ENG-A2001
Computer-aided Tools in Engineering

Autumn 2021, periods I-II, 5 credits, BSc level
Week 4

Fundamentals of Finite Element Analysis

Jarkko Niiranen
Associate Professor

Department of Civil Engineering
School of Engineering
Aalto University

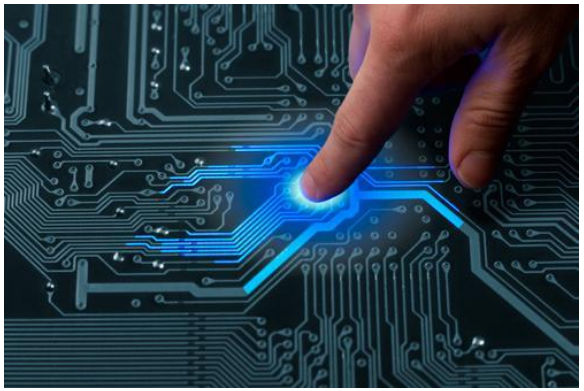
2021, October 4

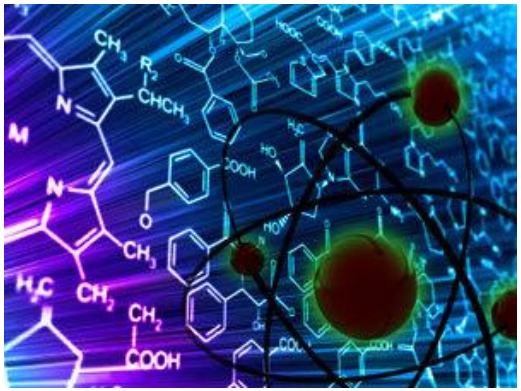
A MOTIVATION TO

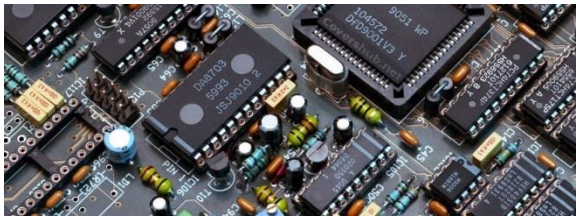
computational engineering

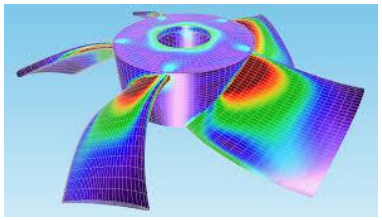
Motivation to computational engineering

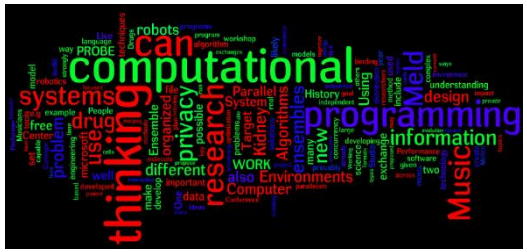
Motivation to computational engineering

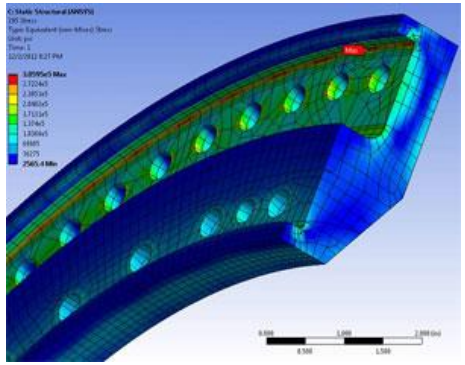


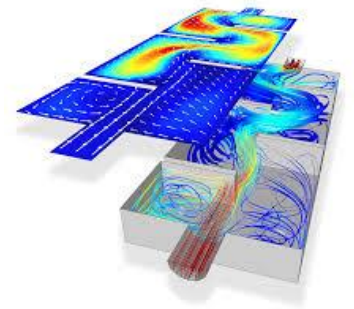




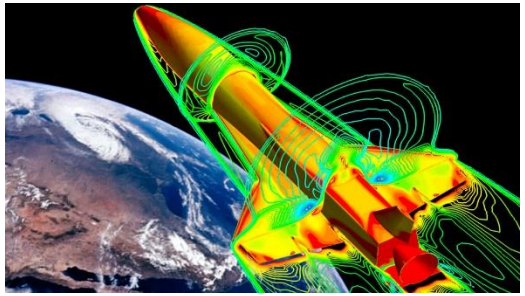


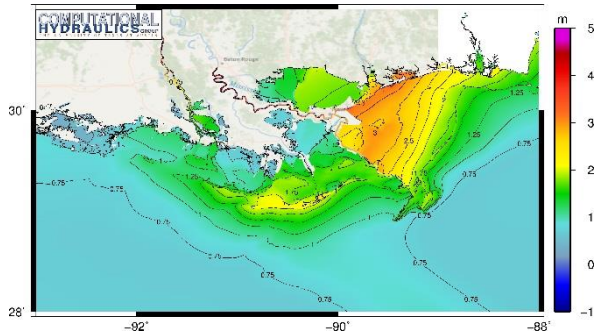




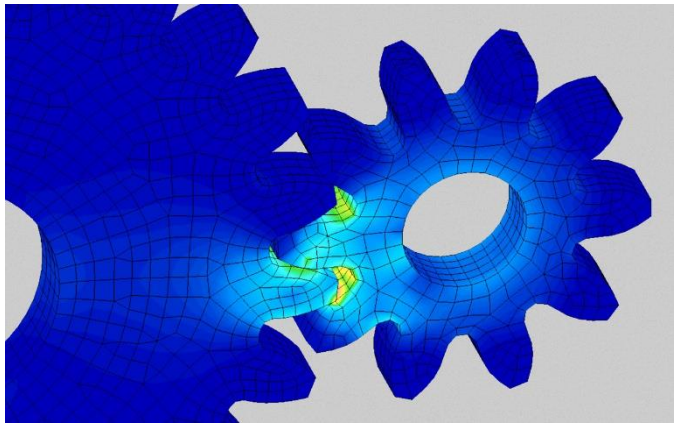


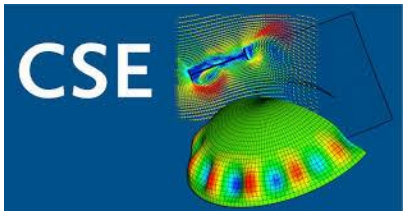
The role of CAx?

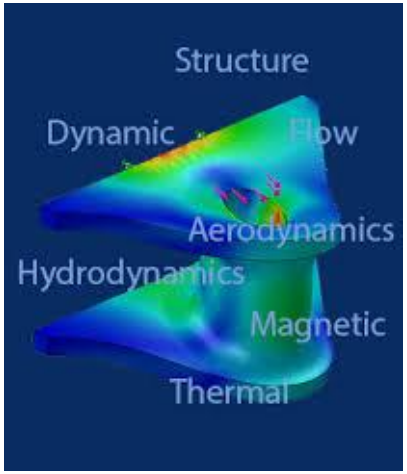


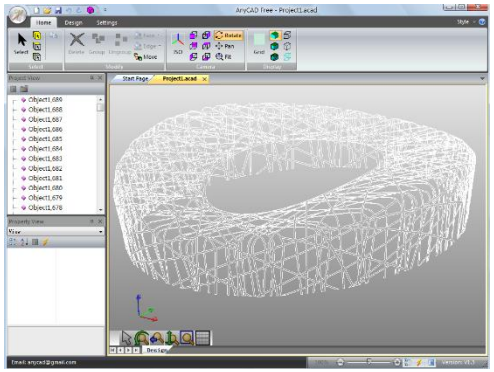


Motivation to computational engineering

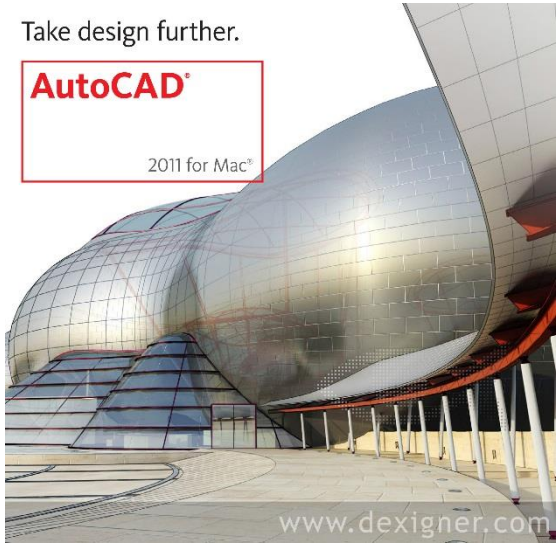


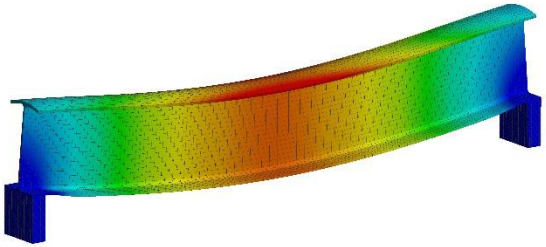


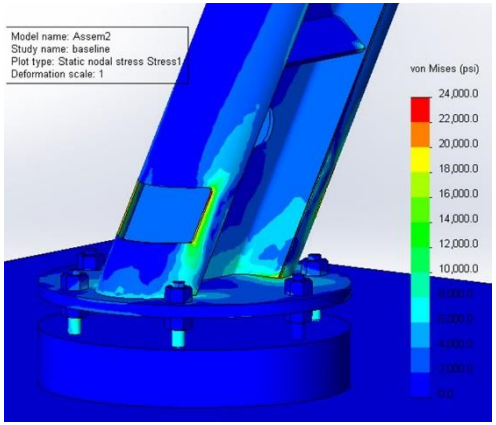




The role of CAx?

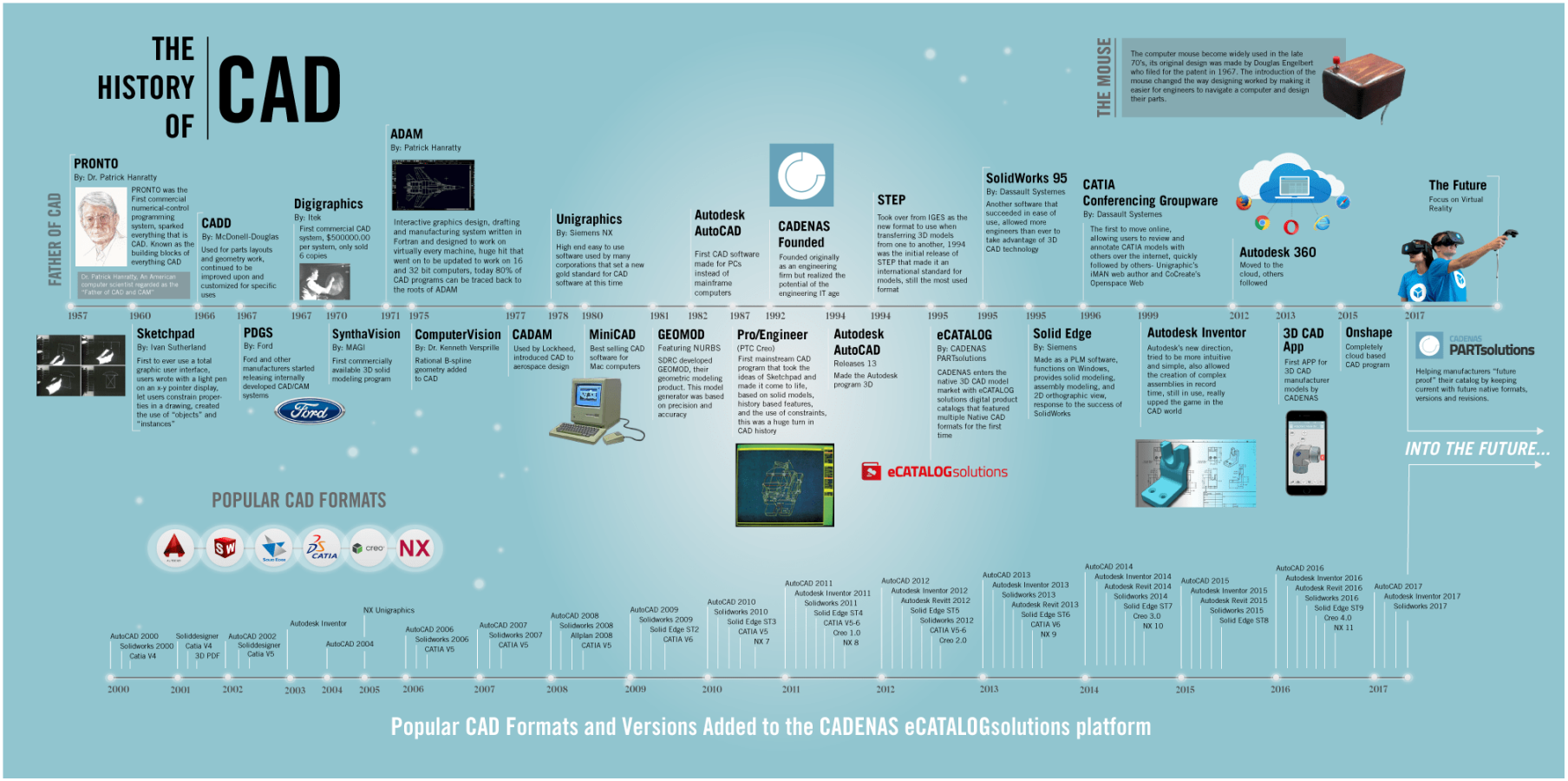






Motivation to computational engineering

Today, the world is full of CAD (BIM, PDM, PLM etc.) and CAE software, but they all rely on the same (old or new) principles of *computer science* and *mathematics*.



Motivation to computational engineering

Today, the world is full of CAE software, especially FEM software, but they all rely on the same (historical) principles of *mathematics*, *physics* and *computer science*.

The FEM has its roots in mathematics and structural mechanics (especially in aeronautics)

- 1941 – Hrennikoff and Courant,…
- 1953 – Turner (Boeing),...
- 1954 – Argyris,...
- 1956 – Turner, Clough, Martin and Topp,...
- 1965 – NASTRAN (NASA Structural Analysis),…
- 1978 – Abaqus FEA,…
- 2005 – Isogeometric Analysis,…

but its development is linked to the developments in computer-aided design, programming, personal computers, graphical user interfaces and other engineering software:

- 1957 – FORTRAN (Formula Translator),…
- 1959 – DAC-1 (by GM and IBM)
- 1972 – C,…
- 1977 – Apple II, Commodore PET,…
- 1985 – C++, Windows, Excel,…

1956
The first paper on the FEM published
The first paper on the finite element method (FEM) was published by Turner, M. J., Clough, R. W., Martin H. C. and Topp, L. J. in 1956. From Wikipedia (on Clough):

> *His article in 1956 was one of the first applications of this computational method. He coined the term "finite elements" in an article in 1960.*

Turner, M. J., Clough, R. W., Martin H. C. and Topp, L. J. Stiffness and Deflection Analysis of Complex Structures . Journal of the Aeronautical Sciences, Vol. 23 No. 9, 1956 pp. 805-823.

1965
NASA RFP for structural analysis software
In 1965, NASA issued a request for a proposal for the development of a structural analysis software tool. The result of this was NASTRAN (NASA Structural Analysis), which implemented the available FEM technology to solve structural problems.

1967
First book on the FEM published
Dr. O.C. Zienkiewicz publishes the first text on the finite element method, called 'The Finite Element Method'. To this day, it remains a standard reference text for the basis of the FEM.

Motivation to computational engineering

Links to the fundamental BSc studies

Mathematics:
MS-A0105 Differentiaali- ja integraalilaskenta 1
MS-A0205 Differentiaali- ja integraalilaskenta 2
MS-A0305 Differentiaali- ja integraalilaskenta 3
MS-A0005 Matriisilaskenta

Computer science:
ENG-A1001 Tietokoneavusteiset työkalut insinööritieteissä
CSE-A1111 Ohjelmoinnin peruskurssi Y1
CSE-A1130 Tietotekniikka sovelluksissa

Mechanics:
KJR-C1001 Statiikka ja Dynamiikka
KJR-C2001 Kiinteän aineen mekaniikan perusteet
KJR-C2002 Kontinuumimekaniikan perusteet
KJR-C2003 Virtausmekaniikan perusteet
ENY-C2001 Termodynamiikka ja lämmönsiirto

Physics and chemistry:
PHYS-A3120 Termodynamiikka
PHYS-A3130 Sähkömagnetismi
CHEM-A1250 Kemian perusteet

Engineering:
KJR-C2004 Materiaalitekniikka
KJR-C2005 Tuotesuunnittelu
KJR-C2006 Tuotantotekniikka
ENG-A1002 ARTS-ENG-projekti

BACK
TO
basic material

Fundamentals of Finite Element Analysis

Lecture 1

1. Modelling principles and boundary value problems in engineering sciences
   - Home exercise 1.1
   - Computer exercise 1.1 – Matlab
2. Energy methods and basic 1D finite element methods
   - bars/rods, beams, heat diffusion, seepage, electrostatics
   - Home exercise 2.1
   - Home exercise 2.2

Lecture 2

3. Basic 2D and 3D finite element methods
   - heat diffusion, seepage
   - Computer exercise 3.1 – Matlab PDE Modeler
   - Computer exercise 3.2 – Matlab PDE Modeler
4. Mesh refinements, adaptive and isogeometric methods
   - Computer exercise 4.1 – Matlab PDE Modeler (VOLUNTARY)

1 Modelling principles and boundary value problems in engineering sciences

Let us start with some simulation examples (far too advanced for this course):

Cutting process http://www.adina.com/newsgH141.shtml

Shell folding http://www.adina.com/newsgH118.shtml

Stamping http://www.adina.com/stamping.shtml

Bar vibrations in fluid http://www.adina.com/newsgH137.shtml

Sail ship mast http://www.adina.com/newsgH146.shtml

Fastener joints http://www.adina.com/newsgH150.shtml

Hemming http://www.adina.com/hemming.shtml

1 Modelling principles and boundary value problems in engineering sciences

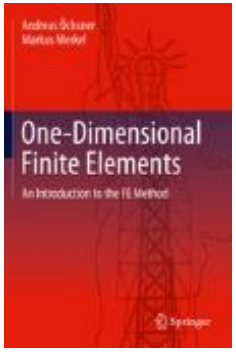
Contents

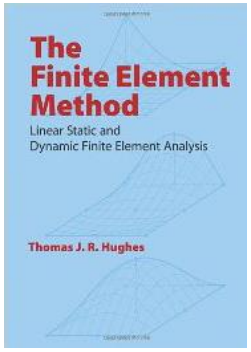
1. *Modelling and computation* in engineering design and analysis
2. *Computer-aided design* (CAD) and *engineering* (CAE)

Learning outcome

A. *Understanding* of the main implications of the approximate nature of computational methods in engineering design and analysis

B. *Recognizing* the importance of computational techniques in modern engineering

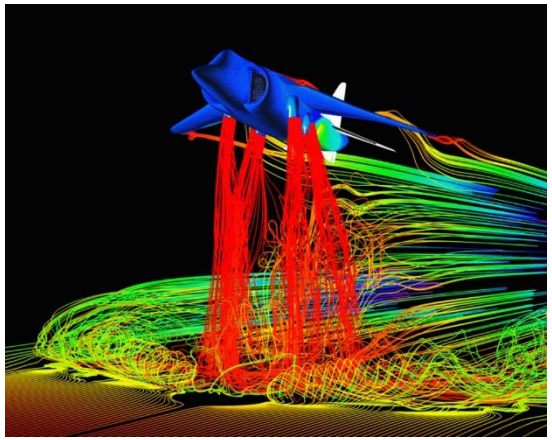


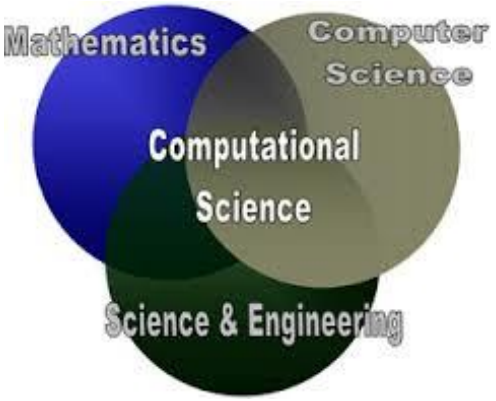


References

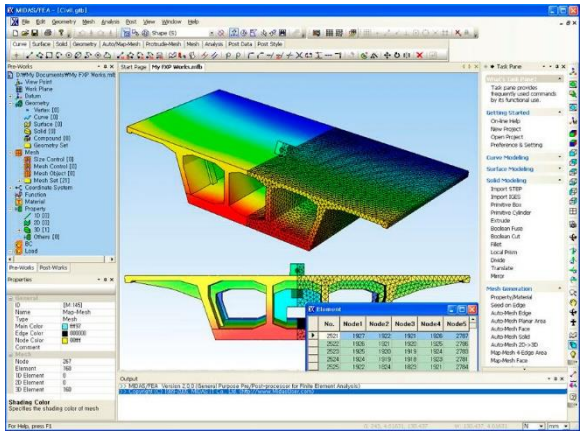
Text book 1 (red): chapter 1
Text book 2 (blue): chapters 1.1−2

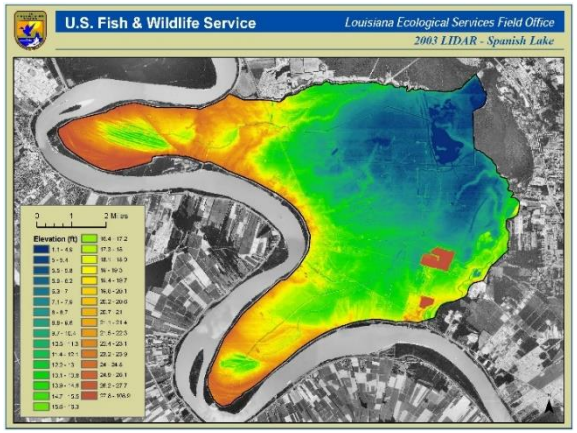
1.0 Questioning computational analysis

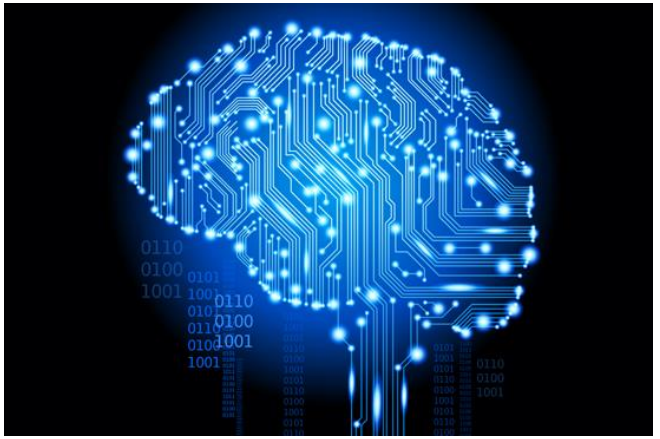




How well do the computational techniques − of different engineering fields − simulate the real life?







