



An Introduction to Finite Element Analysis Using Abaqus

Milad Vahidian, Ph.D. Candidate of Mechanical Engineering

Course Outline

1-Preliminary

Subjected Software
Linking Abaqus and Fortran

2- Introduction to FEA

Basic Concepts
Applications
Analysis Procedures

3-FEA Using Abaqus

Basic Concepts
Consistent Units
Abaqus Documentation
Terminology
Conventions
Dimension of Problems
Mesh Controls
Element Types
Abaqus Solvers
Solution Convergence
Generated Files
Abaqus Automation Possibility

4-Final Project

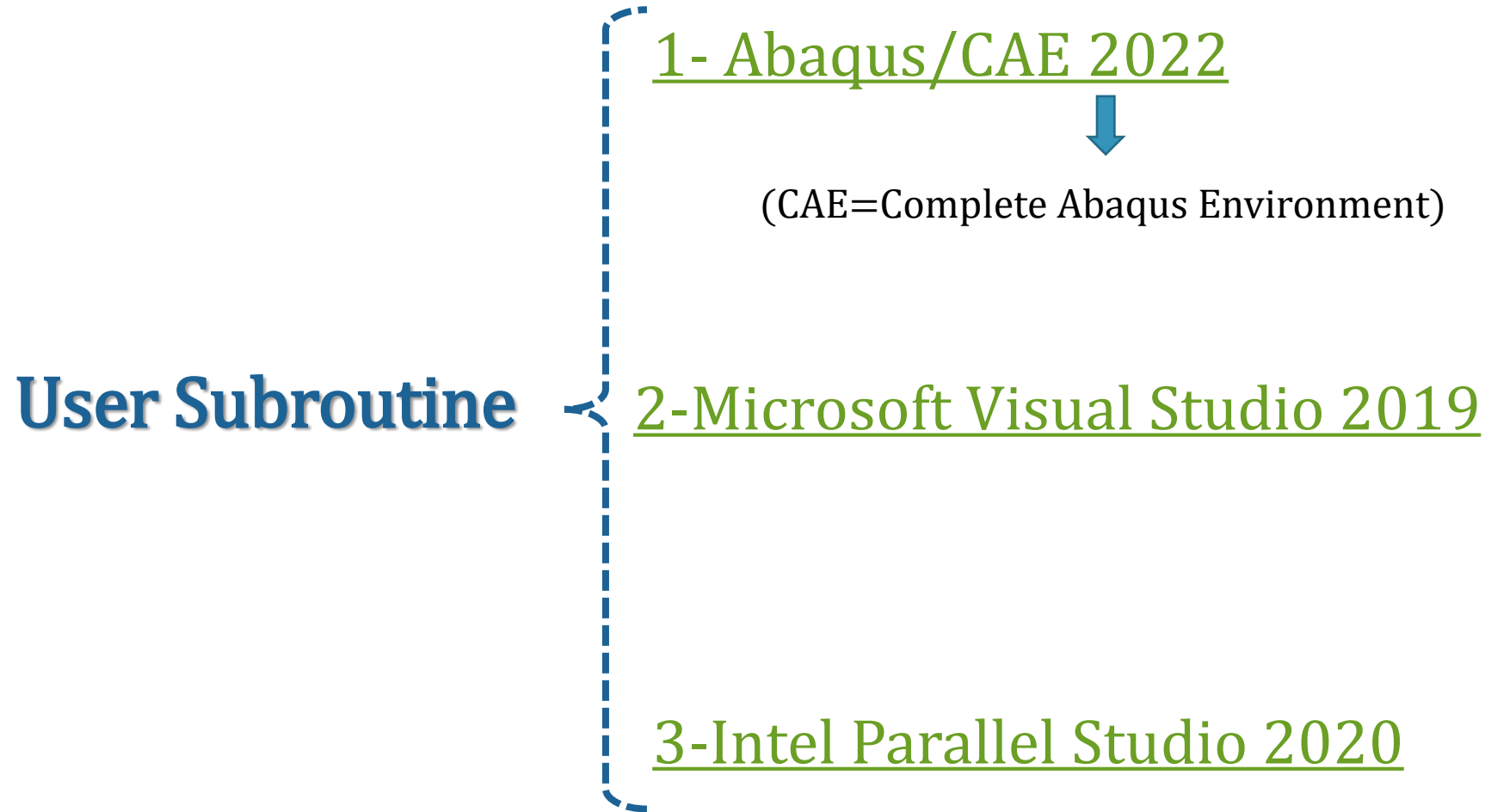
Suggested Subject
Types of Analysis

5-Sample Problems

Truss Problem
Beam Problem
Moving Load
Plane Stress
Natural Frequency
Wrinkling Patterns
Low velocity Impact
Extrusion
Fluid-structure Interaction

Preliminary

Preliminary: Subjected Software



Preliminary: Linking Abaqus and Fortran

Step 1: Installing Abaqus/CAE, Visual Studio, and Intel Parallel Studio respectively.

Step 2: Finding the directory of “ifortvars.bat” and “vcvars64.bat”

By default:

C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2020.4.311\windows\bin

C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Auxiliary\Build

Step 3: Adding these variable and associated directory into “Environment variables”

Step 4: Modifying Target

Adding this address to “Abaqus Command” and “Abaqus CAE” target

“C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2020.4.311\windows\bin\ifortvars.bat” intel64 vs2019 &

Step 5: Verification

- ☐ Abaqus Verification: run Abaqus Verification and check the .log file
- ☐ Abaqus Command: Enter “abaqus info=system” , “abaqus verify -user_std” and “abaqus verify -user_exp”

