

Saullo G. P. Castro

Considerably more efficient implementations

Objective: Learn how to achieve much higher computational efficiency, important for large systems

Bottlenecks of current approach

Cholesky decomposition $M = LL^T$ and computation of L^{-1}

We are using dense matrices to represent sparse systems

Algorithms `scipy.linalg.eigh` and `numpy.linalg.eigh` compute all N pairs of eigenvalues and eigenvectors, even if we only need p for the modal analysis

Pure Python

Lumped Mass

Here $\boldsymbol{M}$ is diagonal, such that $\boldsymbol{L}$ will also be diagonal:

$$L_{ii} = \sqrt{M_{ii}}$$

Thus, $\boldsymbol{L}^{-1}$ is also diagonal:

$$L_{ii}^{-1} = \frac{1}{L_{ii}}$$

SciPy's Sparse Matrices

We can define all structural matrices M, K using
`scipy.sparse.csr_matrix` (row-compact) or
`scipy.sparse.csc_matrix` (column-compact)
Both are easily created from `scipy.sparse.coo_matrix`

Build everything already using sparse matrices

SciPy's Sparse Solvers

Eigenvalue solver `scipy.sparse.linalg.eigsh` works for both generalized and symmetric eigenvalue problems

Allows the computation of the first p eigenvalues that will be used in modal analysis, much more efficient than computing all N.

Saullo G. P. Castro

Numba or Cython

These are options to compile the core part of the code.

Cython requires more experience, especially in C/C++

Numba offers JIT compilers, which really work in functions taking only integers, floats, NumPy arrays, and no classes, dictionaries and other Python objects

Example for 2 million degrees-of-freedom

Saullo G. P. Castro

```
Script: exerciseXX_2million_dofs.py

Creating mesh
    Delaunay
    done (13.800249 s)
done (33.660385 s)
Number of degrees-of-freedom: 2000000
Computing K, M
done (17.646885 s)
Partitioning due to boundary conditions
done (1.233670 s)
Solving symmetric eigenvalue problem
[ 40.11186674 213.93027026 504.00858015 640.84402584]
done (196.419322 s)
```

Modal Analysis
with

python

Saullo G. P. Castro

Closing Remarks

COBEM
UBERLANDIA
2019

TUDelft

This course should be seen as an initial reference for you

Implement other types of finite element

Modal Analysis
with
python

Saullo G. P. Castro

Keep in touch in case you have any questions!

S.G.P.Castro@tudelft.nl

COBEM
UBERLANDIA
2019

TUDelft