1.1 Modeling and computation in engineering design and analysis

The world is full of *data* – think about

- World Wide Web (www)
- Facebook (Fb), Instagram (Ig)
- Internet of Things (IoT).

But which kind of data? Most often, quite simple data – think about

- www (visual, written, verbal): words, pictures, videos,…
- Fb, Ig: likes, hashtags,…
- IoT: sensor values for coordinates, temperatures, hours,…

Data scientists/analysts collect, agregate and analyze data – in order to form information and models for design and decisions making – by developing and utilizing *algorithms* (by means of *mathematics*, *computer science* and the “laws” of applications fields (such as population growth models or economic models).

Data science utilizes the classical tools of *statistics* and *stochastics (*for analyzing the political and economical behavior of people, for instance) and the modern trends of *data mining* (DM), *machine learning* (ML), *artificial intelligence* (AI) etc.

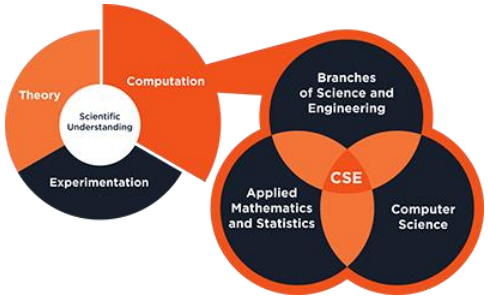
1.1 Modeling and computation in engineering design and analysis

The tools of data analysis are – definitely – demanding and scientific but the nature and origin of data (vote, like, purchase) is often very simple.

In *computational engineering*, most often, data originates from and is related to complex physical systems (such as buildings or machines) and the laws of the application field are quite complex (such as structural or fluid dynamics).

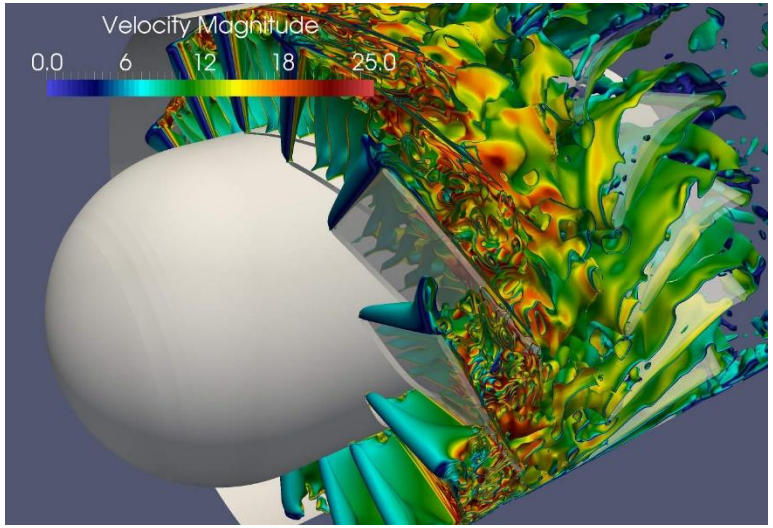
Actually, computational engineering (such as *computational mechanics* or *computational fluid dynamics* is, to large extent, different than other *computational sciences* (such as *data science* or even *computational modeling* such as *geometric modeling*) although the same components and terms are typically present:

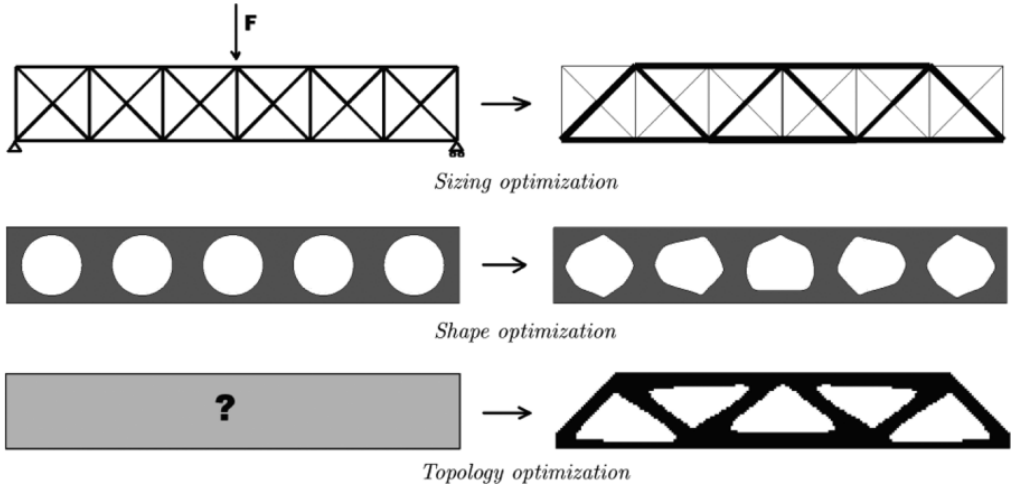
- modeling and simulation
- data structures and algorithms
- data analysis and visualization
- high-performance computing

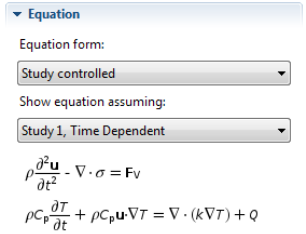


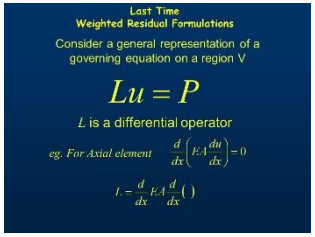
"The computational engineer uses the computer and mathematical algorithms to solve *physics-based equations* to make predictions and simulate scenarios."

1.1 Modeling and computation in engineering design and analysis









1.1 Modeling and computation in engineering design and analysis

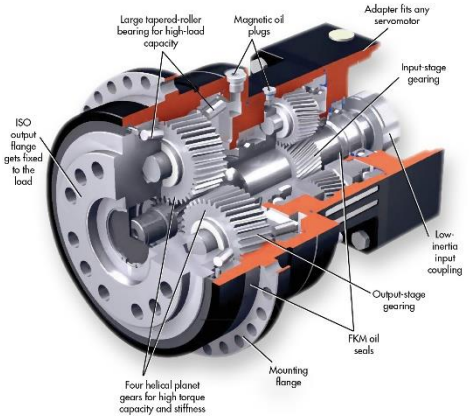
step 0

The role of geometry?

Physical engineering problem with design criteria

solution $\boldsymbol{u}_P$ = ?

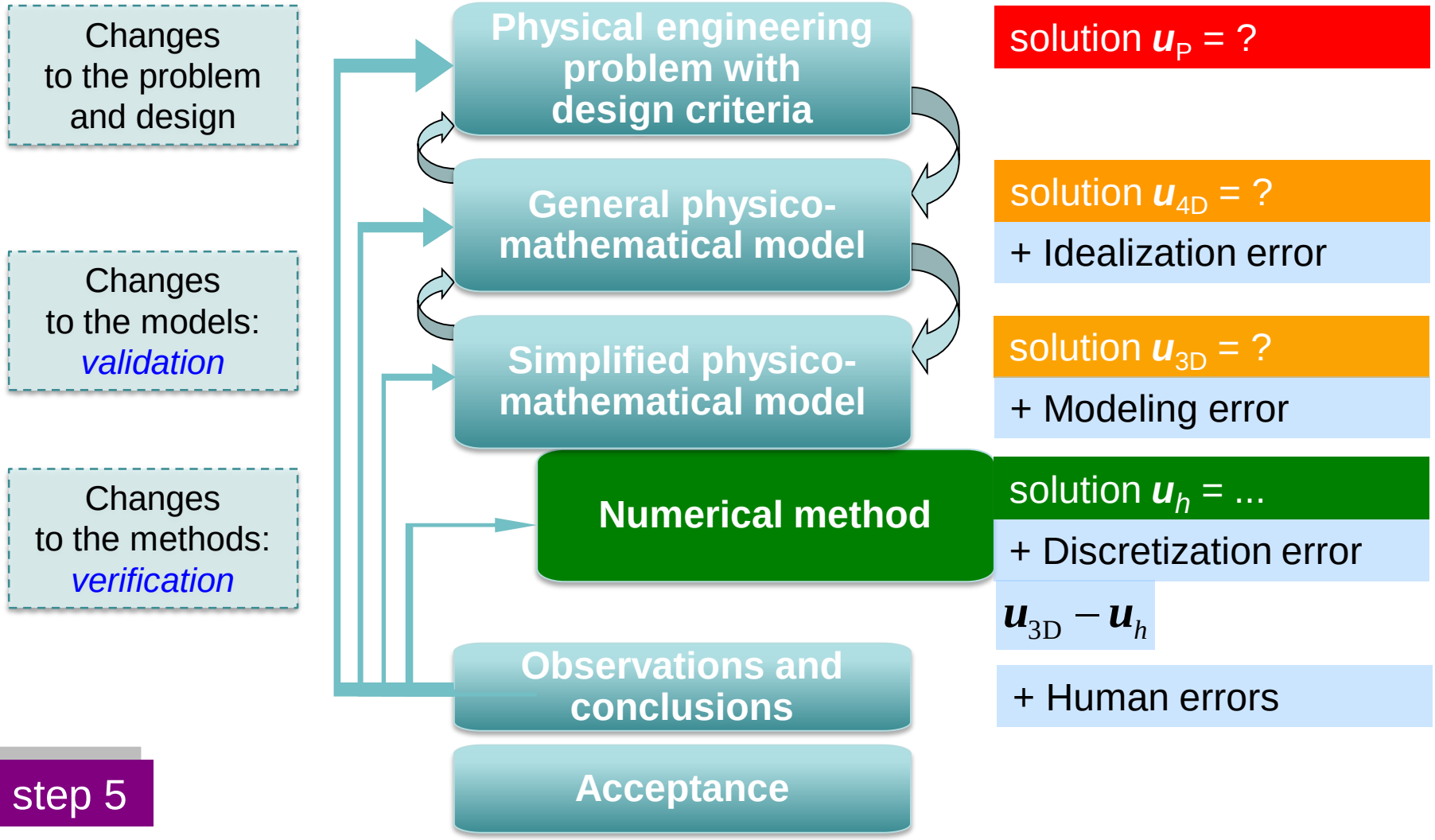
Customer needs!
Dimensions!
Laws and regulations!
Time slot!
Technology available!
Price range!
…



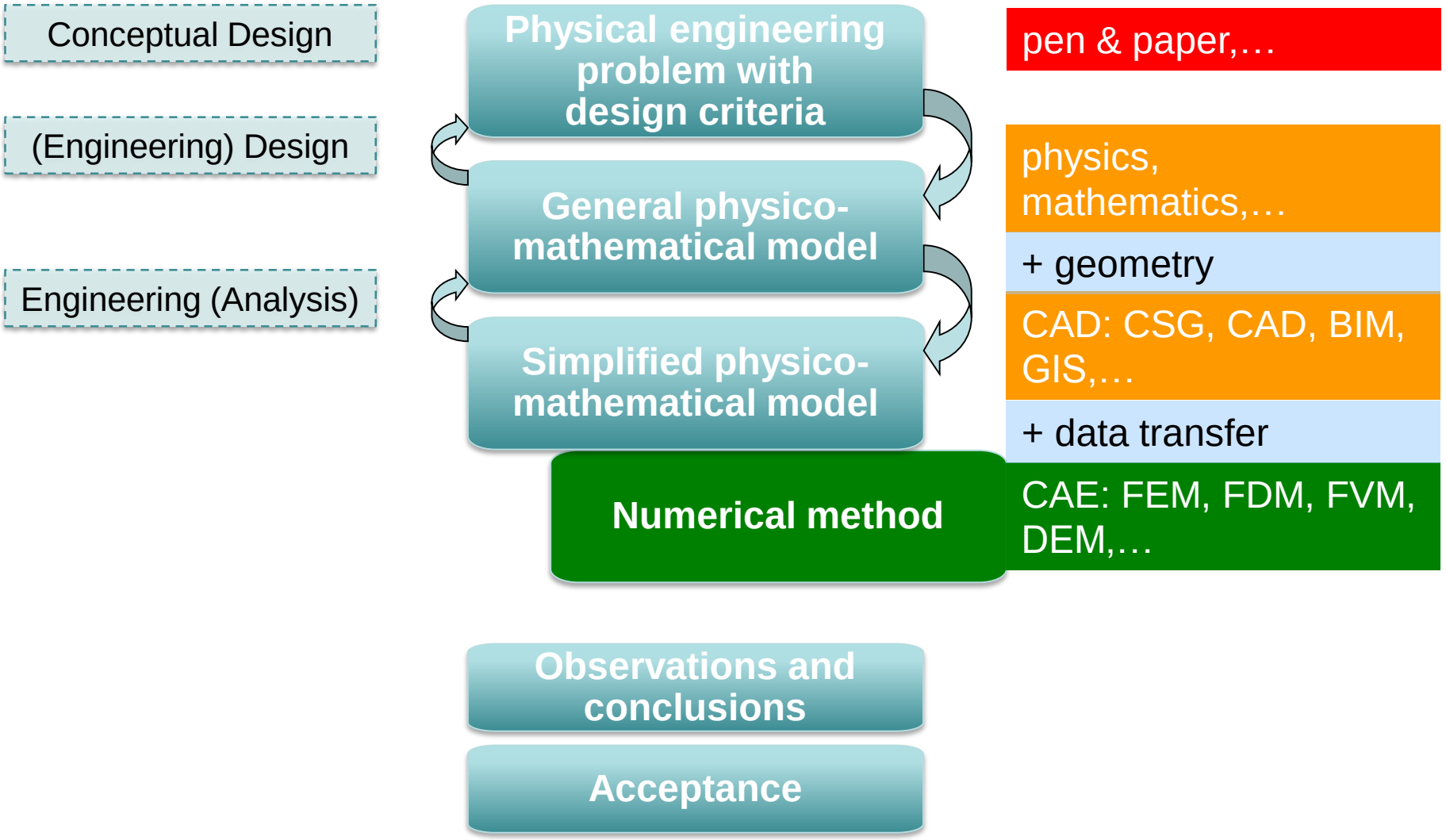
How long?
How thick?
Which material?
How many?
Which joints?
How to construct?
…

How to get answers?

1.1 Modeling and computation in engineering design and analysis



1.2 Computer-aided design and engineering



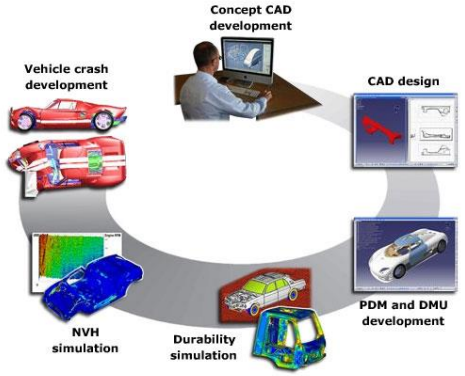
1.2 Computer-aided design and engineering

CAD – Computer-Aided Design

CAM – Computer-Aided Manufacturing

CAE – Computer-Aided Engineering

… Aided (‘70s)
… Based (‘90s)
… Driven



BIM – Building Information Modelling
BDM – Product Data Management
PLM – Product Lifecycle Management

GIS – Geographic Information System

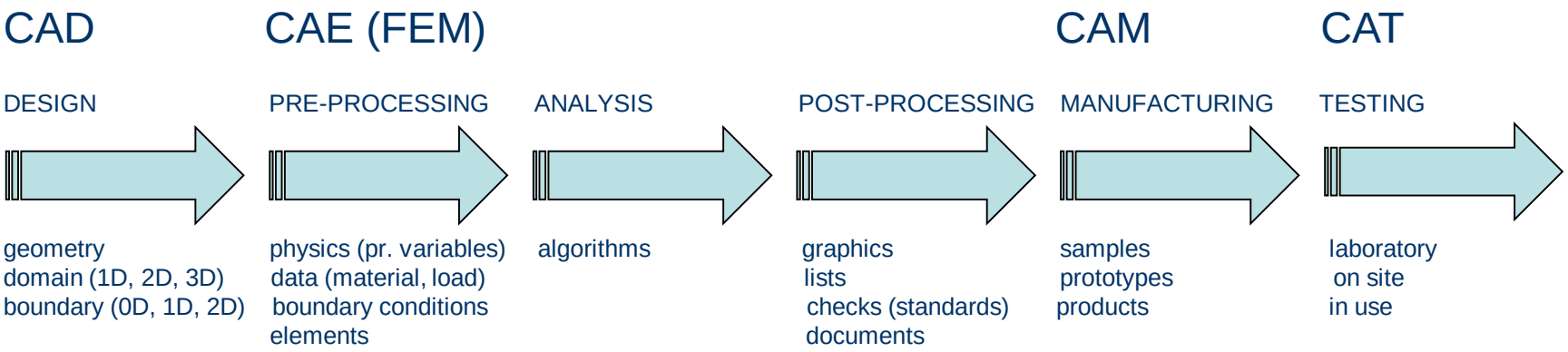
… Data (‘80s)
… Information, Lifecycle (‘60s, ‘80s, ’90s)
… Intelligence

Remark. It seems that the future trends of engineering such as *artificial intelligence* (AI) and robotics, in particular, will heavily rely on and drive computational (computer-aided, computer-driven, algorithm-aided) science and engineering.

Remark. Physics and mathematics will not change too much, however.

1.2 Computer-aided design and engineering

Computer-aided design, engineering, manufacturing and testing as a software-related process. Basically, one can speak about the following process phases (although there are alternative classifications): conceptual, preliminary and detailed *design*, *pre-processing*, *analysis*, *post-processing*, *manufacturing*, *testing*.

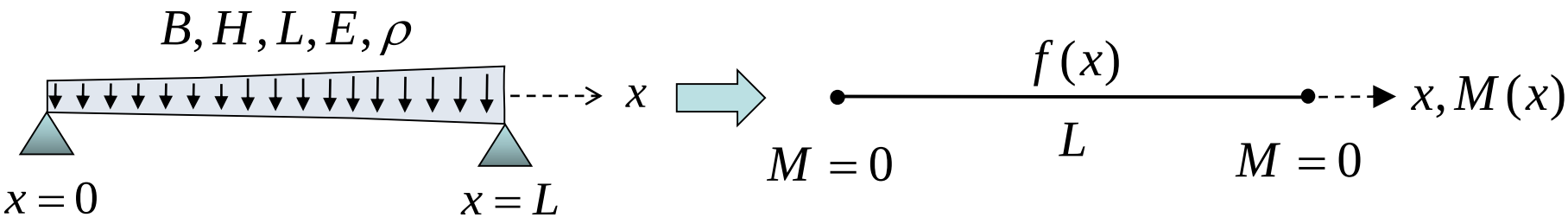


Remark. Typically, different software are used for CAD and CAE, but some CAD-software include some CAE-features, whereas almost every CAE-software includes some basic CAD-features in the GUI for post-processing.

Remark. D (design) is not the core of engineering but E, M and T!