Introduction to FEA

Introduction to FEA: Motivation

Structural Analysis

Thermal Analysis

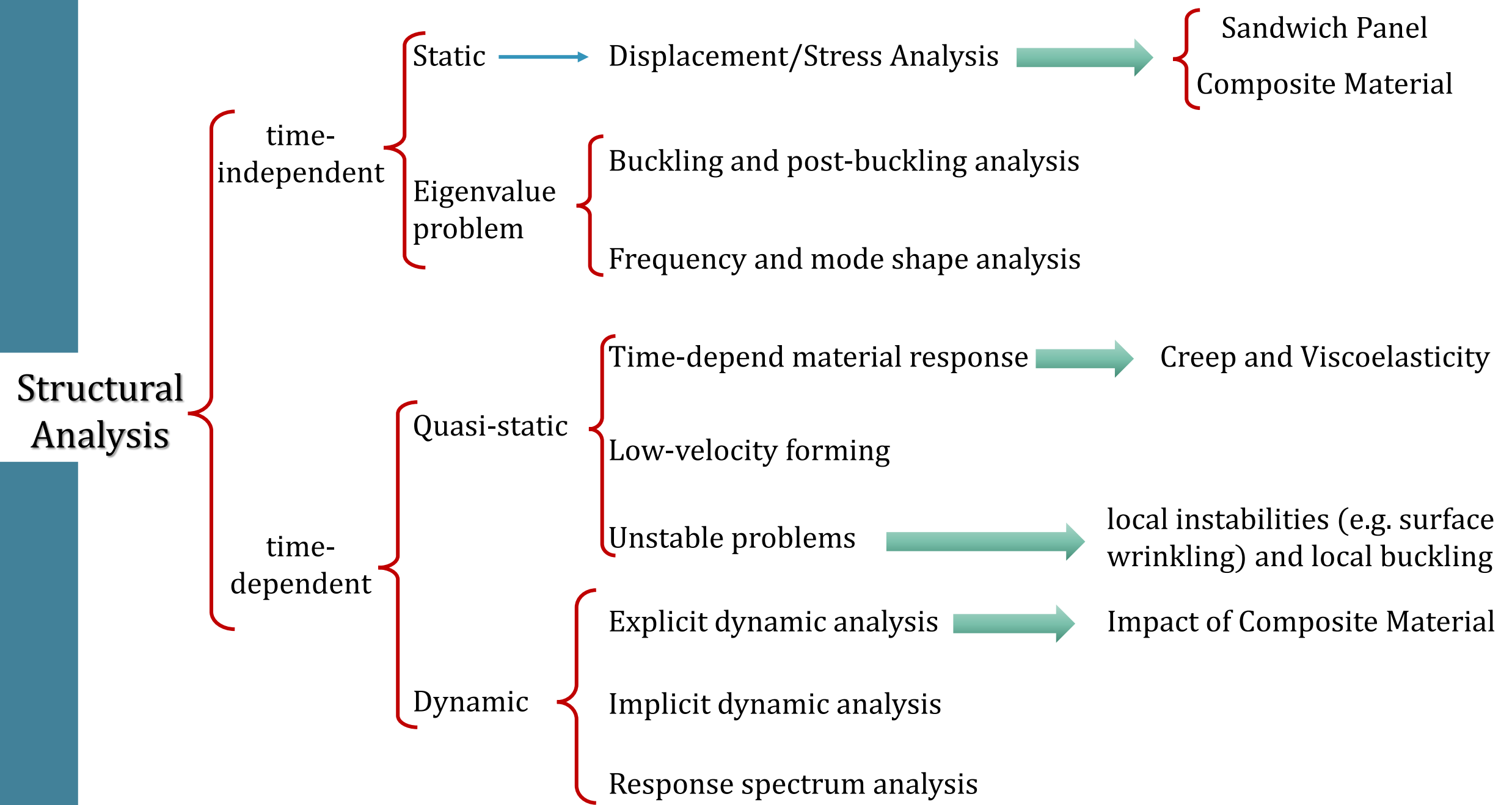
Fluid Structure Analysis

Electromagnetic Analysis

Multiphysics Analysis

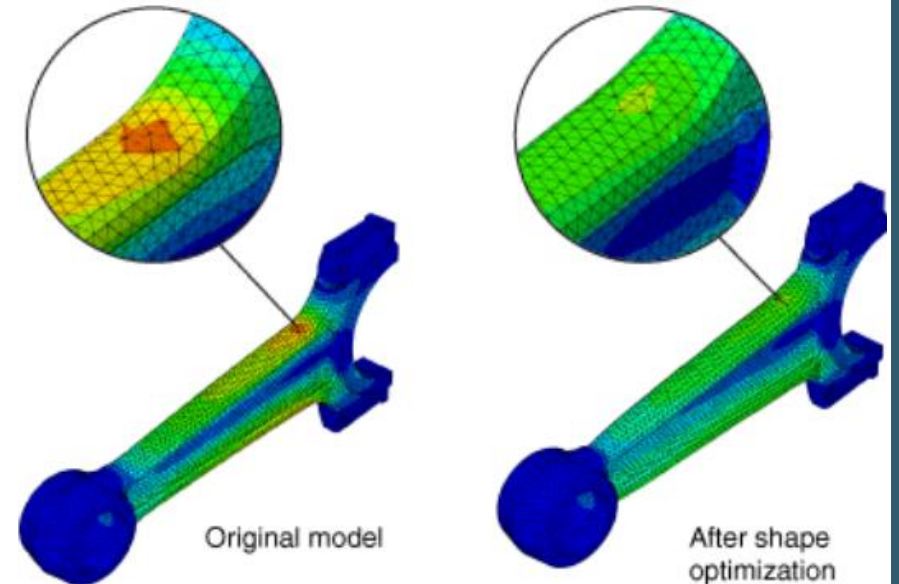
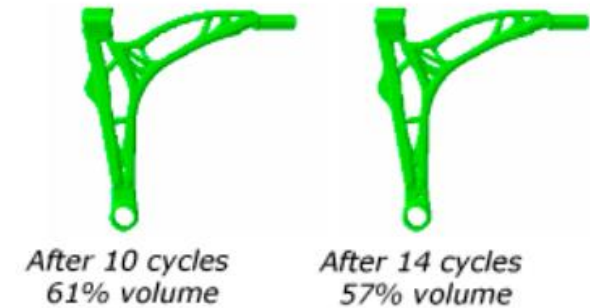
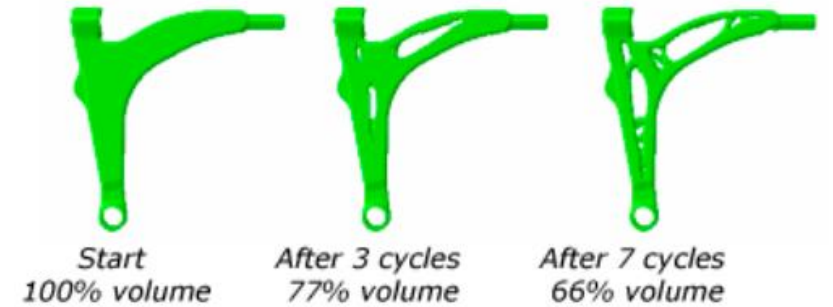
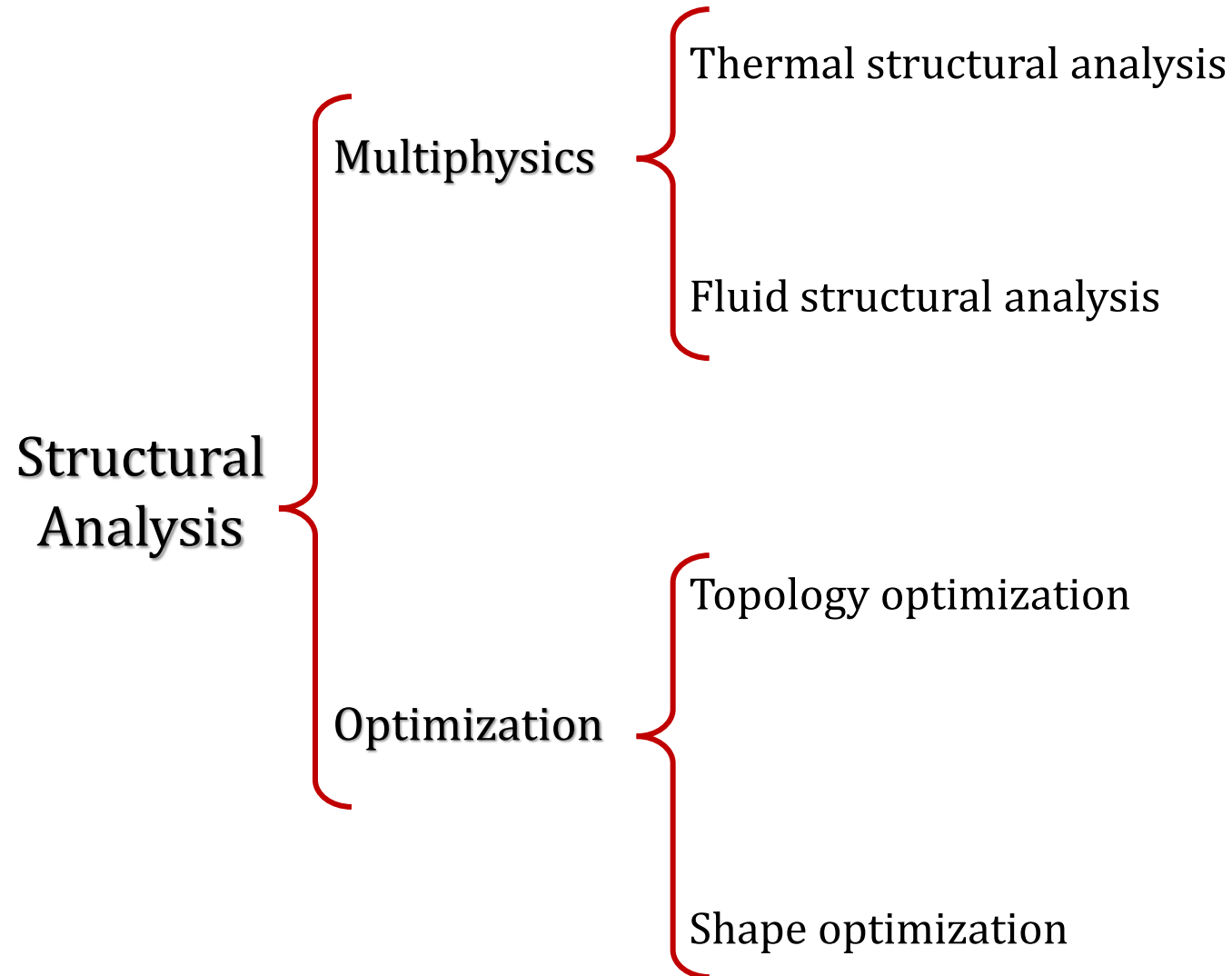
Optimization Analysis

Introduction to FEA: Motivation

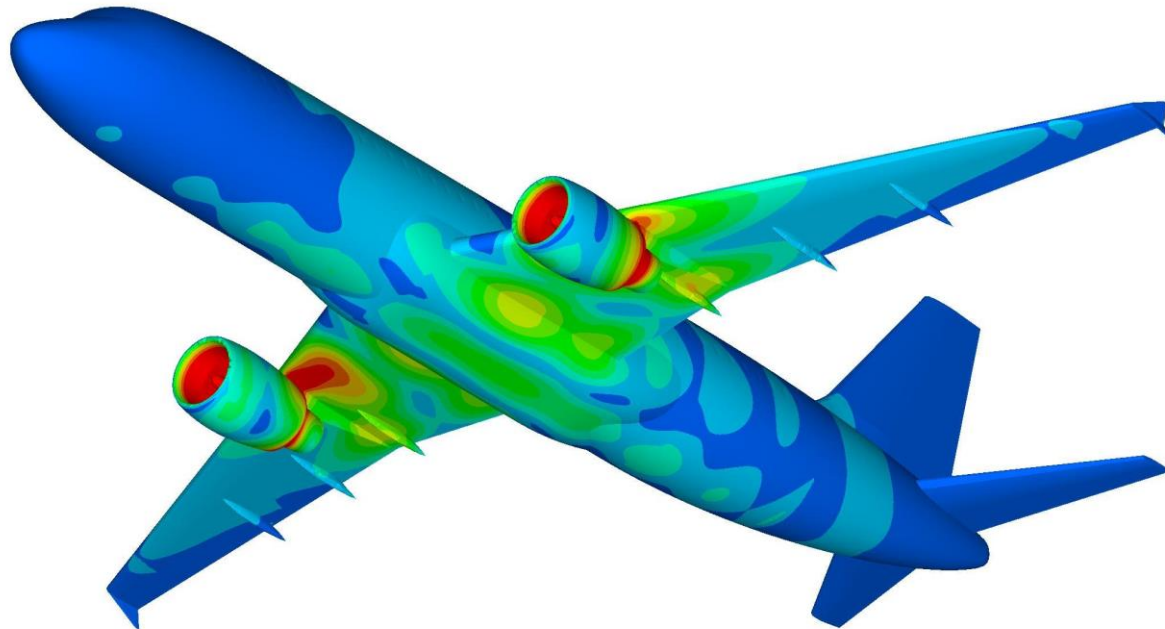
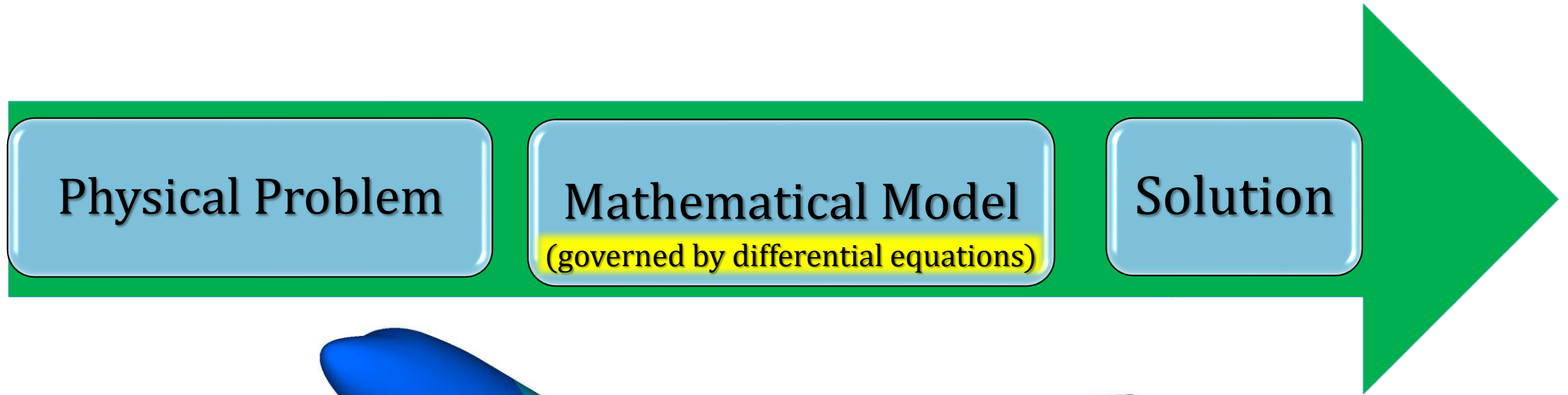


$$\cancel{[M]\{\ddot{a}(t)\} + [C]\{\dot{a}(t)\} + [K]\{a(t)\} = \{F(t)\}}$$
$$[K]\{a\} = \{F\}$$