Home exercise 1.1

Let us consider a vertically loaded 2D/3D *beam* structure (made of steel) with the following dimensions: $B = 4\ \text{cm}, H = 2\ \text{cm}, L = 1\ \text{m}.$



Under certain 2D/3D-to-1D *dimension reduction* assumptions, the *bending moment* of the (statically determined) beam can be solved from the following 1D boundary value problem (of *2nd order ordinary differential equation*):

$$-M''(x) = f(x), \quad 0 < x < L$$

$$M(0) = 0, M(L) = 0$$

Derive the exact (analytical) solution of the problem for a constant loading $f(x) = f_0$ (comparable to the gravity loading).

Hint: Integrate and then determine the integration constants.

Computer exercise 1.1 – Matlab

(i) Represent graphically the exact solution $M = M(x)$ of Home exercise 1.2 with loading $f = f(x) = f_0$ by using the **MATLAB** software.

Plot the curve by choosing 10 and 100 points for discretizing the curve.

Hint: Use Matlab help for learning the syntax of the *plot* command.

(ii) Comment your code, i.e., explain what is done on each line.

2 Energy methods and basic 1D finite element methods

Fundamentals of Finite Element Analysis

Lecture 1

1. Modelling principles and boundary value problems in engineering sciences
   - Home exercise 1.1
   - Computer exercise 1.1 – Matlab
2. Energy methods and basic 1D finite element methods
   - bars/rods, beams, heat diffusion, seepage, electrostatics
   - Home exercise 2.1
   - Home exercise 2.2

Lecture 2

3. Basic 2D and 3D finite element methods
   - heat diffusion, seepage
   - Computer exercise 3.1 – Matlab PDE Modeler
   - Computer exercise 3.2 – Matlab PDE Modeler
4. Mesh refinements, adaptive and isogeometric methods
   - Computer exercise 4.1 – Matlab PDE Modeler (VOLUNTARY)

2 Energy methods and basic 1D finite element methods

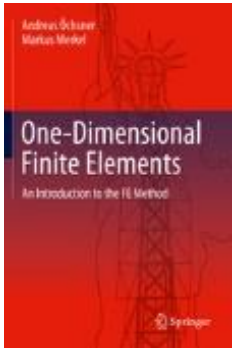
Contents

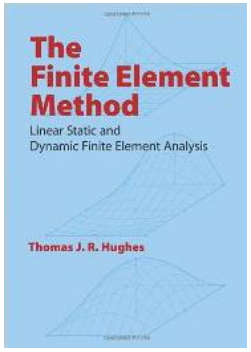
1. *Weak form* (based on the principle of virtual work)
2. *1D finite element method* (based on the weak form)

Learning outcome

A. *Understanding of the main principles* behind the finite element method

B. *Ability to formulate and apply* the finite element method for 1D model problems



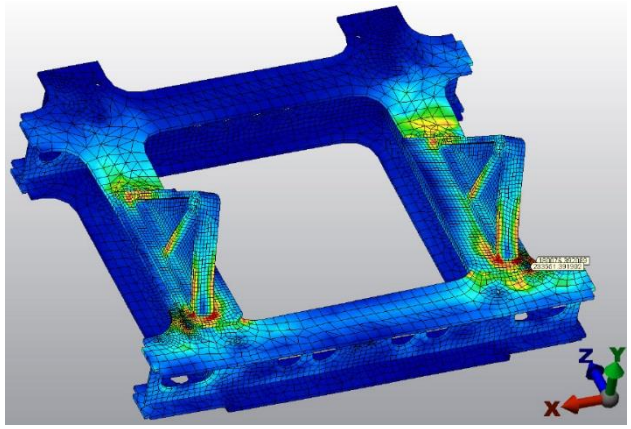


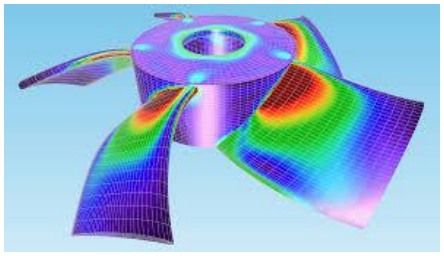
References

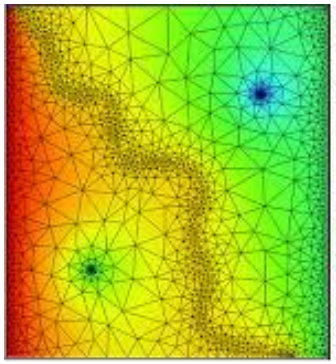
Text book 1: chapters 3 and 4

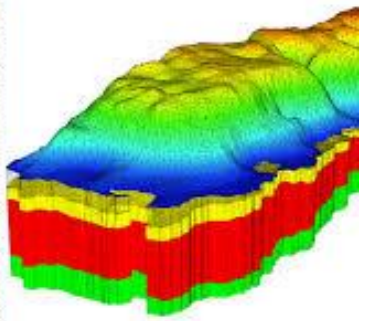
Text book 2: chapters 1.1−11

2.0 Finite element methods intuitively

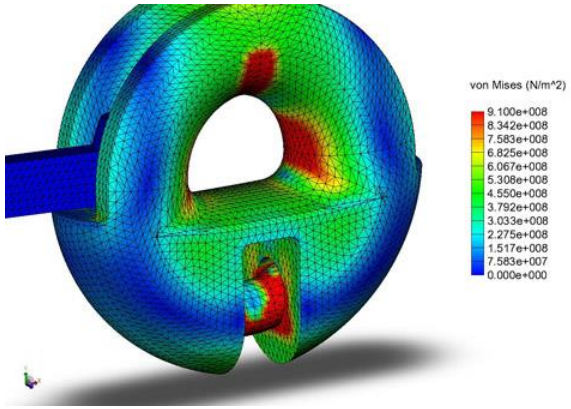


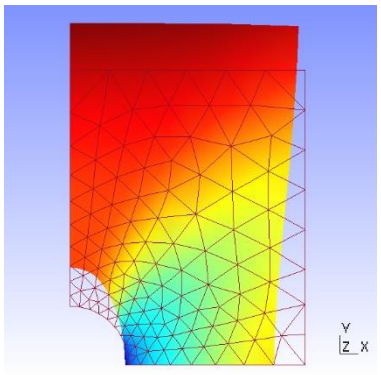


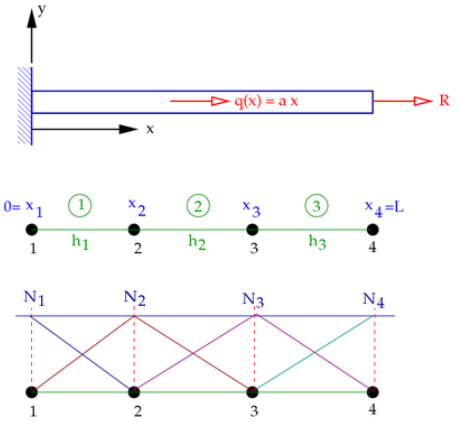




How do you understand the name finite element method − according to the related web images?



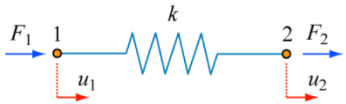




2.1 Weak form − 1D model problem

Introduction. The simplistic form of the finite element method is identical to the classical *direct stiffness method* typically used for solving spring systems or truss systems consisting of hinged extension-compression bars.

Let us consider the simplest spring system of one spring and its two end point dispacements.

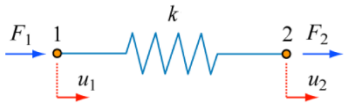


2.1 Weak form − 1D model problem

Introduction. The simplistic form of the finite element method is identical to the classical *direct stiffness method* typically used for solving spring systems or truss systems consisting of hinged extension-compression bars.

Let us consider the simplest spring system of one spring and its two end point dispacements. The force balances (statics) for the end points read as

$$\begin{cases} F_1 = k(u_1 - u_2) = ku_1 - ku_2 \\ F_2 = k(u_2 - u_1) = -ku_1 + ku_2 \end{cases} \Leftrightarrow \begin{bmatrix} k & -k \\ -k & k \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$$

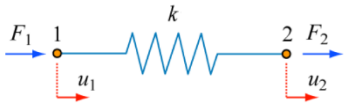


2.1 Weak form − 1D model problem

Introduction. The simplistic form of the finite element method is identical to the classical *direct stiffness method* typically used for solving spring systems or truss systems consisting of hinged extension-compression bars.

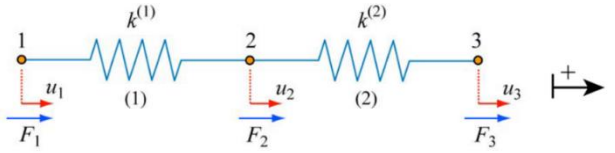
Let us consider the simplest spring system of one spring and its two end point dispacements. The force balances (statics) for the end points read as

$$\begin{cases} F_1 = k(u_1 - u_2) = ku_1 - ku_2 \\ F_2 = k(u_2 - u_1) = -ku_1 + ku_2 \end{cases} \Leftrightarrow \begin{bmatrix} k & -k \\ -k & k \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$$



For a two pring system, the force balance reads as

$$\begin{bmatrix} k^{(1)} & -k^{(1)} & 0 \\ -k^{(1)} & k^{(1)} + k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}$$

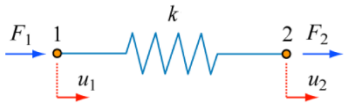


2.1 Weak form − 1D model problem

Introduction. The simplistic form of the finite element method is identical to the classical *direct stiffness method* typically used for solving spring systems or truss systems consisting of hinged extension-compression bars.

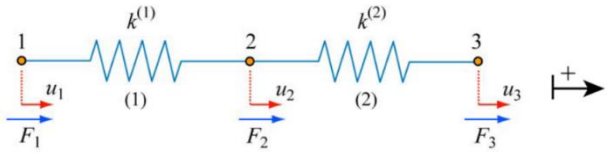
Let us consider the simplest spring system of one spring and its two end point dispacements. The force balances (statics) for the end points read as

$$\begin{cases} F_1 = k(u_1 - u_2) = ku_1 - ku_2 \\ F_2 = k(u_2 - u_1) = -ku_1 + ku_2 \end{cases} \Leftrightarrow \begin{bmatrix} k & -k \\ -k & k \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}$$



For a two pring system, the force balance reads as

$$\begin{bmatrix} k^{(1)} & -k^{(1)} & 0 \\ -k^{(1)} & k^{(1)} + k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix}$$

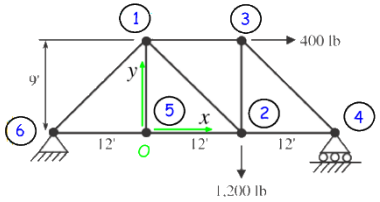


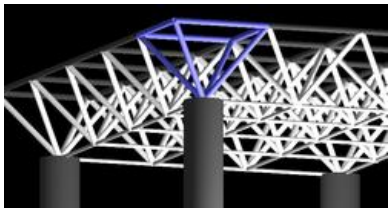
Remark. In both cases, the system matrix is *symmetric* and has a *band structure*.
Remark. In the system of multiple springs, the stiffness matrix is decomposed of system matrices similar to the stiffness matrix of the one spring system.

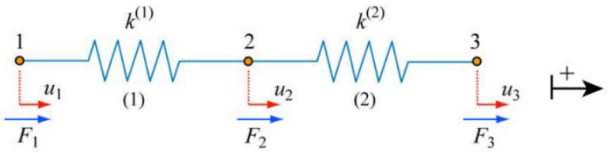
2.1 Weak form – 1D model problem

Remark. In a similar way, one can write the global force balance of a general spring system in which the spring are not necessarily connected as a planar sequence (line): more than two springs can be connected to one end point and the spring system can have a spatial configuration.

Remark. Analogously, one can write the force balance of a truss system: stiffness k is simply identified as the axial stiffness of a truss member (rod/bar): $k = EA$.







Remark. Displacement *boundary conditions* (such as $u_1 = 0$) remove *degrees of freedom* (variables to be solved) from the system:

$$\begin{bmatrix} k^{(1)} & -k^{(1)} & 0 \\ -k^{(1)} & k^{(1)}+k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & k^{(1)}+k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 0 \\ F_2 \\ F_3 \end{bmatrix}$$

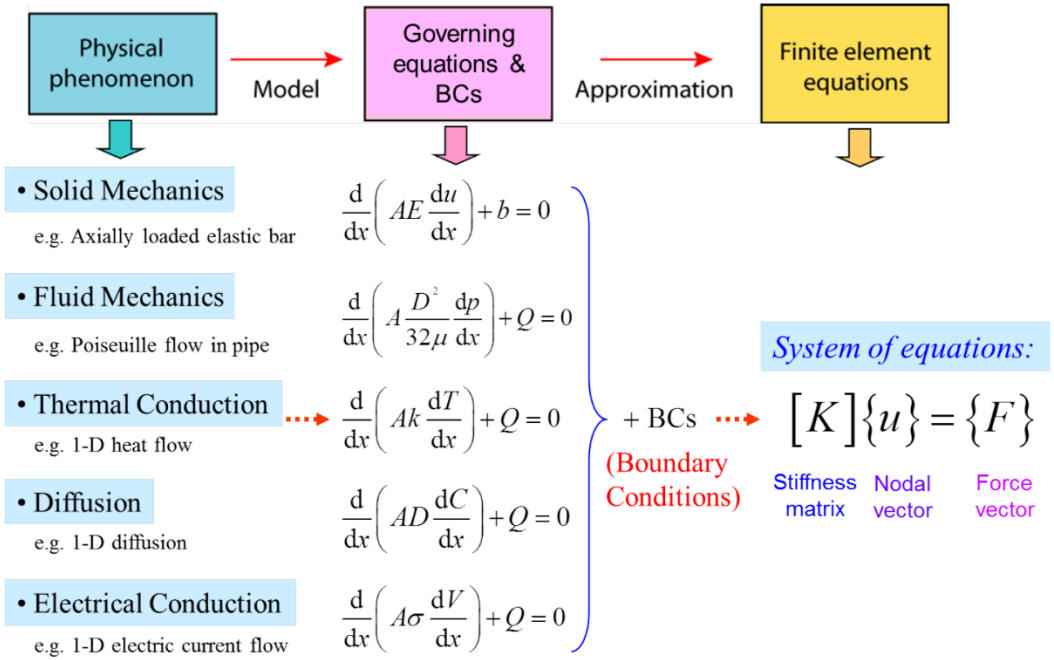
2.1 Weak form − 1D model problem

The standard form of the system equations is a matrix–vector equation (system of coupled equations) ***Ku*** = ***F***, where ***K*** is called *stiffness matrix*, ***u*** denotes *displacement vector* and ***F*** stands for *force vector.*

2.1 Weak form – 1D model problem

The standard form of the system equations is a matrix–vector equation (system of coupled equations) ***Ku*** = ***F***, where ***K*** is called *stiffness matrix*, ***u*** denotes *displacement vector* and ***F*** stands for *force vector.*

More generally, the finite element method is a method for solving *partial differential equations* written in a *weak form*, or energy form (cf. the *principle of virtual work*).



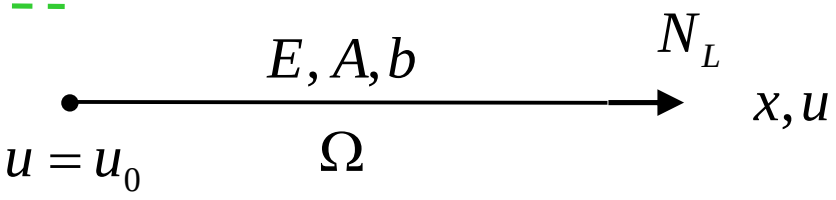
2.1 Weak form – 1D model problem

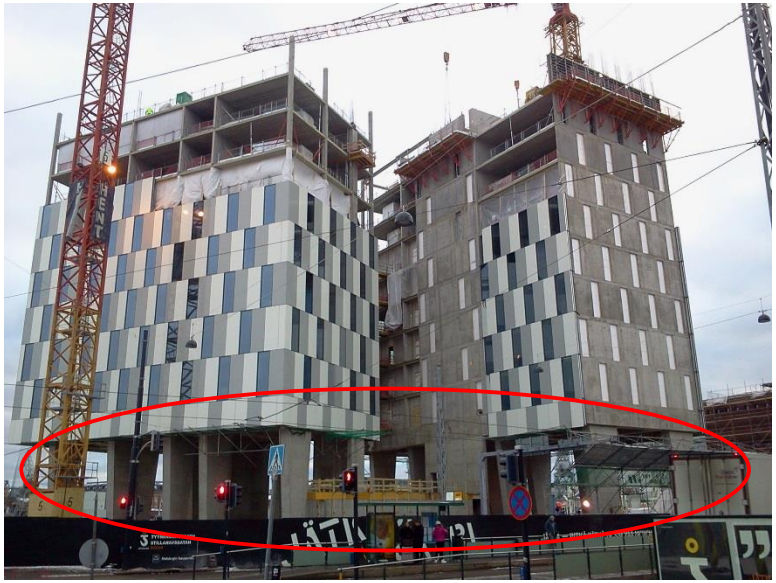
Strong form. The *differential equation* and *boundary conditions* for an *elastic bar/rod/column in tension/compression* read as follows: Find u such that

$$(1\text{-DE}) \quad -\left(EAu'\right)'(x) = b(x), \quad 0 < x < L$$

$$(2\text{-eBC}) \quad u(0) = u_0$$

$$(3\text{-nBC}) \quad (EAu')(L) = N_L$$





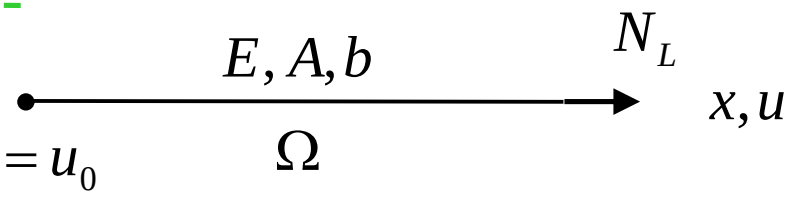
2.1 Weak form – 1D model problem

Strong form. The *differential equation* and *boundary conditions* for an *elastic bar/rod/column in tension/compression* read as follows: Find u such that

$$(1\text{-DE}) \quad -\left(EAu'\right)'(x) = b(x), \quad 0 < x < L$$

$$(2\text{-eBC}) \quad u(0) = u_0$$

$$(3\text{-nBC}) \quad (EAu')(L) = N_L$$



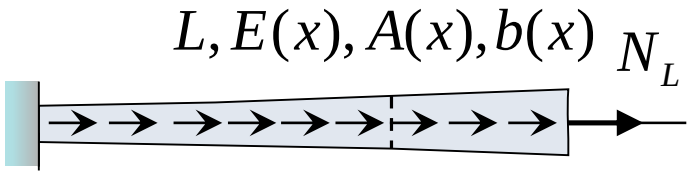
u axial displacement (unknown function)

E Young's modulus (given material data)

A cross-sectional area (given geometrical data)

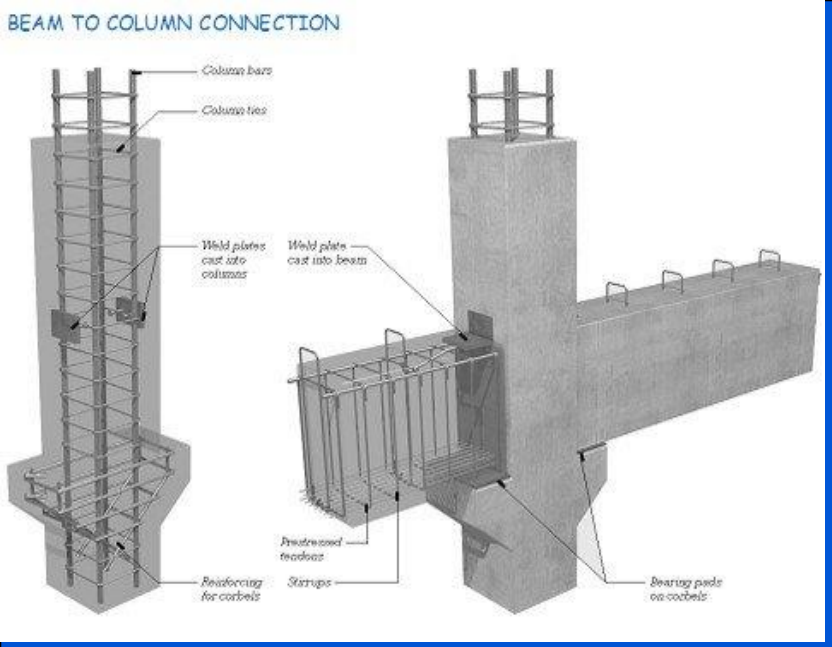
b axial body load (given loading data)

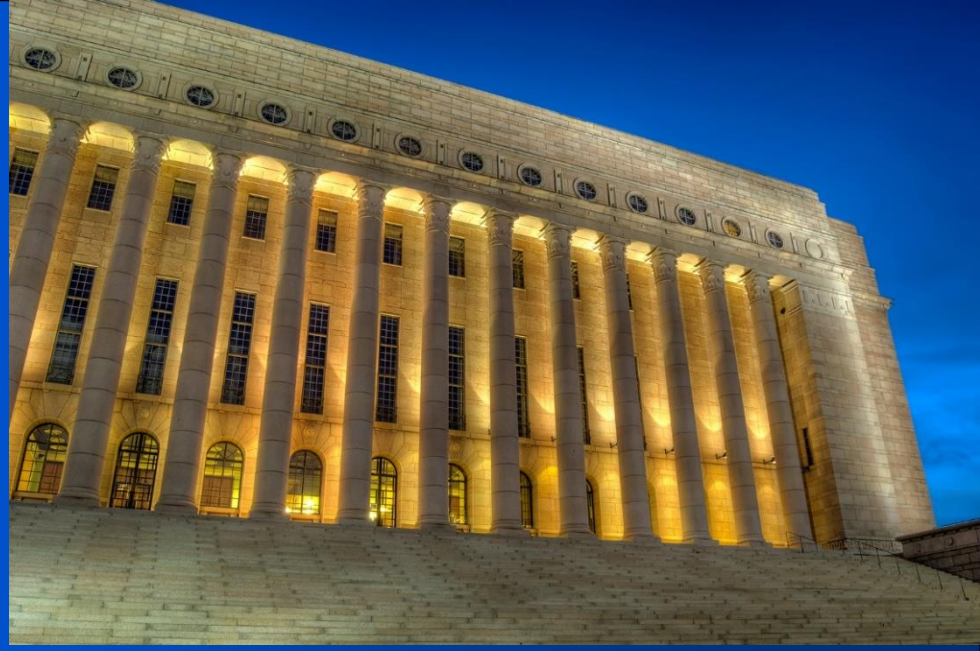
L length (given geometrical data)



u_0 axial end point displacement (given essential/geometric boundary data)

N_L axial end point force (given natural/force boundary data).



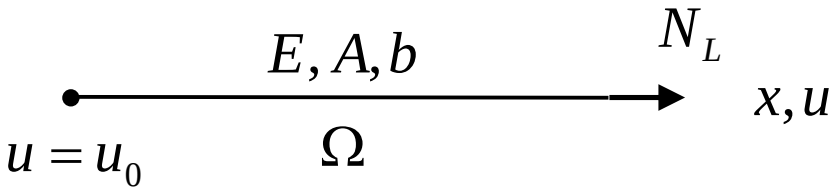


FURTHER MOTIVATION …

2.1 Weak form – 1D model problem

Weak form. Find u such that it satisfies $u(0) = u_0$ and

$$\int_0^L EAu'\,v'\,\mathrm{d}x = N_L\,v(L) + \int_0^L b\,v\,\mathrm{d}x,$$



for all test functions v satisfying $v(0) = 0$.

Remark. Why do we formulate the problem in a weak, or variational, form?

The finite element method (FEM) is based on the weak form which actually present the problem in a form of energy balance:

1. the left hand side corresponds to *strain energy*
   (the derivative of the axial displacement is the axial strain: $\varepsilon(x) = u'(x)$);
2. the right hand side corresponds to *loading energy*
   (work done by a force equals to the product of the force and the corresponding displacement).

2.1 Weak form − 1D model problem

Other 1D problems of the same form. The bar problem serves as a model problem for many other physical phenomena as well. Let us formulate the **Problem** as a pseudo-code ”in words” (cf. the formulation of the bar problem and the table below):

(1) – (**Data**(x) * (**I variable**)')'(x) = **Load**(x), $0 < x < L$

(2) (**I variable**)(0) = **I variable**_value_at_0

(3) (**II variable**)(L) = **II variable**_value_at_L

Remark. In (2), one could set a given value for **II variable**, or in (3) for **I variable**.

Problem	I variable	Data	Load	II variable
Axially loaded bar	displacement	EA	extensional force	force
Torsionally loaded bar	angle	GJ	torsional moment	torque
Seepage in soils	head (pressure)	k	infiltration	velocity
Heat diffusion	temperature	k	heat generation	heat flux
Electrostatics	electric potential	ε	charge density	electric flux

HOW TO DERIVE THE WEAK FORM?

extra material

2.1 Weak form – 1D model problem

0. Start from the differential equation (1) and use the boundary conditions (2) and (3):

$$(1) \quad -\left(EAu'\right)'(x) = b(x) \quad 0 < x < L$$

$$(2) \quad u(0) = u_0$$

$$(3) \quad (EAu')(L) = N_L$$

What shall we do
with the differential equation and the boundary conditions
– one page with a few lines is enough –
in order to reach the integral form below?

Do some problem solving team work in pairs
for a few minutes…

$$\Rightarrow \quad (1) \quad \int_0^L (EAu')(x)\, v'(x) \mathrm{d}x = N_L\, v(L) + \int_0^L b(x)\, v(x)\, \mathrm{d}x$$

$$(2) \quad u(0) = u_0$$

2.1 Weak form − 1D model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\left(EAu'\right)'(x) = b(x) \quad \Rightarrow \quad -\left(EAu'\right)'(x)\, v(x) = b(x)\, v(x), \quad 0 < x < L$$

2.1 Weak form − 1D model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\left(EAu'\right)'(x) = b(x) \quad \Rightarrow \quad -\left(EAu'\right)'(x)\, v(x) = b(x)\, v(x), \quad 0 < x < L$$

2. Integrate over the *domain* (interval):

$$\Rightarrow \quad -\int_0^L (EAu')'(x)\, v(x)\, \mathrm{d}x = \int_0^L b(x)\, v(x)\, \mathrm{d}x$$

2.1 Weak form − 1D model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\left(EAu'\right)'(x) = b(x) \quad \Rightarrow \quad -\left(EAu'\right)'(x)\, v(x) = b(x)\, v(x), \quad 0 < x < L$$

2. Integrate over the *domain* (interval):

$$\Rightarrow \quad -\int_0^L (EAu')'(x)\, v(x)\, \mathrm{d}x = \int_0^L b(x)\, v(x)\, \mathrm{d}x$$

3. Integrate by parts (the left hand side) for moving one derivative from *u* to *v*:

$$\Rightarrow \quad -(EAu')(L)\, v(L) \; + \; (EAu')(0)\, v(0) \; + \; \int_0^L EAu'\, v'\, \mathrm{d}x = \int_0^L b\, v\, \mathrm{d}x$$

2.1 Weak form – 1D model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\left(EAu'\right)'(x) = b(x) \quad \Rightarrow \quad -\left(EAu'\right)'(x)\, v(x) = b(x)\, v(x), \quad 0 < x < L$$

2. Integrate over the *domain* (interval):

$$\Rightarrow \quad -\int_0^L (EAu')'(x)\, v(x)\, \mathrm{d}x = \int_0^L b(x)\, v(x)\, \mathrm{d}x$$

3. Integrate by parts (the left hand side) for moving one derivative from *u* to *v*:

$$\Rightarrow \quad -(EAu')(L)\, v(L) \; + \; (EAu')(0)\, v(0) \; + \; \int_0^L EAu'\, v'\, \mathrm{d}x = \int_0^L b\, v\, \mathrm{d}x$$

4. Utilize the *natural boundary condition* (3): $(EAu')(L) = N_L$

2.1 Weak form – 1D model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\left(EAu'\right)'(x) = b(x) \quad \Rightarrow \quad -\left(EAu'\right)'(x)\, v(x) = b(x)\, v(x), \quad 0 < x < L$$

2. Integrate over the *domain* (interval):

$$\Rightarrow \quad -\int_0^L (EAu')'(x)\, v(x)\, \mathrm{d}x = \int_0^L b(x)\, v(x)\, \mathrm{d}x$$

3. Integrate by parts (the left hand side) for moving one derivative from *u* to *v*:

$$\Rightarrow \quad -(EAu')(L)\, v(L) \; + \; (EAu')(0)\, v(0) \; + \; \int_0^L EAu'\, v'\, \mathrm{d}x = \int_0^L b\, v\, \mathrm{d}x$$

4. Utilize the *natural boundary condition* (3): $(EAu')(L) = N_L$

5. Set a zero *essential boundary condition* (2) for the test function: $v(0) = 0$

$$\Rightarrow \quad \int_0^L EAu'\, v'\, \mathrm{d}x = N_L\, v(L) \; + \; \int_0^L b\, v\, \mathrm{d}x$$

2.1 Weak form − 1D model problem

Weak form. Find u such that it satisfies $u(0) = u_0$ and

$$\int_0^L EAu'\, v'\, \mathrm{d}x = N_L\, v(L) + \int_0^L b\, v\, \mathrm{d}x,$$

for all v satisfying $v(0) = 0$.

2.1 Weak form − 1D model problem

Weak form. Find u such that it satisfies $u(0) = u_0$ and

$$\int_0^L EAu'\, v'\, \mathrm{d}x = N_L\, v(L) + \int_0^L b\, v\, \mathrm{d}x,$$

for all v satisfying $v(0) = 0$.

Remark. Note that the solution and the test function, respectively, have to satisfy the boundary conditions $u(0) = u_0,\ v(0) = 0$ and the *regularity conditions*

$$\int_0^L (u')^2\, \mathrm{d}x < \infty, \quad \int_0^L (v')^2\, \mathrm{d}x < \infty.$$

Then they are called *kinematically admissible.*

2.1 Weak form − 1D model problem

Weak form. Find u such that it satisfies $u(0) = u_0$ and

$$\int_0^L EAu'\, v'\, \mathrm{d}x = N_L\, v(L) + \int_0^L b\, v\, \mathrm{d}x,$$

for all v satisfying $v(0) = 0$.

Remark. Note that the solution and the test function, respectively, have to satisfy the boundary conditions $u(0) = u_0,\ v(0) = 0$ and the *regularity conditions*

$$\int_0^L (u')^2\, \mathrm{d}x < \infty, \quad \int_0^L (v')^2\, \mathrm{d}x < \infty.$$

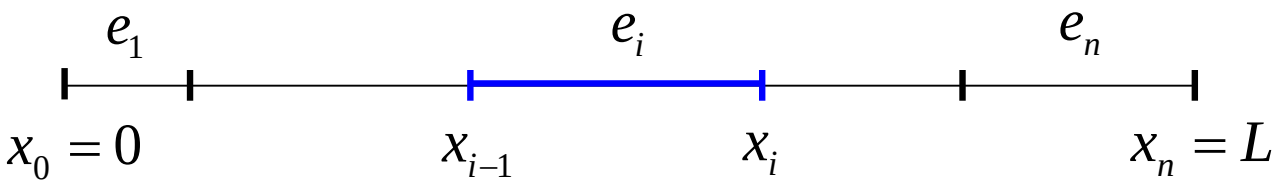
Then they are called *kinematically admissible.*

Remark. Starting from the weak form we could correspondingly derive the strong form (integrating by parts ”backwards”).

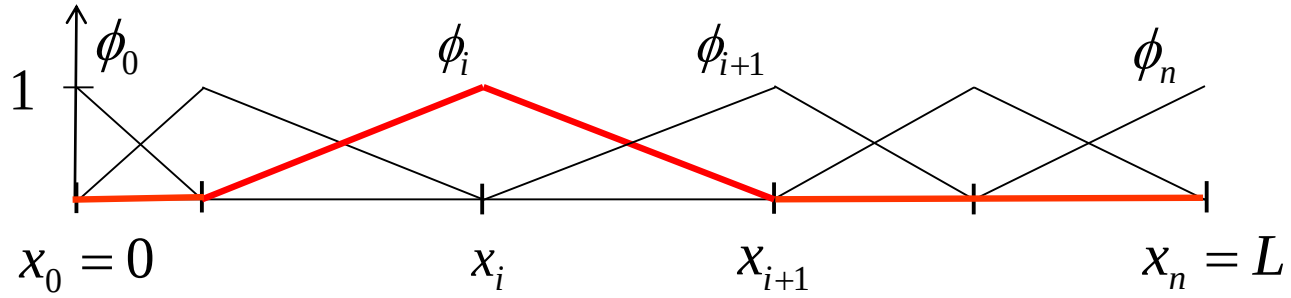
BACK
TO
basic material

2.2 1D finite element method – model problem

Finite element approximation for the 1D bar in tension/compression problem. Let us divide the solution interval (domain) into n subintervals e_i (*elements*) with *nodes* x_i and *element size* $h_i = x_i - x_{i-1}$:



In each element, the displacement field is approximated by *nodal values* of the displacement and in between the nodes by (linear) polynomial *basis functions* which are now (in a 1D problem) functions of the *x*-coordinate:



2.2 1D finite element method − model problem

This finally results in an equation system

$$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$$

with the *stiffness matrix* ***K*** (computable for *i,j = 1, …, n*), *force vector* ***f*** (computable for *i = 1, …, n*) and the *displacement vector* ***d*** (unknown for *i = 1, …, n*):

$$\boldsymbol{K} = \left[K_{ij}\right], \quad K_{ij} = \int_0^L EA\phi_i'\,\phi_j'\,\mathrm{d}x,$$

$$\boldsymbol{f} = \left[f_i\right], \quad f_i = \int_0^L b\,\phi_i\,\mathrm{d}x + N_L\,\phi_i(L) - u_0\int_0^L \frac{\mathrm{d}\phi_i}{\mathrm{d}x}\,AE\,\frac{\mathrm{d}\phi_0}{\mathrm{d}x}\,\mathrm{d}x, \quad \boldsymbol{d} = \left[d_j\right]$$

Remark. The stiffness matrix is (very often) *symmetric* (due to derivative orders) and its entries are concentrated in a narrow diagonal band forming a *band matrix* (due to local trial and test functions). These features can can be utilized in computer impelementation − implying small amounts of memory needs and quick processing.

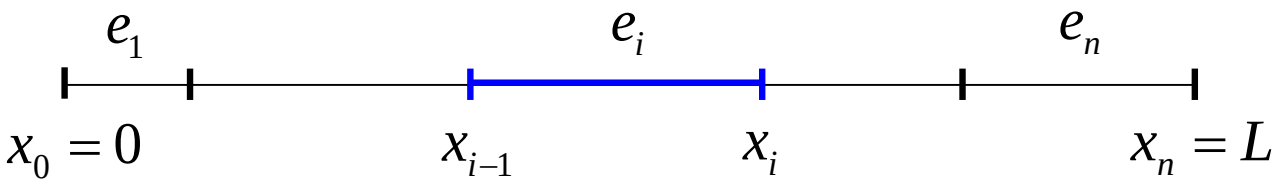
Remark. Test and trial functions have to be (only) once locally differentiable (and will be then integrated over the domain) and (only) locally evaluable on the boundary.

HOW
TO
DERIVE THE FINITE ELEMENT SYSTEM?

extra material

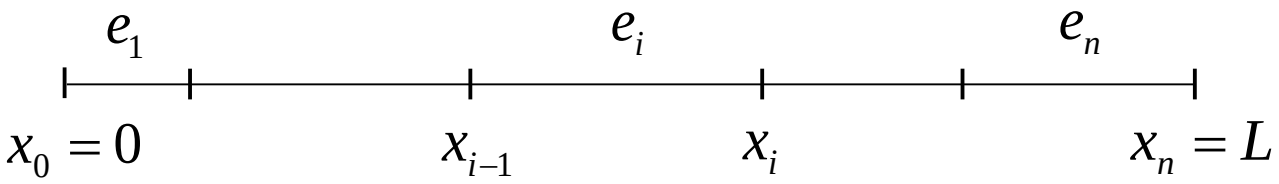
2.2 1D finite element method – model problem

1. Divide the solution interval (domain) into n subintervals e_i (*elements*) with *nodes* x_i and the *element size* $h_i = x_i - x_{i-1}$:



2.2 1D finite element method – model problem

1. Divide the solution interval (domain) into n subintervals e_i (*elements*) with *nodes* x_i and the *element size* $h_i = x_i - x_{i-1}$:

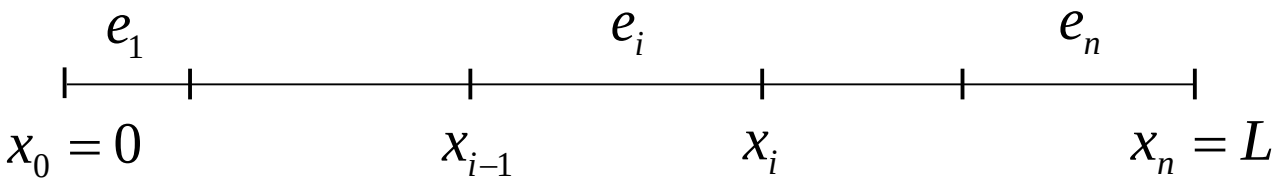


2. Choose a *trial function* for the finite element approximation as a sum

$$u_h(x) = \phi_0(x)d_0 + \phi_1(x)d_1 + \cdots + \phi_n(x)d_n = \sum_{j=0}^{n} \phi_j(x)d_j$$

2.2 1D finite element method – model problem

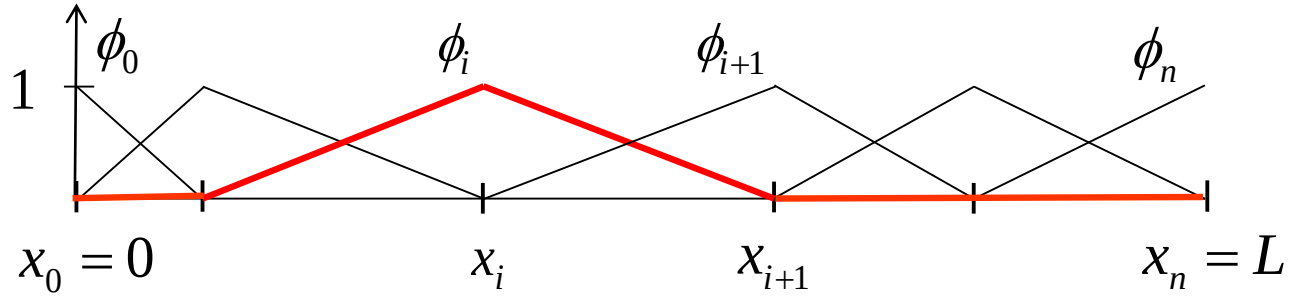
1. Divide the solution interval (domain) into n subintervals e_i (*elements*) with *nodes* x_i and the *element size* $h_i = x_i - x_{i-1}$:



2. Choose a *trial function* for the finite element approximation as a sum

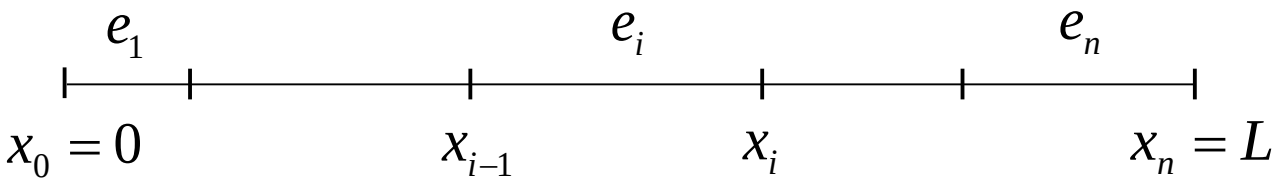
$$u_h(x) = \phi_0(x)d_0 + \phi_1(x)d_1 + \cdots + \phi_n(x)d_n = \sum_{j=0}^{n} \phi_j(x)d_j$$

with suitable *local basis functions* ϕ_i of some polynomial order (now linear)



2.2 1D finite element method – model problem

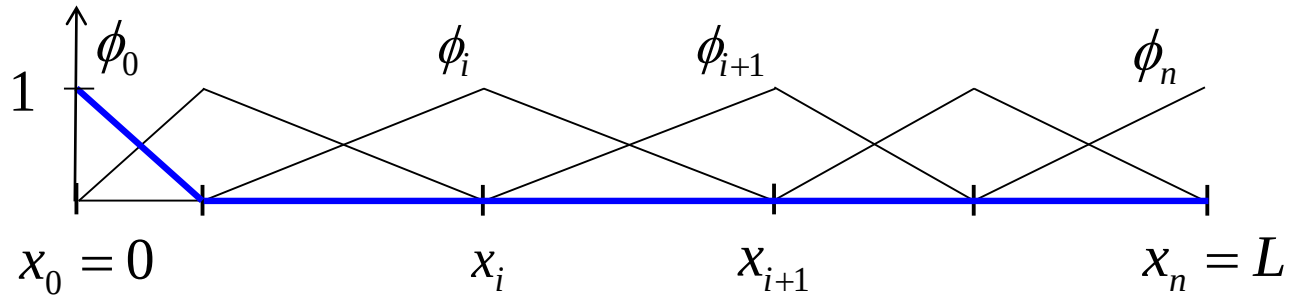
1. Divide the solution interval (domain) into n subintervals e_i (*elements*) with *nodes* x_i and the *element size* $h_i = x_i - x_{i-1}$:



2. Choose a *trial function* for the finite element approximation as a sum

$$u_h(x) = \phi_0(x)d_0 + \phi_1(x)d_1 + \cdots + \phi_n(x)d_n = \sum_{j=0}^{n} \phi_j(x)d_j$$

with suitable *local basis functions* ϕ_i of some polynomial order (now linear)



$$\phi_i(x_j) = \delta_{ij}$$

The unknown scalar values $d_i = u_h(x_i)$ are called the *degrees of freedom.*

2.2 1D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$u_0 = u_h(0) = \phi_0(0)d_0 + \phi_1(0)d_1 + \cdots + \phi_n(0)d_n = d_0$$

2.2 1D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$u_0 = u_h(0) = \phi_0(0)d_0 + \phi_1(0)d_1 + \cdots + \phi_n(0)d_n = d_0$$

3. Choose a test function of a similar form (*Galerkin method*) with the corresponding condition:

$$v(x) = \phi_0(x)c_0 + \phi_1(x)c_1 + \cdots + \phi_n(x)c_n = \sum_{i=0}^{n} \phi_i(x)c_i$$

$$0 = v(0) \Rightarrow c_0 = 0$$

2.2 1D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$u_0 = u_h(0) = \phi_0(0)d_0 + \phi_1(0)d_1 + \cdots + \phi_n(0)d_n = d_0$$

3. Choose a test function of a similar form (*Galerkin method*) with the corresponding condition:

$$v(x) = \phi_0(x)c_0 + \phi_1(x)c_1 + \cdots + \phi_n(x)c_n = \sum_{i=0}^{n} \phi_i(x)c_i$$

$$0 = v(0) \Rightarrow c_0 = 0$$

4. Insert the functions – trial and test – into the weak form:

$$\int_0^L EAu_h' v' \mathrm{d}x = N_L v(L) + \int_0^L b\, v \,\mathrm{d}x$$

2.2 1D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$u_0 = u_h(0) = \phi_0(0)d_0 + \phi_1(0)d_1 + \cdots + \phi_n(0)d_n = d_0$$

3. Choose a test function of a similar form (*Galerkin method*) with the corresponding condition:

$$v(x) = \phi_0(x)c_0 + \phi_1(x)c_1 + \cdots + \phi_n(x)c_n = \sum_{i=0}^{n} \phi_i(x)c_i$$

$$0 = v(0) \Rightarrow c_0 = 0$$

4. Insert the functions – trial and test – into the weak form:

$$\int_0^L EAu_h' v' \mathrm{d}x = N_L v(L) + \int_0^L b\, v \,\mathrm{d}x$$

$$\Rightarrow \int_0^L EA \sum_{j=0}^{n} \phi_j' d_j \sum_{i=0}^{n} \phi_i' c_i \,\mathrm{d}x = N_L \sum_{i=0}^{n} \phi_i(L)c_i + \int_0^L b \sum_{i=0}^{n} \phi_i\, c_i \,\mathrm{d}x$$

2.2 1D finite element method – model problem

This results in a simple equation system

$$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$$

with the *stiffness matrix* ***K*** (computable for *i,j = 1, …, n*), *force vector* ***f*** (computable for *i = 1, …, n*) and the *displacement vector* ***d*** (unknown for *i = 1, …, n*):

$$\boldsymbol{K} = \left[K_{ij}\right], \quad K_{ij} = \int_0^L EA\phi_i' \phi_j' \mathrm{d}x,$$

$$\boldsymbol{d} = \left[d_i\right], \quad \boldsymbol{f} = \left[f_i\right], \quad f_i = \int_0^L b\,\phi_i \,\mathrm{d}x + N_L\,\phi_i(L) - u_0 \int_0^L \frac{\mathrm{d}\phi_i}{\mathrm{d}x}\,AE\,\frac{\mathrm{d}\phi_0}{\mathrm{d}x}\,\mathrm{d}x.$$

2.2 1D finite element method – model problem

This results in a simple equation system

$$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$$

with the *stiffness matrix* ***K*** (computable for *i,j = 1, …, n*), *force vector* ***f*** (computable for *i = 1, …, n*) and the *displacement vector* ***d*** (unknown for *i = 1, …, n*):

$$\boldsymbol{K} = \left[K_{ij}\right], \quad K_{ij} = \int_0^L EA\phi_i' \phi_j' \mathrm{d}x,$$

$$\boldsymbol{d} = \left[d_i\right], \quad \boldsymbol{f} = \left[f_i\right], \quad f_i = \int_0^L b\,\phi_i \,\mathrm{d}x + N_L\,\phi_i(L) - u_0 \int_0^L \frac{\mathrm{d}\phi_i}{\mathrm{d}x}\,AE\,\frac{\mathrm{d}\phi_0}{\mathrm{d}x}\,\mathrm{d}x.$$

2.2 1D finite element method – model problem

5. Use an appropriate *solver* for the equation system:

$$\boldsymbol{d} = \boldsymbol{K}^{-1}\,\boldsymbol{f} \quad \Rightarrow \quad u_h(x) = \sum_{j=1}^{n} \phi_j(x) d_j$$

2.2 1D finite element method – model problem

5. Use an appropriate *solver* for the equation system:

$$\boldsymbol{d} = \boldsymbol{K}^{-1}\,\boldsymbol{f} \quad \Rightarrow \quad u_h(x) = \sum_{j=1}^{n} \phi_j(x) d_j$$

6. Recover (and *postprocess)* the stress quantities and visualize:

$$\Rightarrow \quad N_h(x) = (EAu_h)'(x) = \sum_{j=1}^{n} (EA\phi_j)'(x) d_j \quad \Rightarrow \quad \sigma_h(x) = \frac{N_h(x)}{A(x)}$$

2.2 1D finite element method − model problem

5. Use an appropriate *solver* for the equation system:

$$\boldsymbol{d} = \boldsymbol{K}^{-1}\,\boldsymbol{f} \quad \Rightarrow \quad u_h(x) = \sum_{j=1}^{n} \phi_j(x) d_j$$

6. Recover (and *postprocess)* the stress quantities and visualize:

$$\Rightarrow \quad N_h(x) = (EAu_h)'(x) = \sum_{j=1}^{n} (EA\phi_j)'(x) d_j \quad \Rightarrow \quad \sigma_h(x) = \frac{N_h(x)}{A(x)}$$

7. Evalute possible *error indicators*, change the *discretization* (steps 1−4) ... rerun ...

2.2 1D finite element method – model problem

5. Use an appropriate *solver* for the equation system:

$$\boldsymbol{d} = \boldsymbol{K}^{-1}\,\boldsymbol{f} \quad \Rightarrow \quad u_h(x) = \sum_{j=1}^{n} \phi_j(x) d_j$$

6. Recover (and *postprocess)* the stress quantities and visualize:

$$\Rightarrow \quad N_h(x) = (EAu_h)'(x) = \sum_{j=1}^{n} (EA\phi_j)'(x) d_j \quad \Rightarrow \quad \sigma_h(x) = \frac{N_h(x)}{A(x)}$$

7. Evalute possible *error indicators*, change the *discretization* (steps 1–4) ... rerun ...

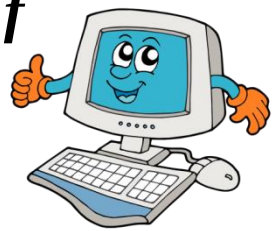
Remark. Steps 1–7 are automated – by means of mathematics and programming:

1. Elements e_i

2–3. Basis functions ϕ_i

4. Matrix entries K_{ij}, f_i

$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$



$u_h(x)$

5. Equation solution

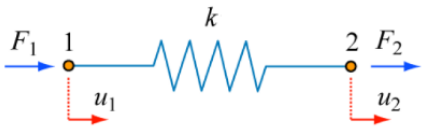
6. Visualization

7. Error evaluation

BACK
TO
basic material

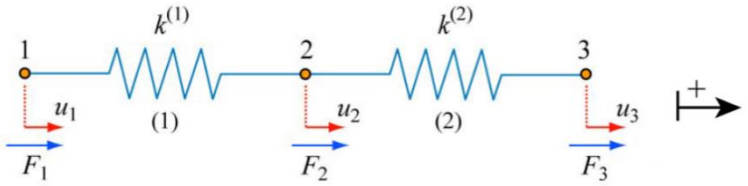
Home exercise 2.1

(i) Let us consider a spring system of one spring and its two end point dispacements and forces (see the figure below).



Derive the force balance equations (statics) regarding the end point displacements as a system of equations (in a matrix form ***Ax*** = ***b***).

(ii) Let us consider a two-spring system of two springs in series with three end point displacements and three end point forces (see the figure).



Derive the force balance equations (statics) regarding the end point displacements as a system of equations (in a matrix form ***Ax*** = ***b***).

(iii) Do you recognize any general rule applying for a system of n = 2, 3,... springs?

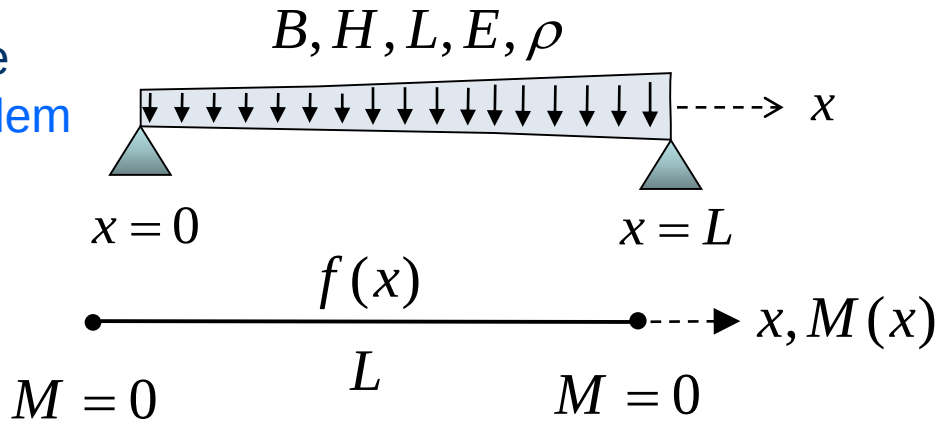
Home exercise 2.2

Let us consider a vertically loaded thin 2D/3D *beam* structure (made of steel) with the following dimensions: $B = 4\ \text{cm}, H = 2\ \text{cm}, L = 1\ \text{m}.$

The *bending moment* of the beam can be solved from the strong form of a 1D problem (see Home exercises 1.1):

$$-M''(x) = f(x), \quad 0 < x < L$$

$$M(0) = 0, M(L) = 0$$



Formulate the *weak form* of the problem.

Hint: Compare the strong form above to the strong form of the rod problem in Section 2.1 and then imitate the weak form of the rod problem (or repeat steps 1–5 of the extra material of Section 2.1).

Note that in this case the strong form above has two essential boundary conditions instead of one essential and one natural.

3 Basic 2D and 3D finite element methods

Fundamentals of Finite Element Analysis

Lecture 1

1. Modelling principles and boundary value problems in engineering sciences
   - Home exercise 1.1
   - Computer exercise 1.1 – Matlab
2. Energy methods and basic 1D finite element methods
   - bars/rods, beams, heat diffusion, seepage, electrostatics
   - Home exercise 2.1
   - Home exercise 2.2

Lecture 2

3. Basic 2D and 3D finite element methods
   - heat diffusion, seepage
   - Computer exercise 3.1 – Matlab PDE Modeler
   - Computer exercise 3.2 – Matlab PDE Modeler
4. Mesh refinements, adaptive and isogeometric methods
   - Computer exercise 4.1 – Matlab PDE Modeler (VOLUNTARY)

3 Basic 2D and 3D finite element methods

Contents

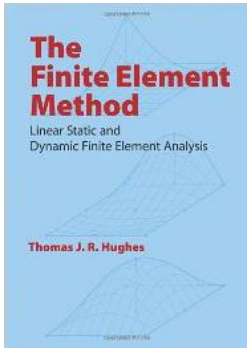
1. *2D (/3D) weak form* (based on the principle of virtual work)
2. *2D (/3D) finite element method* (based on the weak form)

Learning outcome

A. *Understanding of the main principles* behind the 2D (/3D) finite element method

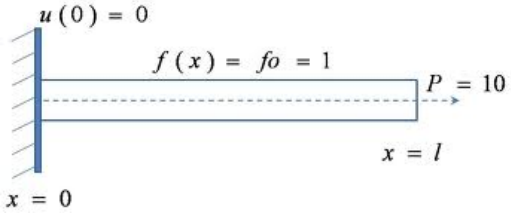
B. *Ability to apply* the finite element method for 2D (/3D) model problems

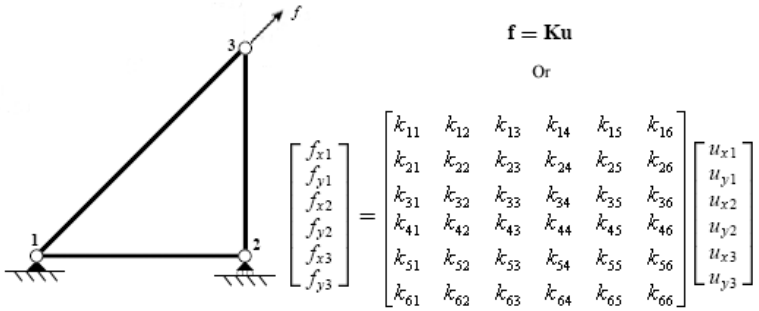
References

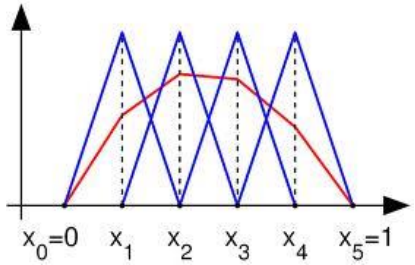


Text book: chapters 2.1−4

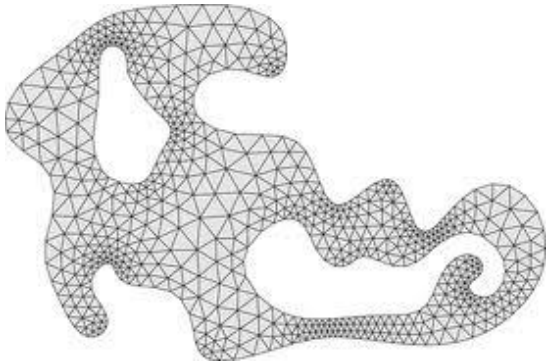
3.0 Generalizing 1D finite element methods to 2D (/3D) intuitively

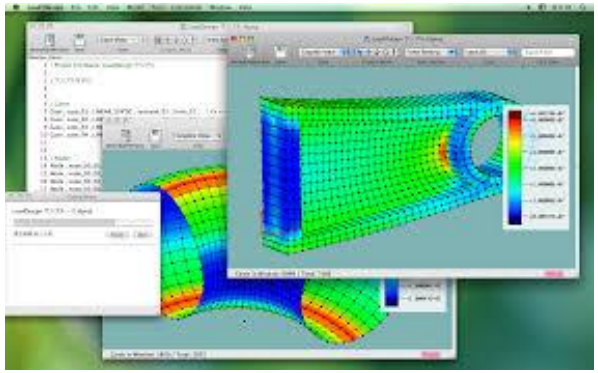


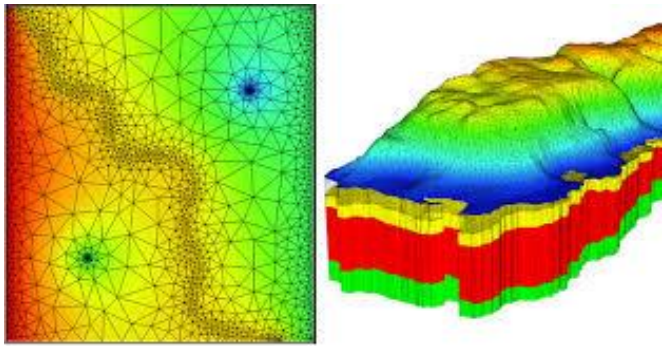




How straightforward it might be to generalize
1D finite element methods for 2D or 3D problems?







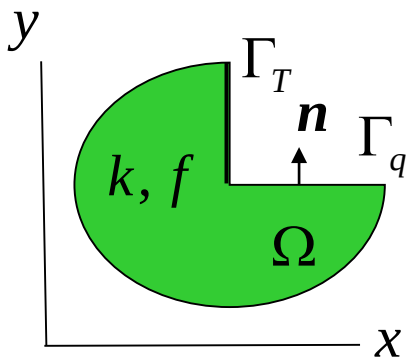
3.1 2D weak form – model problem

Strong form. The *partial differential equation* and *boundary conditions* for *stationary isotropic heat diffusion/conduction* read as follows: Find $T = T(x, y)$ such that

$$(1) \quad -\nabla \cdot (k(x,y)\nabla T(x,y)) = f(x,y), \quad (x,y) \in \Omega$$

$$(2a) \quad T(x,y) = T_0(x,y), \quad (x,y) \in \Gamma_T$$

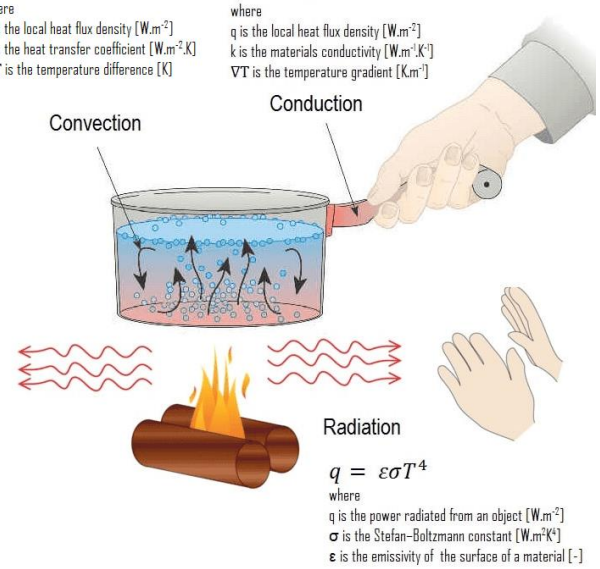
$$(2b) \quad \boldsymbol{q}(x,y) \cdot \boldsymbol{n} = q_0(x,y), \quad (x,y) \in \Gamma_q$$



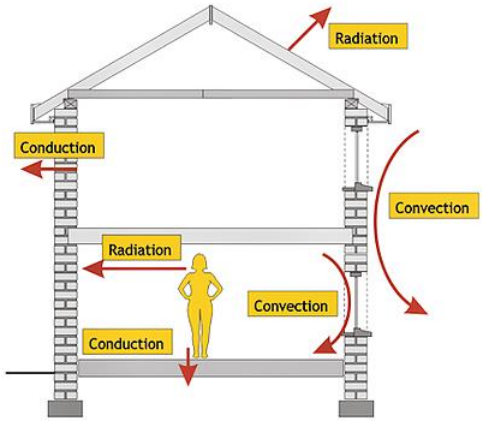
$$hA(T(x,y) - T_\infty(x,y))$$

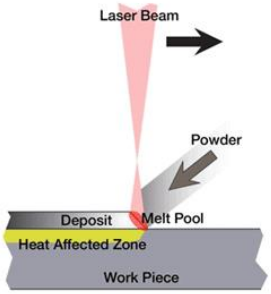
$q = h\Delta T$
where
q is the local heat flux density [W.m^{-2}]
h is the heat transfer coefficient [W.m^{-2}.K]
ΔT is the temperature difference [K]

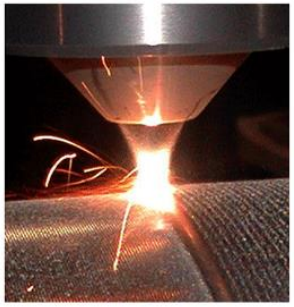
$q = -k\nabla T$
where
q is the local heat flux density [W.m^{-2}]
k is the materials conductivity [W.m^{-1}.K^{-1}]
∇T is the temperature gradient [K.m^{-1}]



$q = \varepsilon\sigma T^4$
where
q is the power radiated from an object [W.m^{-2}]
σ is the Stefan–Boltzmann constant [W.m^{2}K^4]
ε is the emissivity of the surface of a material [-]







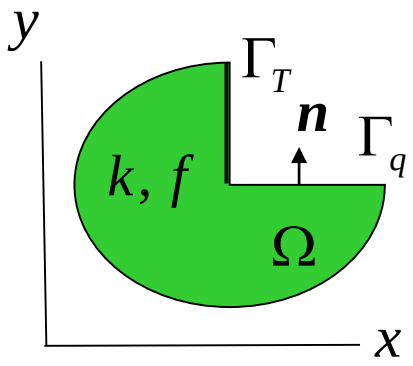
3.1 2D weak form − model problem

Strong form. The *partial differential equation* and *boundary conditions* for *stationary isotropic heat diffusion/conduction* read as follows: Find $T = T(x, y)$ such that

$$(1) \quad -\nabla \cdot (k(x,y)\nabla T(x,y)) = f(x,y), \quad (x,y) \in \Omega$$

$$(2a) \quad T(x,y) = T_0(x,y), \quad (x,y) \in \Gamma_T$$

$$(2b) \quad \boldsymbol{q}(x,y) \cdot \boldsymbol{n} = q_0(x,y), \quad (x,y) \in \Gamma_q$$



T temperature (unknown function)

k thermal conductivity (given material data)

f heat supply (given loading data), Ω domain (given geometrical data)

Γ_T boundary part for given temperature (given boundary data)

Γ_q boundary part for given heat flux (given boundary data)

T_0 temperature on the boundary (given essential, Dirichlet, boundary data)

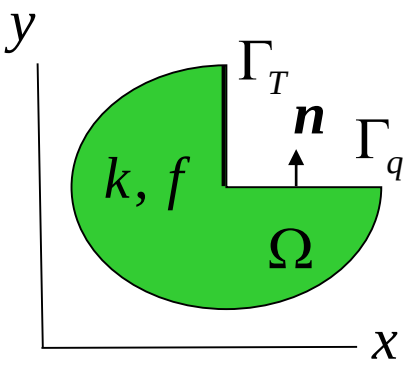
q_0 heat flux on the boundary (given natural, Neumann, boundary data).