Introduction to FEA: Motivation



Introduction to FEA: Basic Concepts



Introduction to FEA: Basic Concepts

Methods of Analysis

Analytical Methods

ODE

PDE → Separation of variables

Semi-analytical (Approximate) Methods

Lumped-parameter Methods

Series Discretization Methods

Numerical Methods

Numerical Integration

Finite Volume Method

Finite Element Method

Finite Difference Method

Boundary Element Method

The existing mathematical tools will not be sufficient to find the exact solution (and sometimes, even an approximate solution) of most of the practical problems.

Introduction to FEA: Basic Concepts

Analytical Methods

Semi-analytical (Approximate) Methods



Series Discretization Methods



Assumed Solution



Variational Approach

Weighted Residual Approach

Weak Form

Strang Form

Numerical Methods

Finite Element Method

Weak Form

Strang Form

Must be satisfied
Essential (geometry)
Boundary conditions

Must be satisfied
Essential (geometry)
as well as
Natural (force)
Boundary conditions

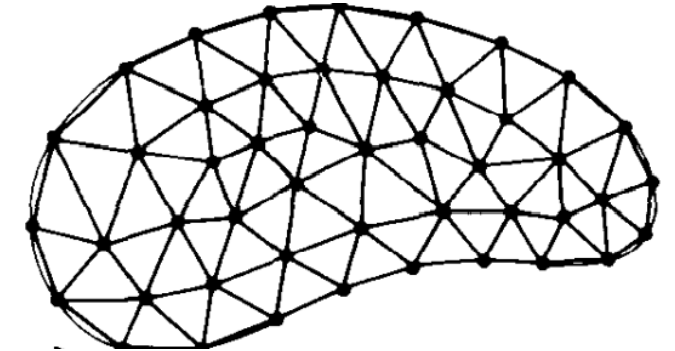
Introduction to FEA: Basic Concepts

What is Finite Element Analysis ?

The Finite Element Analysis (FEA) is the simulation of any given physical phenomenon using the numerical technique called Finite Element Method (FEM).

The basic idea behind the finite element method is to divide the structure, body, or region being analyzed into a large number of finite elements, or simply elements.

The solution region is considered to be built of many small, interconnected subregions called elements.



$$u(x, y) = c_1 + c_1 x + c_2 y$$

Space Discretization



FEM subdivides a large system into smaller, simpler parts that are called finite elements



construction of a **mesh** of the object

Introduction to FEA: Analysis Procedures

Procedures

1-Discretization

2-Interpolation (Shape Function)

shape function

$$u(x,y,z) = [N]_{(x,y,z)} \{d\}$$

3-Derivation of Characteristic Matrices (element stiffness matrix and load vectors)

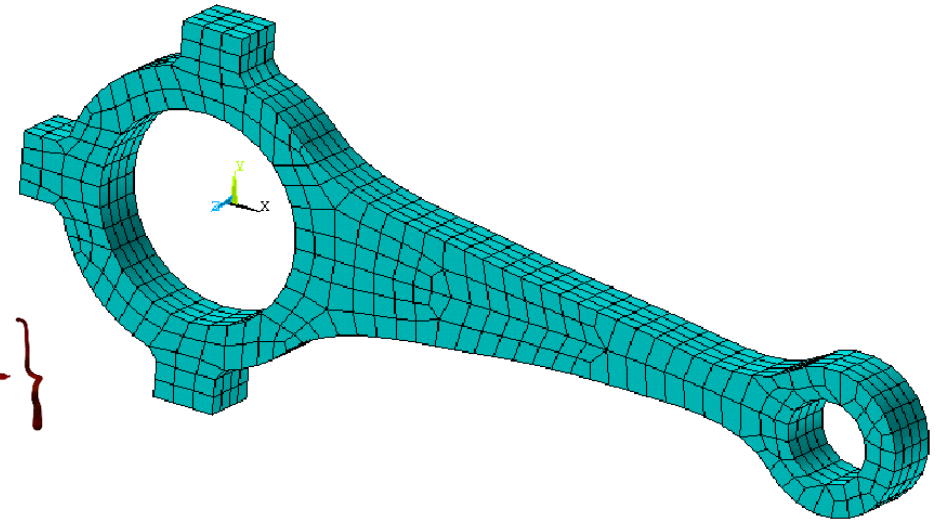
4-Assembly

$$[K_e] \{d_e\} = \{F_e\}$$

5-Applying Boundary Conditions

6-Solving Unknown

$$[K] \{d\} = \{F\}$$



Introduction to FEA: Analysis Procedures

Static Problem
(ODEs or PDEs)

$$\frac{d}{dx} \left(AE \frac{du(x)}{dx} \right) = w(x)$$

FEM

System of Algebraic Equations
(Linear or Non-linear)

$$[K]\{a\} = f \xrightarrow{\text{Abaqus/Standard (Implicit)}} \{a\}$$

Dynamic Problem
(PDEs)

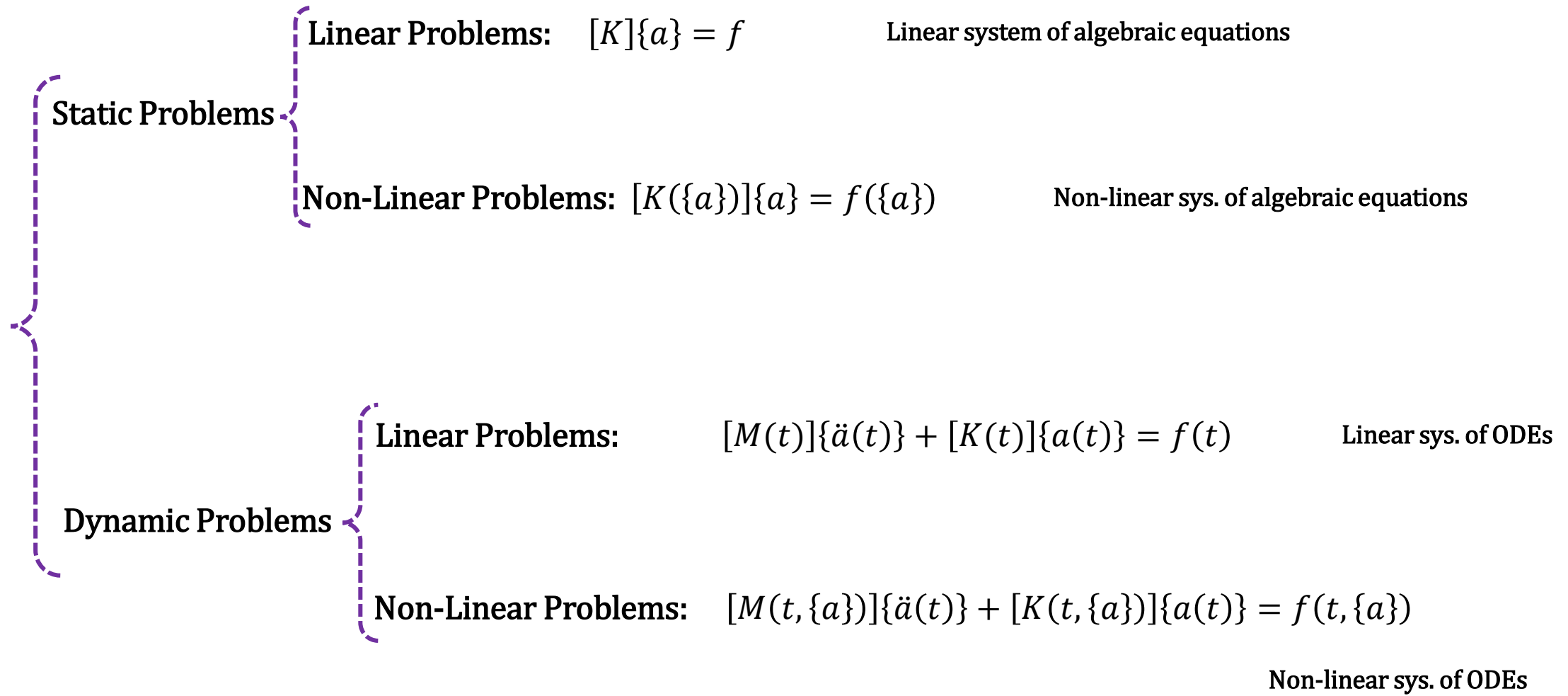
$$w(x, t) + \frac{d}{dx} \left(AE \frac{\partial u(x, t)}{\partial x} \right) = \rho \frac{\partial^2 u(x, t)}{\partial t^2}$$

FEM

System of ODEs
(Linear or Non-linear)

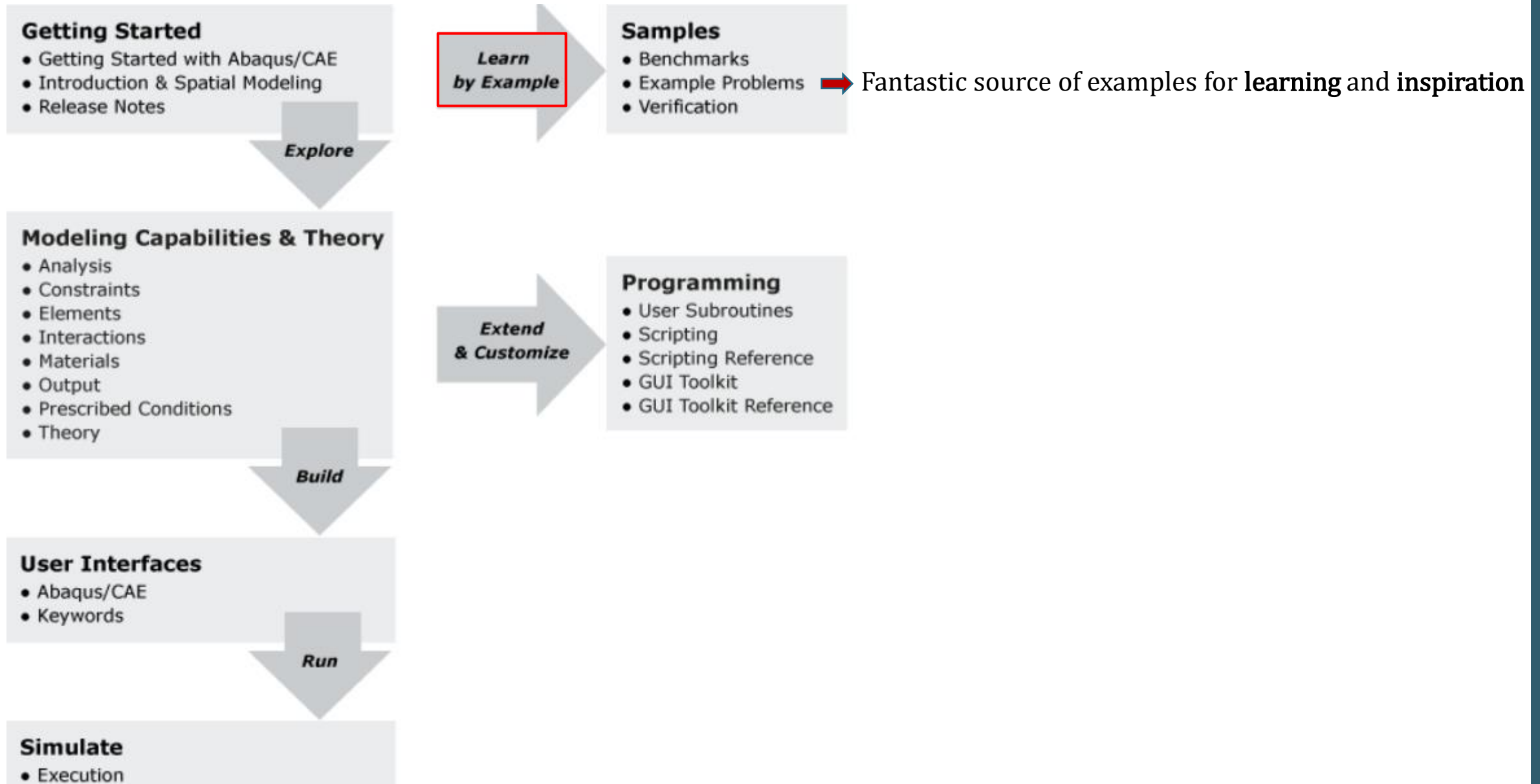
$$[M]\{\ddot{a}(t)\} + [K]\{a(t)\} = f(t) \xrightarrow[\text{Abaqus/Explicit (Dynamic Explicit)}]{\text{Abaqus/Standard (Dynamic Implicit)}} \{a\}$$