3.1 2D weak form − model problem

Weak form. Find $T = T(x, y)$ such that it satisfies $T_{|\Gamma_T} = T_0$,

$$\int_\Omega (k\nabla T)\cdot\nabla\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A - \int_{\Gamma_q} q_0\hat{T}\,\mathrm{ds}$$

for all $\hat{T} = \hat{T}(x, y)$ satisfying $\hat{T}_{|\Gamma_T} = 0.$



Remark. Why do we formulate the problem in a weak, or variational, form?

The finite element method (FEM) is based on the weak form which actually present the problem in a form of energy balance.

Remark. *PDE* often stands for *partial differential equation*, *ODE* for *ordinary differential equation* (both of which can be linear or nonlinear). Differential equations have natural connections to *analysis* (evolved from *calculus*), *functional analysis* (when seen in abstract forms) and *numerical analysis* (when numerical solutions are in mind, e.g. *finite element methods*, *FEM*). For more information on classifications, see https://www.wolframalpha.com/examples/mathematics/.

HOW TO
DERIVE THE WEAK FORM?

extra material

3.1 2D weak form − model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\nabla \cdot (k\nabla T) = f \quad \Rightarrow \quad -\nabla \cdot (k\nabla T)\hat{T} = f\,\hat{T}, \quad \text{in } \Omega \subset R^2 = R \times R$$

3.1 2D weak form − model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\nabla \cdot (k\nabla T) = f \quad \Rightarrow \quad -\nabla \cdot (k\nabla T)\hat{T} = f\,\hat{T}, \quad \text{in } \Omega \subset R^2 = R \times R$$

2. Integrate over the *domain*:

$$\Rightarrow \quad -\int_\Omega \nabla \cdot (k\nabla T)\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A, \quad \mathrm{d}A = \mathrm{d}x\mathrm{d}y$$

3.1 2D weak form − model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\nabla\cdot(k\nabla T) = f \quad \Rightarrow \quad -\nabla\cdot(k\nabla T)\hat{T} = f\,\hat{T}, \quad \text{in } \Omega \subset R^2 = R\times R$$

2. Integrate over the *domain*:

$$\Rightarrow \quad -\int_\Omega \nabla\cdot(k\nabla T)\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A, \quad \mathrm{d}A = \mathrm{d}x\mathrm{d}y$$

3. Integrate by parts (the left hand side):

$$\Rightarrow \quad -\int_{\partial\Omega} \boldsymbol{n}\cdot(k\nabla T)\hat{T}\,\mathrm{d}s + \int_\Omega (k\nabla T)\cdot\nabla\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A$$

3.1 2D weak form − model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\nabla\cdot(k\nabla T) = f \quad \Rightarrow \quad -\nabla\cdot(k\nabla T)\hat{T} = f\,\hat{T}, \quad \text{in } \Omega \subset R^2 = R\times R$$

2. Integrate over the *domain*:

$$\Rightarrow \quad -\int_\Omega \nabla\cdot(k\nabla T)\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A, \quad \mathrm{d}A = \mathrm{d}x\mathrm{d}y$$

3. Integrate by parts (the left hand side):

$$\Rightarrow \quad -\int_{\partial\Omega} \boldsymbol{n}\cdot(k\nabla T)\hat{T}\,\mathrm{d}s + \int_\Omega (k\nabla T)\cdot\nabla\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A$$

4. Utilize the *natural boundary condition* (2b): $\boldsymbol{q}(x,y)\cdot\boldsymbol{n} = q_0(x,y), \quad (x,y)\in\Gamma_q$

3.1 2D weak form − model problem

1. Multiply the differential equation (1) by a (smooth) *test function* (specified later):

$$-\nabla\cdot(k\nabla T) = f \quad \Rightarrow \quad -\nabla\cdot(k\nabla T)\hat{T} = f\,\hat{T}, \quad \text{in } \Omega \subset R^2 = R\times R$$

2. Integrate over the *domain*:

$$\Rightarrow \quad -\int_\Omega \nabla\cdot(k\nabla T)\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A, \quad \mathrm{d}A = \mathrm{d}x\mathrm{d}y$$

3. Integrate by parts (the left hand side):

$$\Rightarrow \quad -\int_{\partial\Omega} \boldsymbol{n}\cdot(k\nabla T)\hat{T}\,\mathrm{d}s + \int_\Omega (k\nabla T)\cdot\nabla\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A$$

4. Utilize the *natural boundary condition* (2b): $\boldsymbol{q}(x,y)\cdot\boldsymbol{n} = q_0(x,y), \quad (x,y)\in\Gamma_q$

5. Set a zero *essential boundary condition* (2a) for the test function: $\hat{T} = 0$ on Γ_T

$$\Rightarrow \quad \int_\Omega (k\nabla T)\nabla\hat{T}\,\mathrm{d}A = \int_\Omega f\,\hat{T}\,\mathrm{d}A - \int_{\Gamma_q} q_0\hat{T}\,\mathrm{d}s$$

unknown — known — known

3.1 2D weak form − model problem

Weak form. Find $T = T(x, y)$ such that it satisfies $T_{|\Gamma_T} = T_0$,

$$\int_\Omega (k\nabla T) \cdot \nabla \hat{T} \, \mathrm{d}A = \int_\Omega f \, \hat{T} \, \mathrm{d}A - \int_{\Gamma_q} q_0 \hat{T} \, \mathrm{ds}$$

for all $\hat{T} = \hat{T}(x, y)$ satisfying $\hat{T}_{|\Gamma_T} = 0.$

3.1 2D weak form − model problem

Weak form. Find $T = T(x, y)$ such that it satisfies $T_{|\Gamma_T} = T_0$,

$$\int_\Omega (k\nabla T) \cdot \nabla \hat{T} \, \mathrm{d}A = \int_\Omega f \, \hat{T} \, \mathrm{d}A - \int_{\Gamma_q} q_0 \hat{T} \, \mathrm{ds}$$

for all $\hat{T} = \hat{T}(x, y)$ satisfying $\hat{T}_{|\Gamma_T} = 0.$

Remark. Note that the solution and the test function, respectively, have to satisfy the boundary conditions $T_{|\Gamma_T} = T_0$, $\hat{T}_{|\Gamma_T} = 0$ and the *regularity conditions*

$$\int_\Omega |\nabla T|^2 \mathrm{d}A < \infty, \quad \int_\Omega |\nabla \hat{T}|^2 \mathrm{d}A < \infty.$$

Then they are called *kinematically admissible.*

3.1 2D weak form − model problem

Weak form. Find $T = T(x, y)$ such that it satisfies $T_{|\Gamma_T} = T_0$,

$$\int_\Omega (k\nabla T) \cdot \nabla \hat{T} \, \mathrm{d}A = \int_\Omega f \, \hat{T} \, \mathrm{d}A - \int_{\Gamma_q} q_0 \hat{T} \, \mathrm{ds}$$

for all $\hat{T} = \hat{T}(x, y)$ satisfying $\hat{T}_{|\Gamma_T} = 0.$

Remark. Note that the solution and the test function, respectively, have to satisfy the boundary conditions $T_{|\Gamma_T} = T_0$, $\hat{T}_{|\Gamma_T} = 0$ and the *regularity conditions*

$$\int_\Omega |\nabla T|^2 \mathrm{d}A < \infty, \quad \int_\Omega |\nabla \hat{T}|^2 \mathrm{d}A < \infty.$$

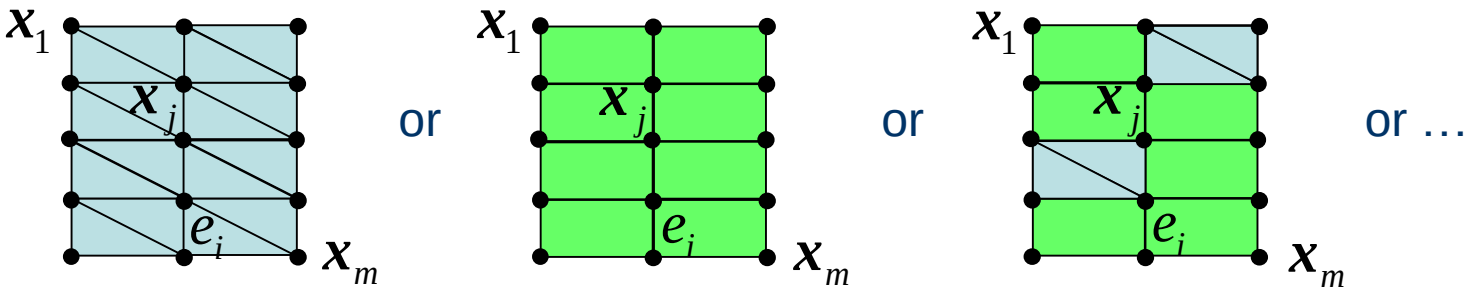
Then they are called *kinematically admissible.*

Remark. Starting from the weak form we could correspondingly derive the strong form (integrating by parts ”backwards”).

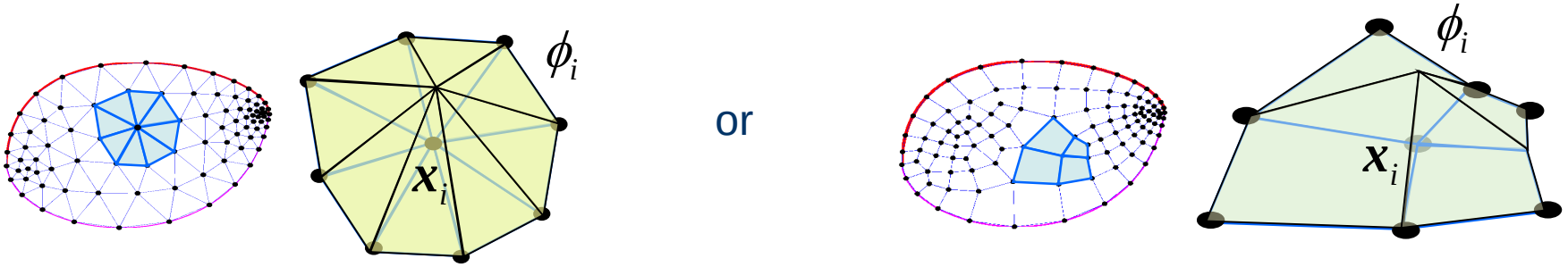
BACK
TO
basic material

3.2 2D finite element method – model problem

Finite element approximation for the 2D heat diffusion problem. Let us divide the solution area (domain) into n subdomains, *elements* e_i (*triangles, quadrangles, …*) with *nodes* $\boldsymbol{x}_j = (x_j, y_j)$ and *element size* $h_i = \mathrm{diam}(e_i)$:



In each element, the temperature field is approximated by *nodal values* of the temperature and in between the nodes by (linear) polynomial *basis functions* which are now (in a 2D problem) functions of the *x*- and *y*-coordinate:



3.2 2D finite element method − model problem

This finally results in a simple equation system

$$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$$

with the *stiffness matrix* (computable for *i,j = 1, …, m-p*), *force vector* (computable for *i = 1, …, m-p*) and the *displacement vector* (unknown for *i = 1, …, m-p*):

$$\boldsymbol{K} = \left[K_{ij}\right], \quad K_{ij} = \int_\Omega (k\nabla\phi_j)\cdot\nabla\phi_i \,\mathrm{d}\Omega,$$

$$\boldsymbol{f} = \left[f_i\right], \quad f_i = \int_\Omega f\,\phi_i \,\mathrm{d}\Omega - \int_{\Gamma_q} q_0\phi_i \,\mathrm{ds} - \sum_{\boldsymbol{x}_j\in\Gamma_T} T_0(\boldsymbol{x}_j)K_{ij},$$

$$\boldsymbol{d} = \left[d_j\right].$$

Remark. The stiffness matrix is (very often) *symmetric* (due to derivative orders) and its entries are concentrated in a narrow diagonal band forming a *band matrix* (due to local trial and test functions). These features can be utilized in computer impelementation − implying small amounts of memory needs and quick processing.

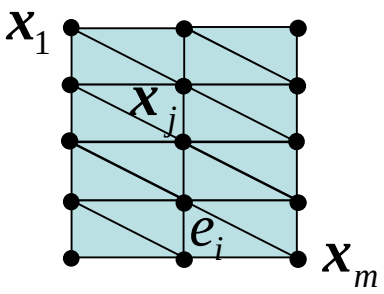
Remark. Test and trial functions have to be (only) once differentiable (and will be then integrated over the domain) and (only) evaluable on the boundary.

HOW TO
DERIVE THE FINITE ELEMENT SYSTEM?

extra material

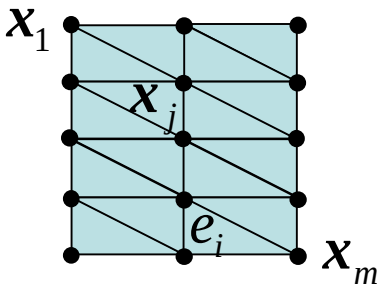
3.2 2D finite element method − model problem

1. Divide the solution area (domain) into n subdomains, *elements* e_i (*triangles, quadrangles, …*) with *nodes* $\boldsymbol{x}_j = (x_j, y_j)$ and *element size* $h_i = \mathrm{diam}(e_i)$:

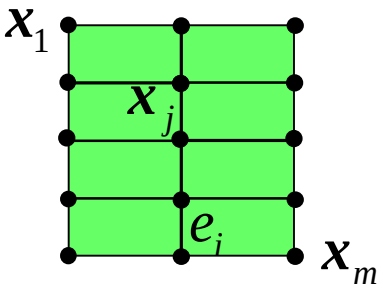


3.2 2D finite element method – model problem

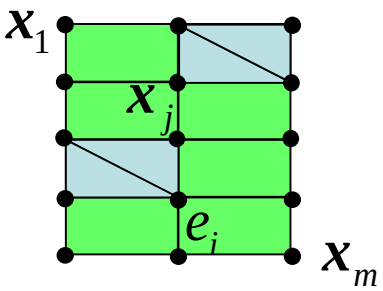
1. Divide the solution area (domain) into n subdomains, *elements* e_i (*triangles, quadrangles, …*) with *nodes* $\boldsymbol{x}_j = (x_j, y_j)$ and *element size* $h_i = \operatorname{diam}(e_i)$:



or



or



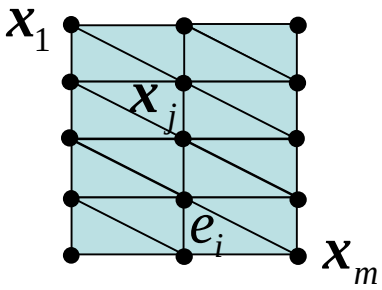
or …

2. Choose a *trial function* for the finite element approximation as a sum

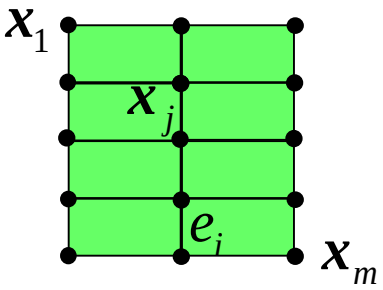
$$T_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})d_0 + \phi_1(\boldsymbol{x})d_1 + \cdots + \phi_m(\boldsymbol{x})d_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})d_j$$

3.2 2D finite element method – model problem

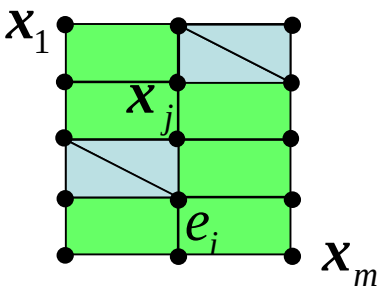
1. Divide the solution area (domain) into n subdomains, *elements* e_i (*triangles, quadrangles, …*) with *nodes* $\boldsymbol{x}_j = (x_j, y_j)$ and *element size* $h_i = \operatorname{diam}(e_i)$:



or



or

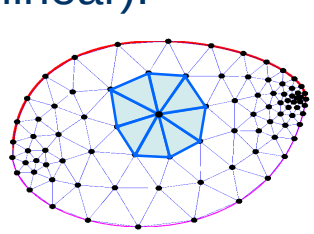


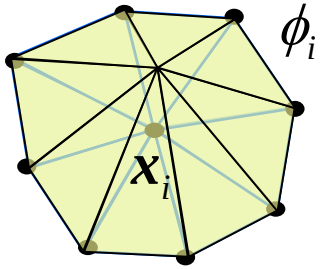
or …

2. Choose a *trial function* for the finite element approximation as a sum

$$T_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})d_0 + \phi_1(\boldsymbol{x})d_1 + \cdots + \phi_m(\boldsymbol{x})d_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})d_j$$

with suitable *local basis functions* ϕ_i of some polynomial order (now linear or bilinear):

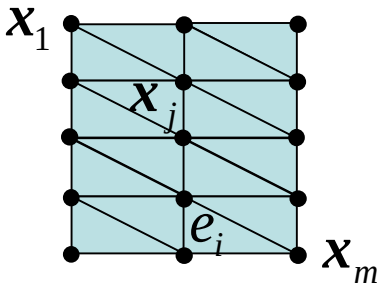




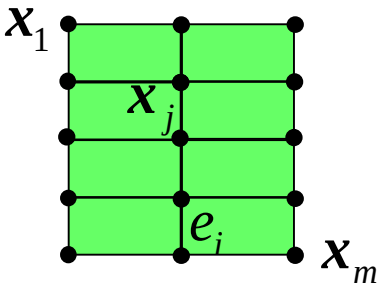
$$\phi_i(\boldsymbol{x}_j) = \delta_{ij}$$

3.2 2D finite element method – model problem

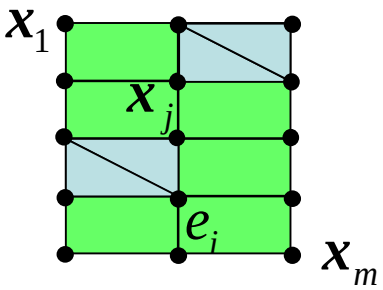
1. Divide the solution area (domain) into n subdomains, *elements* e_i (*triangles, quadrangles, …*) with *nodes* $\boldsymbol{x}_j = (x_j, y_j)$ and *element size* $h_i = \mathrm{diam}(e_i)$:



or



or

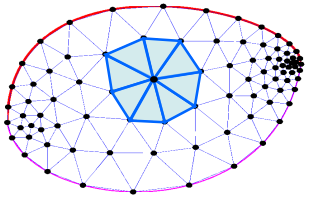


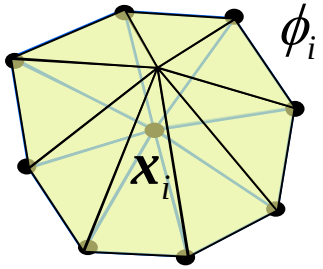
or …

2. Choose a *trial function* for the finite element approximation as a sum

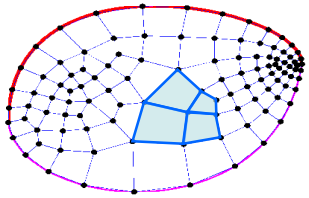
$$T_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})d_0 + \phi_1(\boldsymbol{x})d_1 + \cdots + \phi_m(\boldsymbol{x})d_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})d_j$$

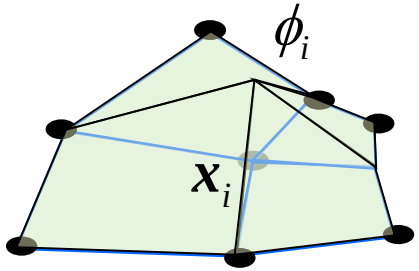
with suitable *local basis functions* ϕ_i of some polynomial order (now linear or bilinear):





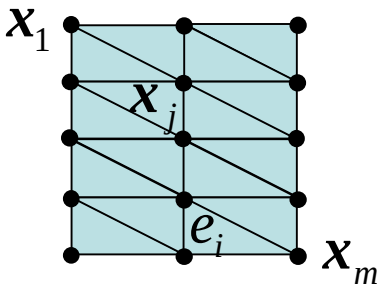
$$\phi_i(\boldsymbol{x}_j) = \delta_{ij}$$