Introduction to FEA: Analysis Procedures

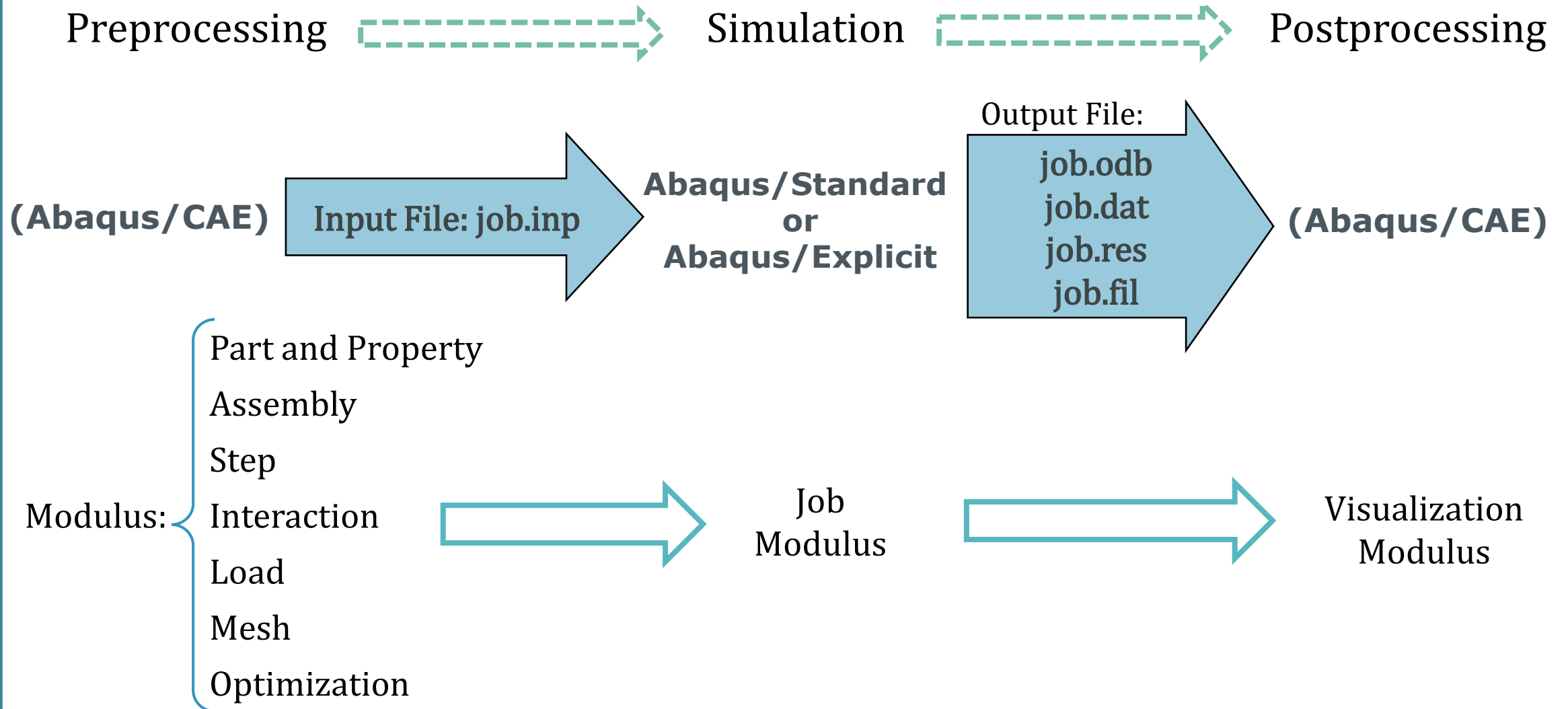


FEA Using Abaqus

FEA Using Abaqus: Abaqus Documentation



FEA Using Abaqus : Basic Concepts



FEA Using Abaqus: Consistent Unit

All input data must be specified in consistent units

Quantity	SI	SI (mm)	US Unit (ft)	US Unit (inch)
Length	m	mm	ft	in
Force	N	N	lbf	lbf
Mass	kg	tonne (10^3 kg)	slug	lbf s ² /in
Time	s	s	s	s
Stress	Pa (N/m ²)	MPa (N/mm ²)	lbf/ft ²	psi (lbf/in ²)
Energy	J	mJ (10^{-3} J)	ft lbf	in lbf
Density	kg/m ³	tonne/mm ³	slug/ft ³	lbf s ² /in ⁴

FEA Using Abaqus: Consistent Unit

LENGTH	MASS	TIME	FORCE	STRESS	ENERGY	VELOCITY	ACCELERATION
mm	ton	S	N	MPa	mJ	1e-03 m/s	1e-03 m/s ²
mm	kg	ms	kN	GPa	1e+03 mJ	m/s	1e+03 m/s ²
mm	g	ms	N	MPa	mJ	m/s	1e+03 m/s ²
mm	kg	S	mN	kPa	1e-03 mJ	1e-03 m/s	1e-03 m/s ²
mm	g	S	1e-06 N	Pa	1e-06 mJ	1e-03 m/s	1e-03 m/s ²
mm	kgf-s ² /mm	S	kgf	kgf/mm ²	kgf-mm	1e-03 m/s	1e-03 m/s ²
m	kg	S	N	Pa	J	m/s	m/s ²
cm	kg	S	1e-02 N	1e+02 Pa	1e-04 J	1e-02 m/s	1e-02 m/s ²
cm	kg	ms	1e+04 N	1e+08 Pa	1e+02 J	1e+01 m/s	1e+04 m/s ²
cm	kg	us	1e+10 N	1e+14 Pa	1e+08 J	1e+04 m/s	1e+10 m/s ²
cm	g	S	dyne	dyne/cm ²	erg	1e-02 m/s	1e-02 m/s ²
cm	g	ms	1e+01 N	bar	1e-01 J	1e+01 m/s	1e+04 m/s ²
cm	g	us	1e+07 N	Mbar	1e+05 J	1e+04 m/s	1e+10 m/s ²
in	lbf-s ² /in	S	lbf	psi	lbf-in	in/s	in/s ²
ft	slug	S	lbf	psf	lbf-ft	ft/s	ft/s ²

FEA Using Abaqus: Consistent Unit

	Commonly used unit	SI value	SI-mm value
Stiffness of steel	210 GPa	$210 \cdot 10^9$ Pa	210000 MPa
Density of steel	7850 kg/m ³		$7.85 \cdot 10^{-9}$ tonne/mm ³
Gravitational constant	9.81 m/s ²		9810 mm/s ²
pressure	1 bar	10^5 Pa	0.1 MPa
Absolute zero temperature	-273.15 °C	0 K	°C and K both acceptable
Stefan-Boltzmann constant	$5.67 \cdot 10^{-8}$ W·m ⁻² ·K ⁻⁴		$5.67 \cdot 10^{-11}$ mW·mm ⁻² ·K ⁻⁴
Universal gas constant	8.31 J·K ⁻¹ ·mol ⁻¹		$8.31 \cdot 10^3$ mJ·K ⁻¹ ·mol ⁻¹



Edit Model Attributes

Name: Model-1

Model type: Standard & Explicit

Description:

☐ Do not use parts and assemblies in input files

Physical Constants

☒ Absolute zero temperature: -273.15

☒ Stefan-Boltzmann constant: 5.67E-11

☒ Universal gas constant: 8314.46

☐ Specify acoustic wave formulation:

Restart Submodel Model Instances

Note: Specify these settings to reuse state data from a previous analysis of this model.

☐ Read data from job:

Restart Location:

Step name:

☒ Restart from the end of the step

☐ Restart from increment, interval, iteration, or cycle:

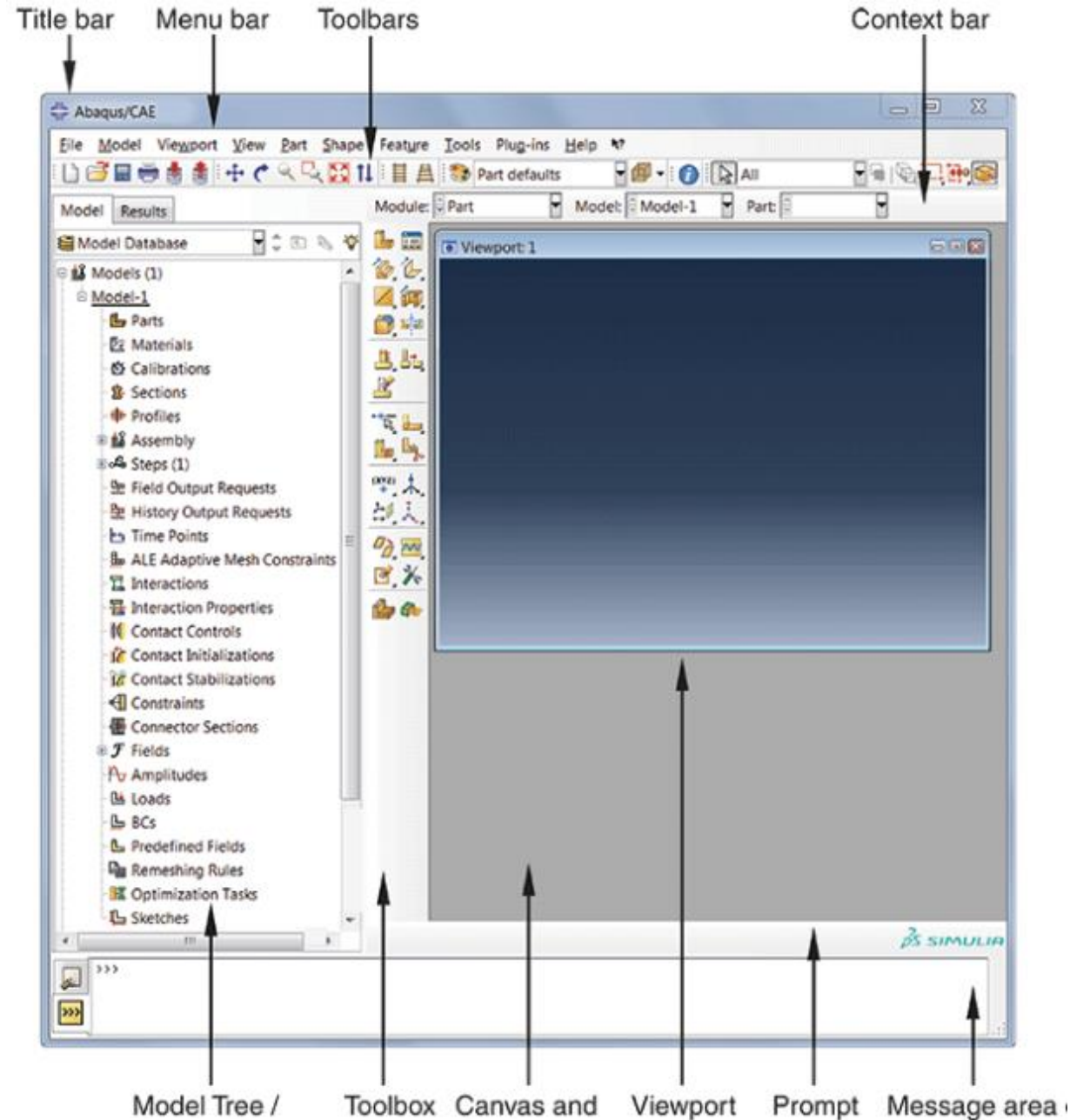
☒ and terminate the step at this point

☐ and complete the step

OK Cancel

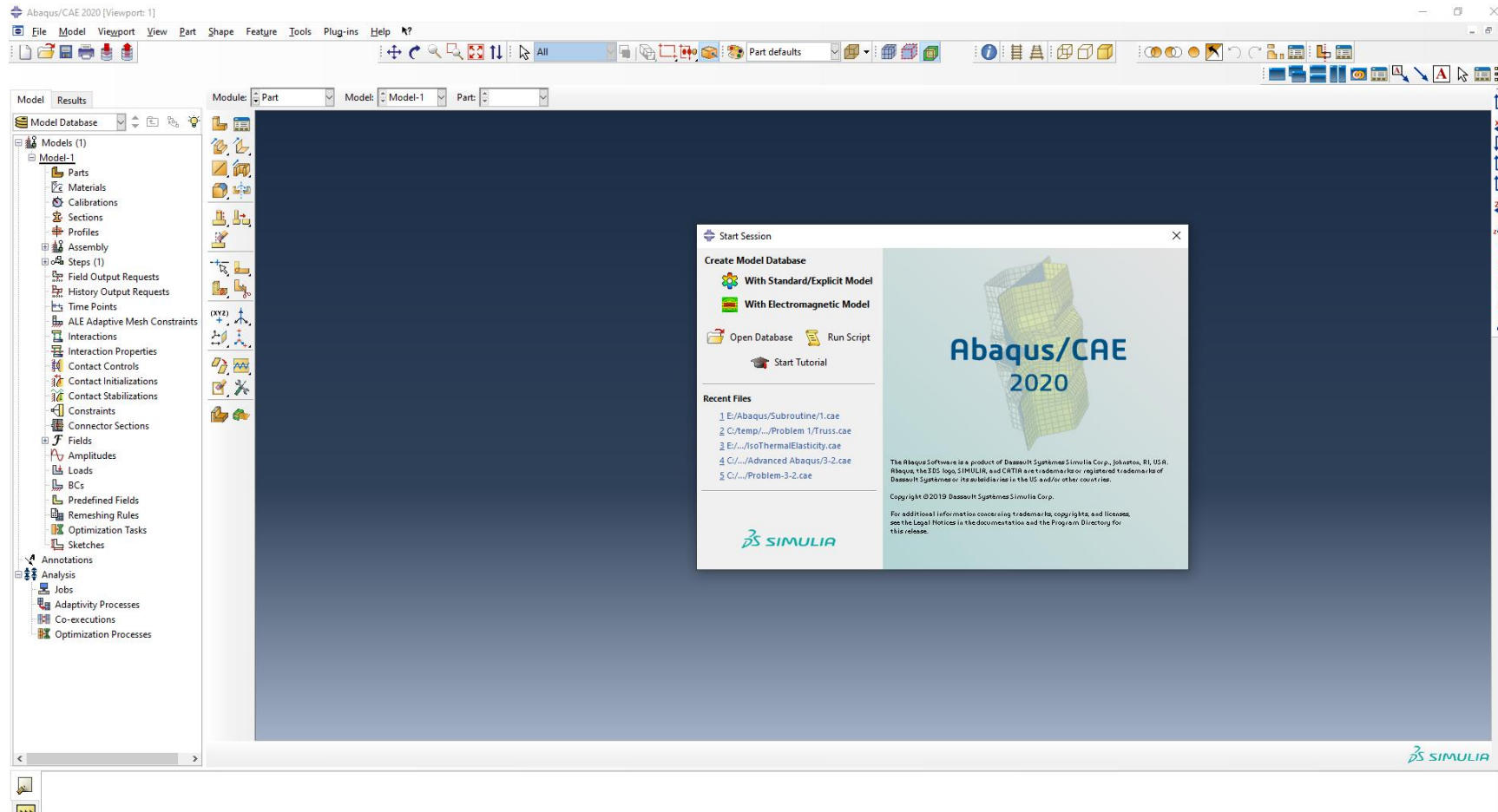
SI-mm

FEA Using Abaqus: Terminology



FEA Using Abaqus: Terminology

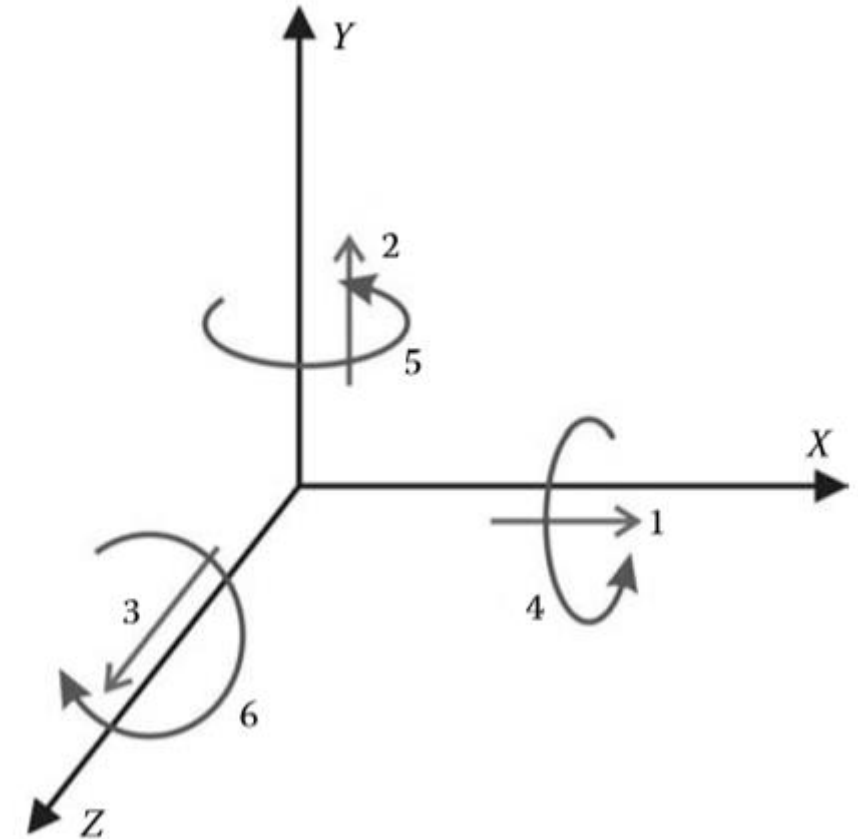
Abaqus/CAE (Complete Abaqus Environment): is an interactive, graphical environment for Abaqus. It allows models to be created quickly and easily by producing or importing the geometry of the structure to be analyzed and **decomposing** the geometry into meshable regions.



FEA Using Abaqus: Conventions

Degrees of freedom

- 1 x -displacement
- 2 y -displacement
- 3 z -displacement
- 4 Rotation about the x -axis, in radians
- 5 Rotation about the y -axis, in radians
- 6 Rotation about the z -axis, in radians
- 7 Warping amplitude (for open-section beam elements)
- 8 Pore pressure, hydrostatic fluid pressure, or acoustic pressure
- 9 Electric potential
- 10 Connector material flow (units of length)
- 11 Temperature (or normalized concentration in mass diffusion analysis)
- 12 Second temperature (for shells or beams)
- 13 Third temperature (for shells or beams)



FEA Using Abaqus: Conventions

Degrees of freedom: Axisymmetric elements

- 1 r -displacement
- 2 z -displacement