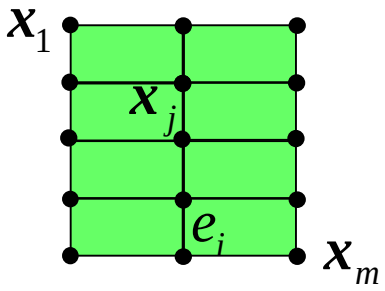


3.2 2D finite element method – model problem

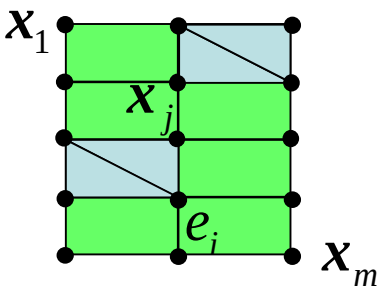
1. Divide the solution area (domain) into n subdomains, *elements* e_i (*triangles, quadrangles, …*) with *nodes* $\boldsymbol{x}_j = (x_j, y_j)$ and *element size* $h_i = \operatorname{diam}(e_i)$:



or



or

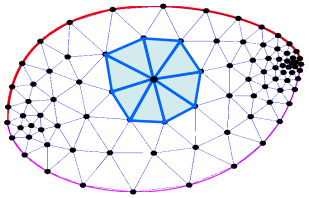


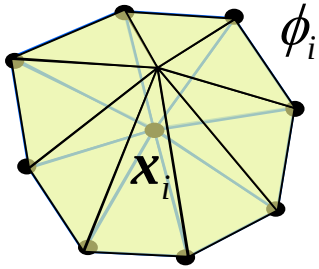
or …

2. Choose a *trial function* for the finite element approximation as a sum

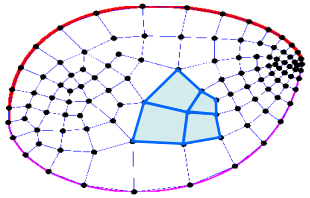
$$T_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})d_0 + \phi_1(\boldsymbol{x})d_1 + \cdots + \phi_m(\boldsymbol{x})d_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})d_j$$

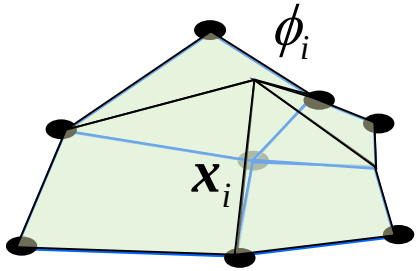
with suitable *local basis functions* ϕ_i of some polynomial order (now linear or bilinear):





$$\phi_i(\boldsymbol{x}_j) = \delta_{ij}$$





The unknown scalar values $d_i = T_h(\boldsymbol{x}_i)$ are called the *degrees of freedom.*

3.2 2D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$T_0(\boldsymbol{x}_j) = T_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j) d_j = d_j \quad \forall \boldsymbol{x}_j = (x_j, y_j) \in \Gamma_T \quad (\text{say, } p \ j\text{'s})$$

3.2 2D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$T_0(\boldsymbol{x}_j) = T_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j) d_j = d_j \quad \forall \boldsymbol{x}_j = (x_j, y_j) \in \Gamma_T \quad (\text{say, } p \ j\text{'s})$$

3. Choose a test function of a similar form (*Galerkin method*) with the corresponding (but zero) condition:

$$\hat{T}_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})c_0 + \phi_1(\boldsymbol{x})c_1 + \cdots + \phi_m(\boldsymbol{x})c_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})c_j$$

$$0 = \hat{T}_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j)c_j = c_j \quad \forall \boldsymbol{x}_j \in \Gamma_T$$

3.2 2D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$T_0(\boldsymbol{x}_j) = T_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j) d_j = d_j \quad \forall \boldsymbol{x}_j = (x_j, y_j) \in \Gamma_T \quad (\text{say, } p \ j\text{'s})$$

3. Choose a test function of a similar form (*Galerkin method*) with the corresponding (but zero) condition:

$$\hat{T}_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})c_0 + \phi_1(\boldsymbol{x})c_1 + \cdots + \phi_m(\boldsymbol{x})c_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})c_j$$

$$0 = \hat{T}_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j)c_j = c_j \quad \forall \boldsymbol{x}_j \in \Gamma_T$$

4. Insert the functions – trial and test – into the weak form:

$$\int_\Omega (k\nabla T) \cdot \nabla \hat{T} \, \mathrm{d}\Omega = \int_\Omega f \hat{T} \, \mathrm{d}\Omega - \int_{\Gamma_q} q_0 \hat{T} \, \mathrm{ds}$$

3.2 2D finite element method – model problem

Ensure that the trial function satisfies the essential boundary conditions:

$$T_0(\boldsymbol{x}_j) = T_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j) d_j = d_j \quad \forall \boldsymbol{x}_j = (x_j, y_j) \in \Gamma_T \quad (\text{say, } p \text{ } j\text{'s})$$

3. Choose a test function of a similar form (*Galerkin method*) with the corresponding (but zero) condition:

$$\hat{T}_h(\boldsymbol{x}) = \phi_0(\boldsymbol{x})c_0 + \phi_1(\boldsymbol{x})c_1 + \cdots + \phi_m(\boldsymbol{x})c_m = \sum_{j=0}^{m} \phi_j(\boldsymbol{x})c_j$$

$$0 = \hat{T}_h(\boldsymbol{x}_j) = \sum_{j=0}^{m} \phi_j(\boldsymbol{x}_j)c_j = c_j \quad \forall \boldsymbol{x}_j \in \Gamma_T$$

4. Insert the functions – trial and test – into the weak form:

$$\int_\Omega (k\nabla T) \cdot \nabla \hat{T} \, \mathrm{d}\Omega = \int_\Omega f \hat{T} \, \mathrm{d}\Omega - \int_{\Gamma_q} q_0 \hat{T} \, \mathrm{ds} \quad \Rightarrow$$

$$\int_\Omega \left(k \sum_{j=0}^{m} \nabla \phi_j d_j\right) \cdot \sum_{i=0}^{m} \nabla \phi_i c_i \, \mathrm{d}\Omega = \int_\Omega f \sum_{i=0}^{m} \phi_i \, c_i \, \mathrm{d}\Omega - \int_{\Gamma_q} q_0 \sum_{i=0}^{m} \phi_i \, c_i \, \mathrm{ds}$$

3.2 2D finite element method – model problem

This results in a simple equation system

$$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$$

with the *stiffness matrix* (computable for *i,j = 1, …, m-p*), *force vector* (computable for *i = 1, …, m-p*) and the *displacement vector* (unknown for *i = 1, …, m-p*):

$$\boldsymbol{K} = \left[K_{ij}\right], \quad K_{ij} = \int_\Omega (k\nabla\phi_j)\cdot\nabla\phi_i \,\mathrm{d}\Omega,$$

$$\boldsymbol{f} = \left[f_i\right], \quad f_i = \int_\Omega f\,\phi_i \,\mathrm{d}\Omega - \int_{\Gamma_q} q_0\phi_i \,\mathrm{ds} - \sum_{\boldsymbol{x}_j\in\Gamma_T} \int_\Omega (k\nabla\phi_j\, T_0(\boldsymbol{x}_j))\cdot\nabla\phi_i \,\mathrm{d}\Omega,$$

$$\boldsymbol{d} = \left[d_j\right]$$

3.2 2D finite element method – model problem

This results in a simple equation system

$$\boldsymbol{K}\,\boldsymbol{d} = \boldsymbol{f}$$

with the *stiffness matrix* (computable for *i,j = 1, …, m-p*), *force vector* (computable for *i = 1, …, m-p*) and the *displacement vector* (unknown for *i = 1, …, m-p*):

$$\boldsymbol{K} = \left[K_{ij}\right], \quad K_{ij} = \int_\Omega (k\nabla\phi_j)\cdot\nabla\phi_i \,\mathrm{d}\Omega,$$

$$\boldsymbol{f} = \left[f_i\right], \quad f_i = \int_\Omega f\,\phi_i \,\mathrm{d}\Omega - \int_{\Gamma_q} q_0\phi_i \,\mathrm{ds} - \sum_{\boldsymbol{x}_j\in\Gamma_T} T_0(\boldsymbol{x}_j)K_{ij},$$

$$\boldsymbol{d} = \left[d_j\right]$$

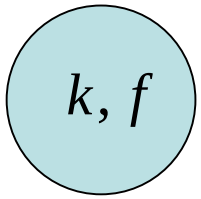
BACK
TO
basic material

Computer exercise 3.1 − Matlab PDE Modeler

Let us consider isotropic and homogeneous heat diffusion in a circle with radius R and the following problem data:

$$k = 0.1\,\mathrm{W/(m^\circ C)}, R = 1\,\mathrm{m}$$

$$f = 10\,\mathrm{W/m^3}, T_0 = 0\ ^\circ\mathrm{C}$$



(i) Solve the temperature distribution approximately via the finite element method by applying Matlab PDE Modeler with three different mesh sizes.

(ii) Plot the corresponding triangular meshes, temperature distributions (T) and heat flux fields ($\boldsymbol{q}$) for each finite element solution.

(iii) List three, perhaps simplified, engineering problems which you can be seen as applications of this model problem.

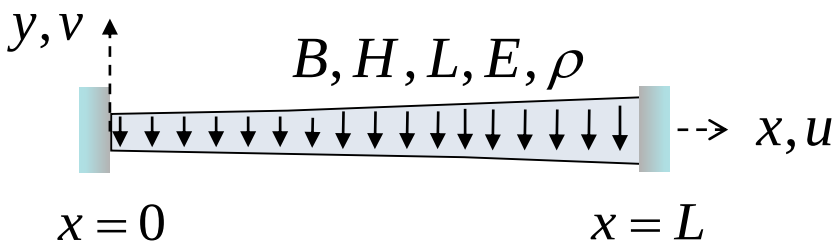
Hint: Take part in the guided tour in the exercise session.

Computer exercise 3.2 − Matlab PDE Modeler

Let us consider a 2D/3D *beam bending* problem as a 2D *plane stress elasticity* problem with displacements $u = u(x,y)$ and $v = v(x,y)$ in the x- and y-directions, correspondingly, as primary variables (i.e., displacement vector is $\boldsymbol{u} = (u, v)$) and self-weight as a body load $\boldsymbol{f} = (f_x, f_y)$:

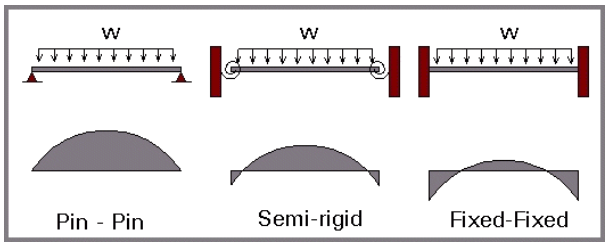
$$B = 4\ \text{cm}, H = 2\ \text{cm}, L = 1\ \text{m}$$

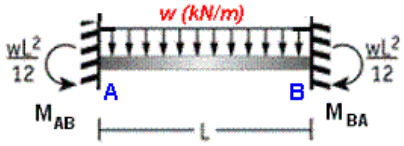
$$E = 200\ \text{GPa},\ \rho = 8\ \text{kg/m}^3$$

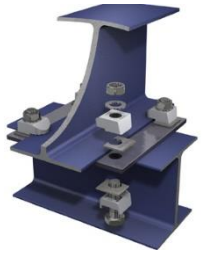


Solve the 2D deformation and stress state of the beam (meaning displacement components u and v and stress components by utilizing Matlab *PDE Modeler* relying on the finite element method (with triangular elements of linear basis functions).

Hint: Take part in the guided tour in the exercise session.







4 Mesh refinements, adaptive and isogeometric methods

Fundamentals of Finite Element Analysis

Lecture 1

1. Modelling principles and boundary value problems in engineering sciences
   - Home exercise 1.1
   - Computer exercise 1.1 – Matlab
2. Energy methods and basic 1D finite element methods
   - bars/rods, beams, heat diffusion, seepage, electrostatics
   - Home exercise 2.1
   - Home exercise 2.2

Lecture 2

3. Basic 2D and 3D finite element methods
   - heat diffusion, seepage
   - Computer exercise 3.1 – Matlab PDE Modeler
   - Computer exercise 3.2 – Matlab PDE Modeler
4. Mesh refinements, adaptive and isogeometric methods
   - Computer exercise 4.1 – Matlab PDE Modeler (VOLUNTARY)

4 Mesh refinements, adaptive and isogeometric methods

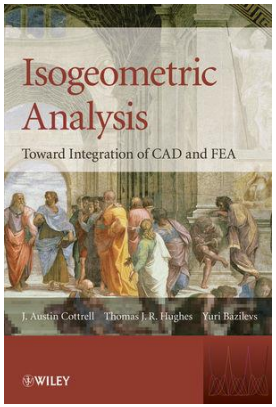
Contents

1. Mesh refinements and adaptive methods (basic idea)
2. Isogeometric finite element analysis (basic idea)

Learning outcome

A. *Understanding* the basic ideas of mesh refinements

B. *Understanding* the basic idea of isogeometric finite element analysis

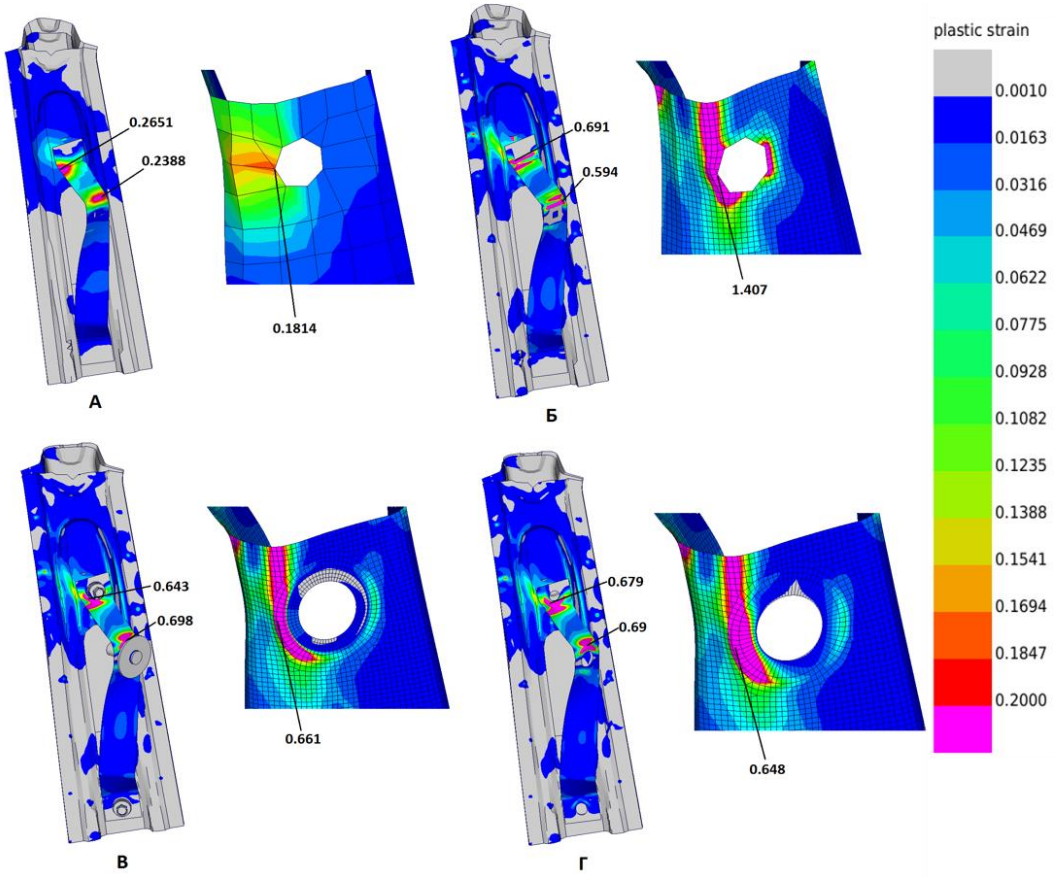


References

Chapters 1.1–2

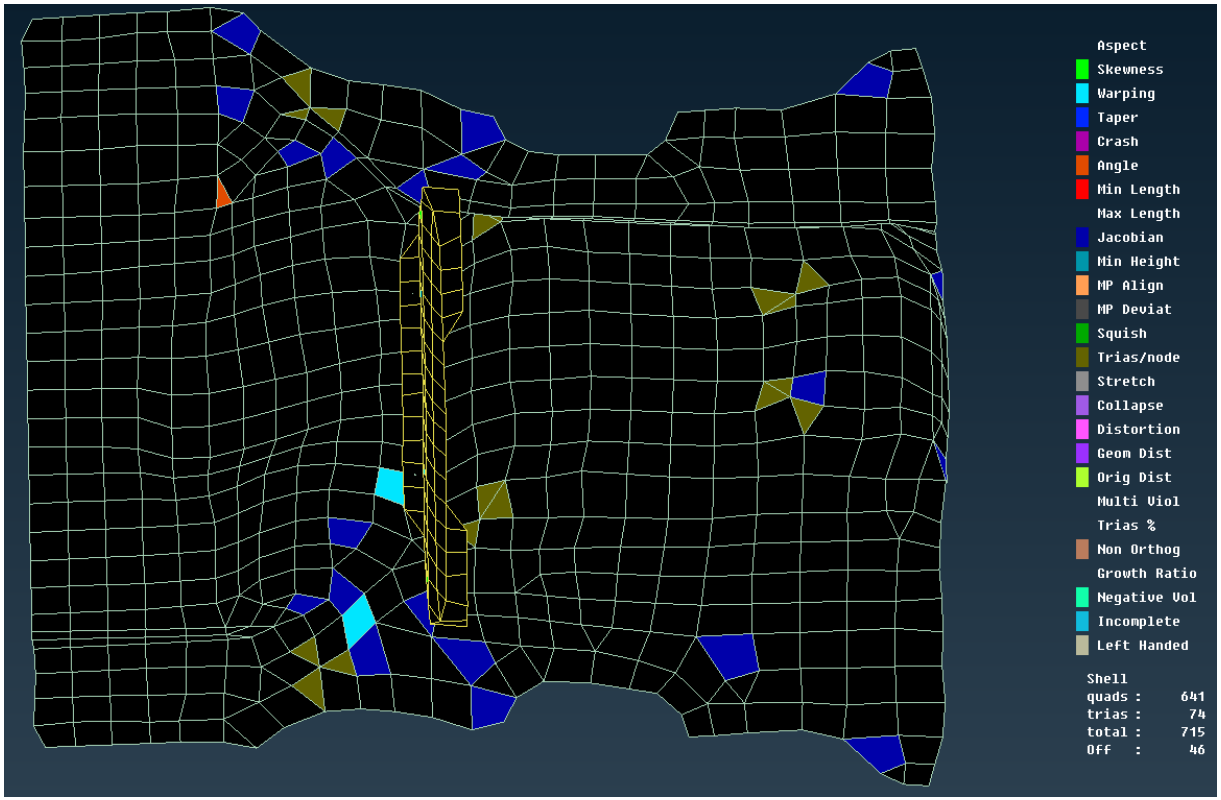
4.1 Mesh refinements and adaptive methods

Example of mesh refinements affecting the field approximation and geometry. For improving the accuracy of finite element field approximations and/or improving the accuracy of finite element geometries, mesh refinements are typically applied:



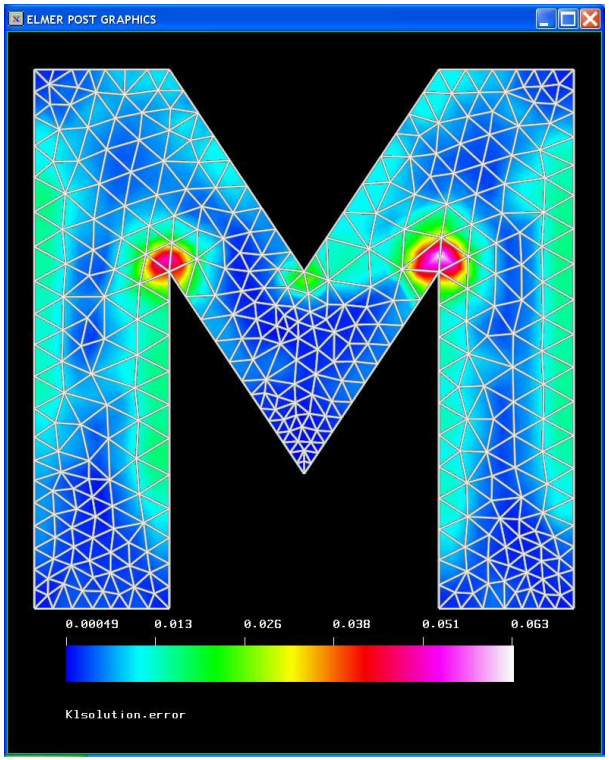
4.1 Mesh refinements and adaptive methods

Example of mesh quality. Automatic mesh refinements might produce poor mesh quality (warped or collapsed elements, for instance) instead of good quality quasi-uniform meshes (close to equilateral elements) reducing the accuracy of computations – not only the accuracy of the representation of the geometry.



4.1 Mesh refinements and adaptive methods

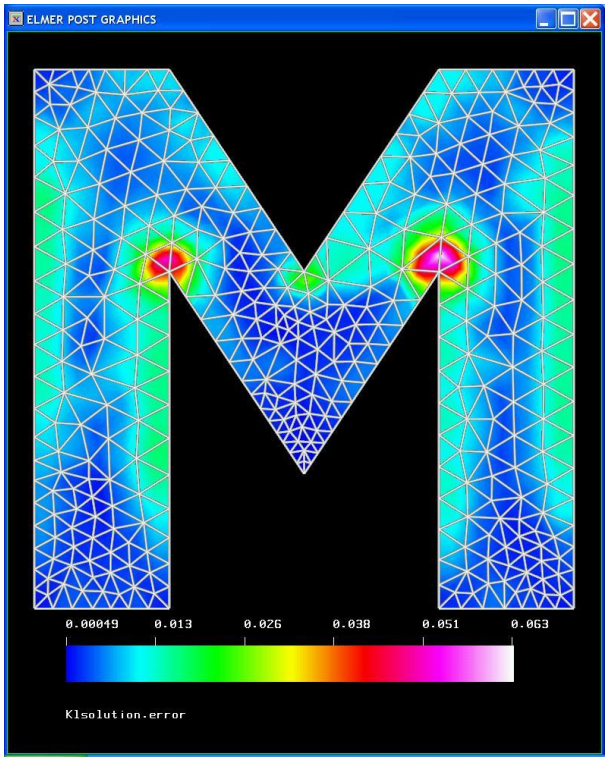
Example for *adaptive mesh refinements* with an error indicator developed for the Morley element (*2008*). Simply supported, uniformly loaded Kirchhoff−Love plate; *residual-based error indicators*; automated *Delaunay triangulations*.



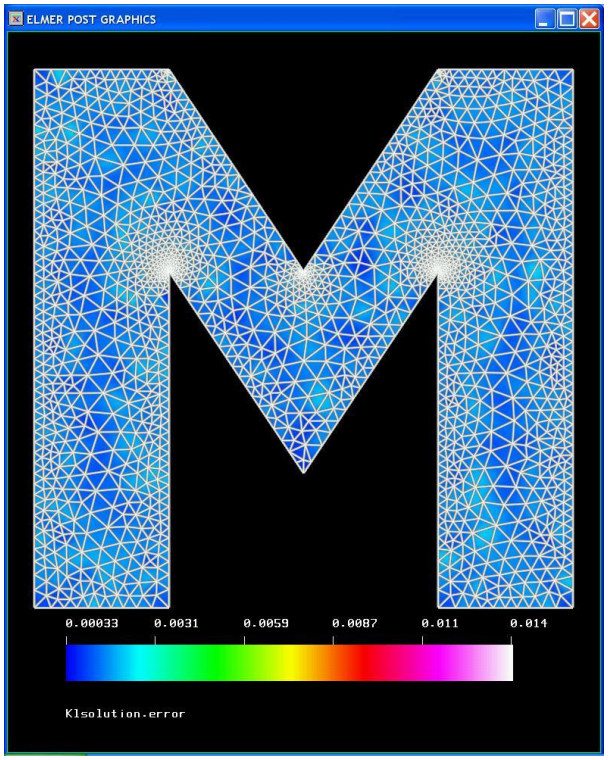
(a) Energy error and mesh for the iteration step *3*.

4.1 Mesh refinements and adaptive methods

Example for *adaptive mesh refinements* with an error indicator developed for the Morley element (*2008*). Simply supported, uniformly loaded Kirchhoff−Love plate; *residual-based error indicators*; automated *Delaunay triangulations*.

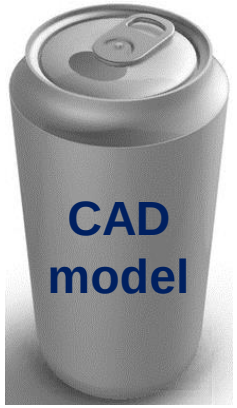


(a) Energy error and mesh for the iteration step *3*.



(b) Energy error and mesh for the iteration step *11*.

4.2 Isogeometric finite element analysis

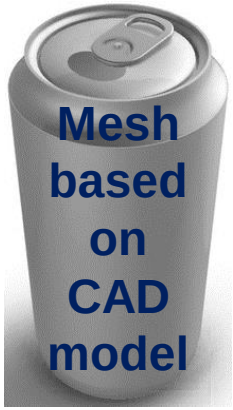


FEA Mesh and FE Geometry

IGA Mesh on CAD Geometry

Inexact

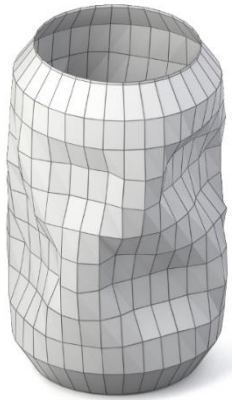
Exact



FEA Solution

IGA Solution

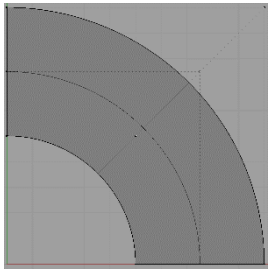
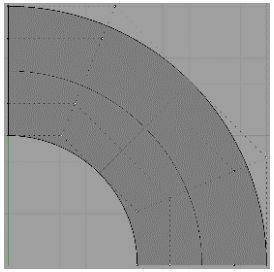
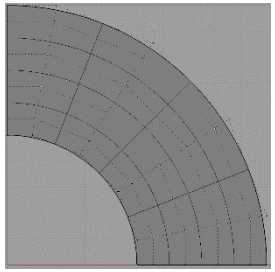
Edgy



Smooth or Edgy

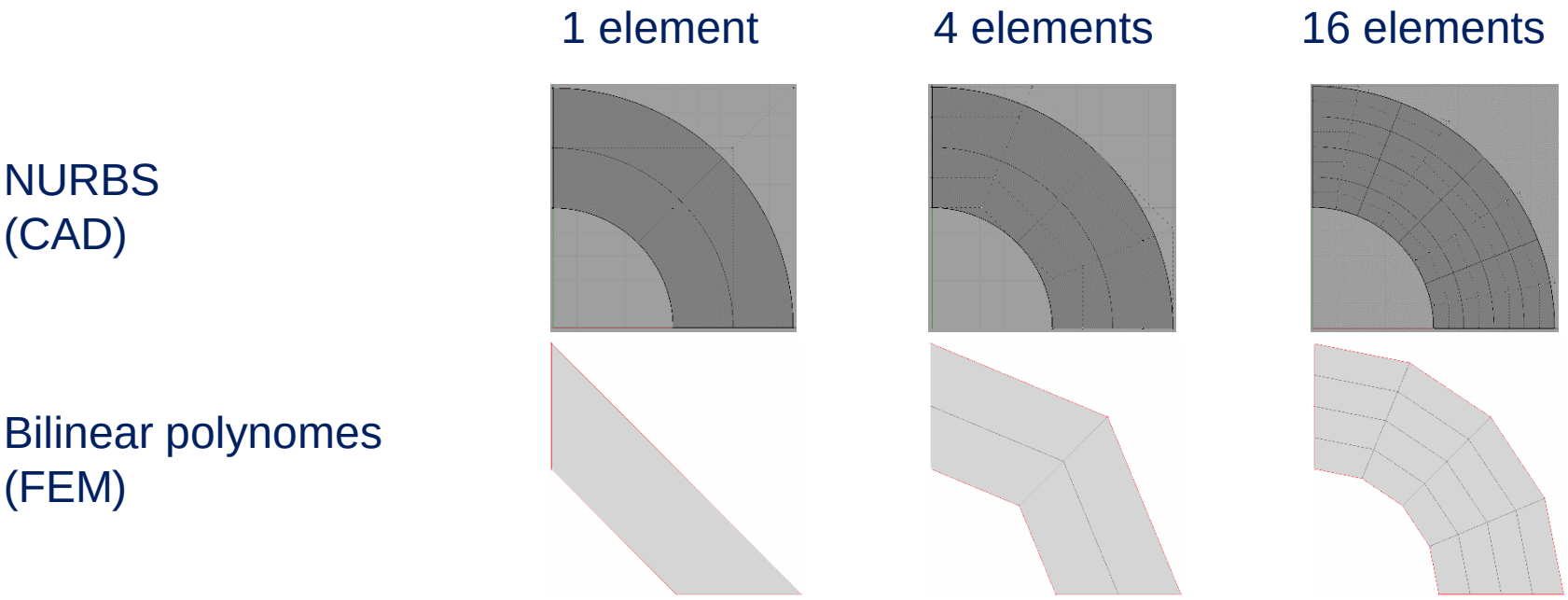
4.2 Isogeometric finite element analysis

Comparison of geometry mappings. CAD geometry versus quadrangular FEM geometries of first (linear) and second (quadratic) order.

	1 element	4 elements	16 elements
NURBS (CAD)			

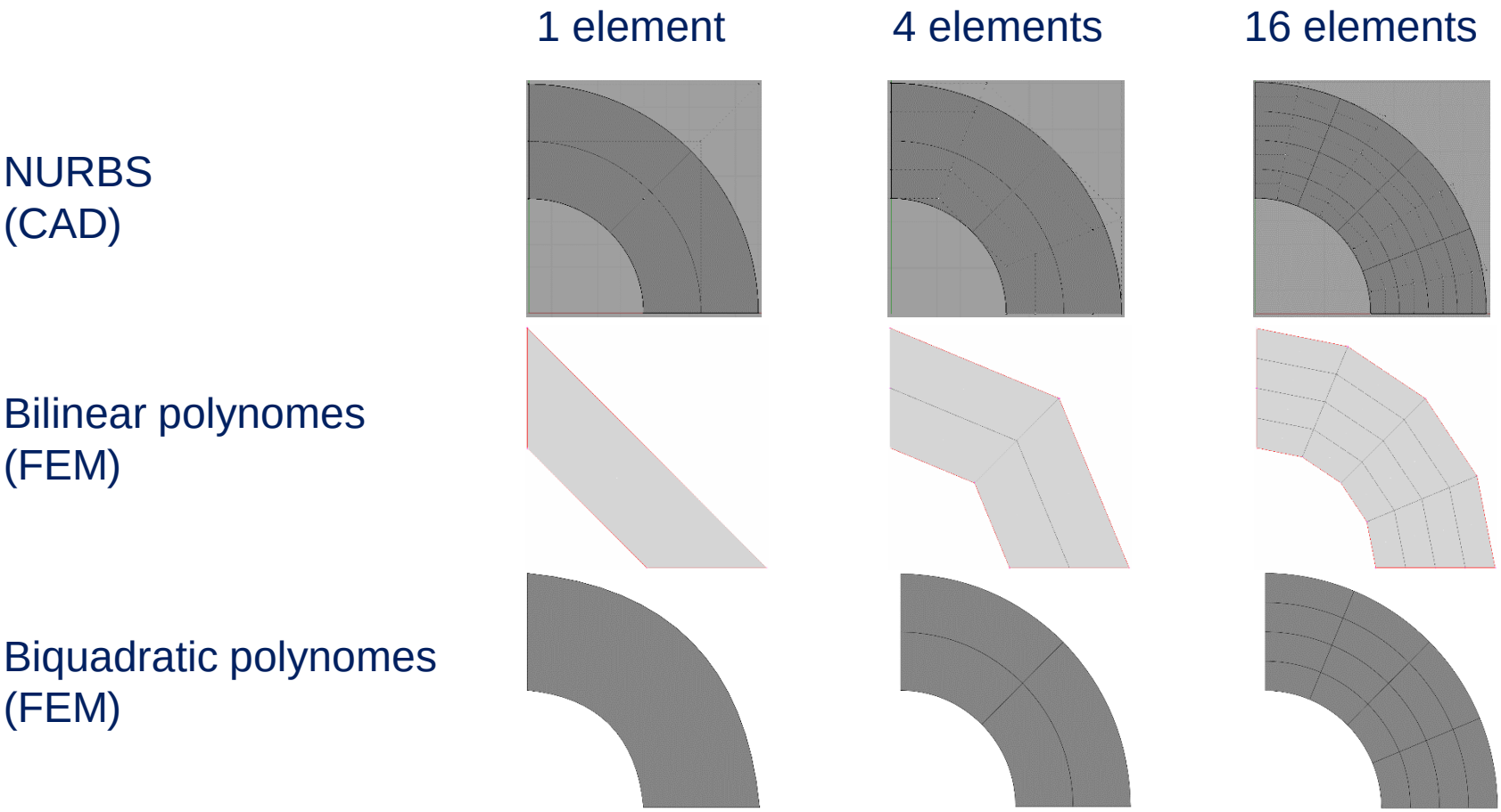
4.2 Isogeometric finite element analysis

Comparison of geometry mappings. CAD geometry versus quadrangular FEM geometries of first (linear) and second (quadratic) order.



4.2 Isogeometric finite element analysis

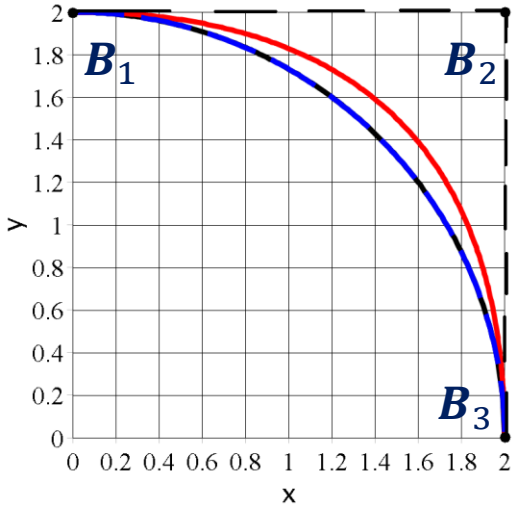
Comparison of geometry mappings. CAD geometry versus quadrangular FEM geometries of first (linear) and second (quadratic) order.

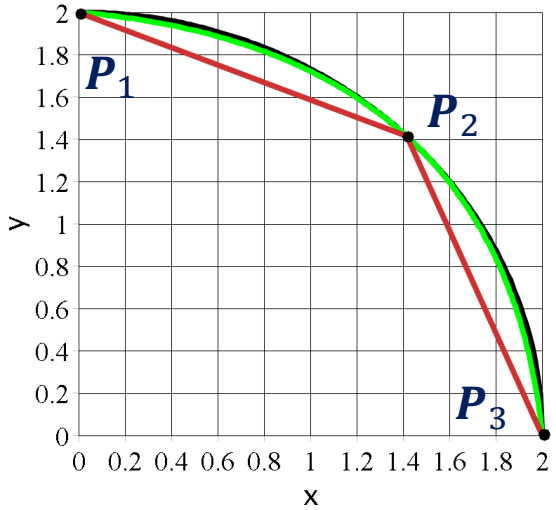


4.2 Isogeometric finite element analysis

Remark. NURBS are able to accurately represent conic curves but B-splines and polynomes can only approximate conic curves.

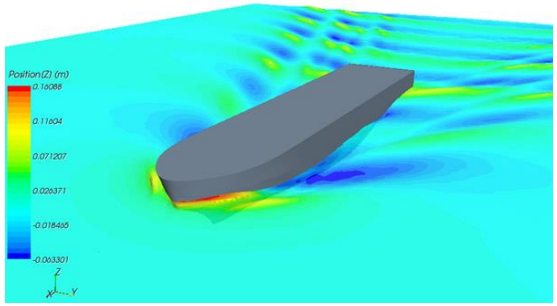
- Analytical curve: $y = \sqrt{4 - x^2}$, $x \in [0; 2]$
- NURBS curve: $T(\xi) = \sum_{i=1}^{3} R_i^2(\xi)\boldsymbol{B}_i$, $\xi \in [-1; 1]$
- B-spline curve: $C(\xi) = \sum_{i=1}^{3} N_{i,2}(\xi)\boldsymbol{B}_i$, $\xi \in [-1; 1]$
- Quadratic polynome interpolation: $Q(\xi) = \sum_{i=1}^{3} L_{i,2}(\xi)\boldsymbol{P}_i$, $\xi \in [-1; 1]$
- Linear polynome interpolation: $L(\xi) = \sum_{i=1}^{3} L_{i,1}(\xi)\boldsymbol{P}_i$, $\xi \in [-1; 1]$

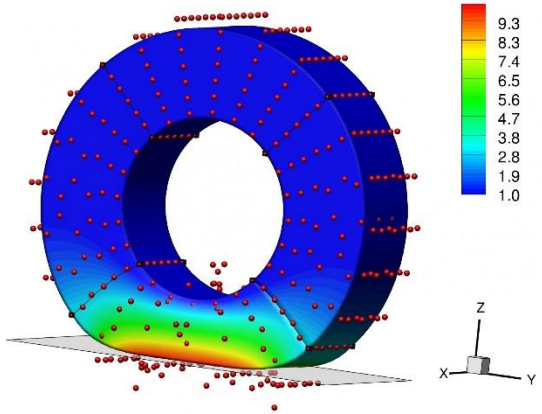


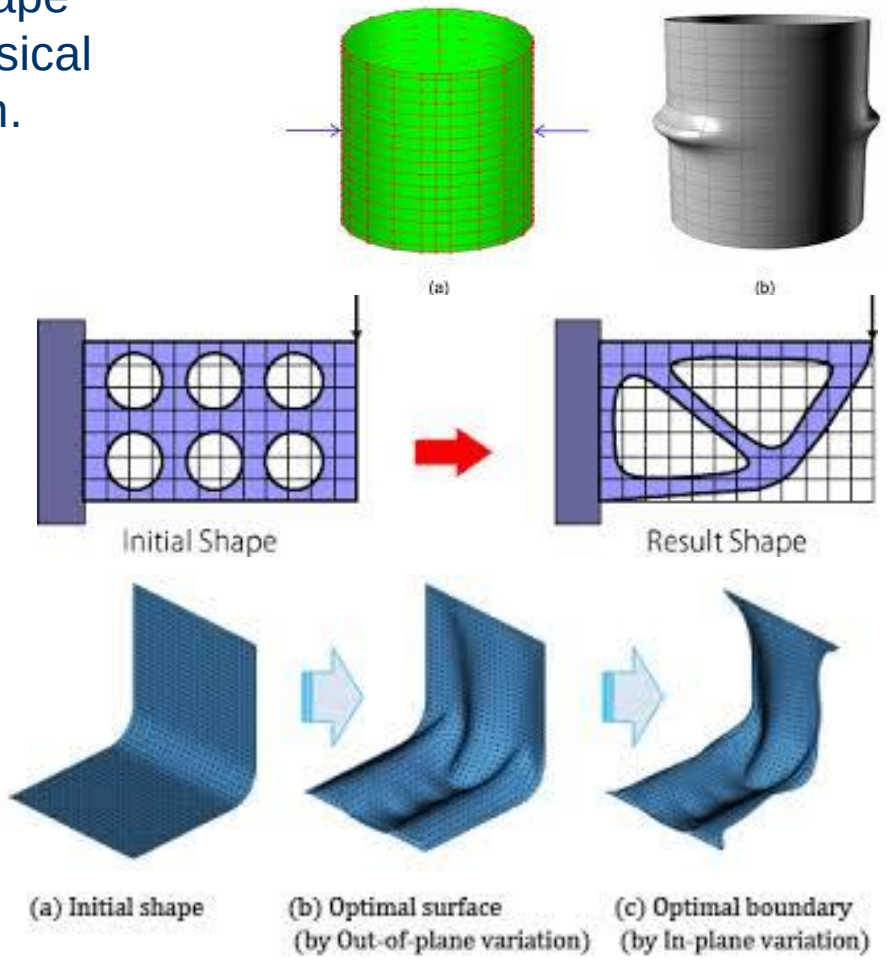


4.2 Isogeometric finite element analysis

Benefits expected from isogeometric analysis. Benefits for interoperability between CAD and CAE (such as FEA); shape optimization; contact mechanics; multi-physical problems such as fluid-structure interaction.







Computer exercise 4.1 − Matlab PDE Modeler

Let us consider the heat diffusion in a pizza oven made of bricks (see below). The temperature inside is assumed to be 400 Celsius degrees, whereas the outside temperature is $T\_inf = 20$ degrees. The dimensions of the oven are the following: wall thickness 15 cm, radius of the inner circle of the wall 35 cm. Newton's cooling is taken into account implying the following data for the boundary conditions: $T\_0 = 400$ and $k = 0.56$, $q = 6$, $g = qT\_inf = 120$ ($q = 5.6 + 4v = 6$ with wind speed $v = 0.4$).

Find the estimate for heat distribution diffusion from inside the oven through the circular brick wall and the flat bottom brick plate by a 2D heat diffusion model of Matlab *PDEModeler* for a cross section of the "tunnel" by neglecting the effect of the chimney (for out-going smoke).

THE END