- 5 Rotation about the z -axis (for axisymmetric elements with twist), in radians
- 6 Rotation in the r - z plane (for axisymmetric shells), in radians

Here the r - and z -directions coincide with the global X - and Y -directions, respectively

FEA Using Abaqus: Conventions

Symbols used in Abaqus for units

Dimension	Indicator	Example (S.I. units)
length	L	meter
mass	M	kilogram
time	T	second
temperature	θ	degree Celsius
electric current	A	ampere
force	F	newton
energy	J	joule
electric charge	C	coulomb
electric potential	φ	volt
mass concentration	P	Parts per million

FEA Using Abaqus: Conventions

Convention used for stress and strain components

The convention used for stress and strain components in Abaqus is that they are ordered:

$$\begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \tau_{12} \\ \tau_{13} \\ \tau_{23} \end{pmatrix} = \begin{bmatrix} D_{1111} & D_{1122} & D_{1133} & D_{1112} & D_{1113} & D_{1123} \\ & D_{2222} & D_{2233} & D_{2212} & D_{2213} & D_{2223} \\ & & D_{3333} & D_{3312} & D_{3313} & D_{3323} \\ & \text{symm.} & & D_{1212} & D_{1213} & D_{1223} \\ & & & & D_{1313} & D_{1323} \\ & & & & & D_{2323} \end{bmatrix} \begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{33} \\ \gamma_{12} \\ \gamma_{13} \\ \gamma_{23} \end{pmatrix}$$

σ_{11} Direct stress in the 1-direction

σ_{22} Direct stress in the 2-direction

σ_{33} Direct stress in the 3-direction

τ_{12} Shear stress in the 1–2 plane

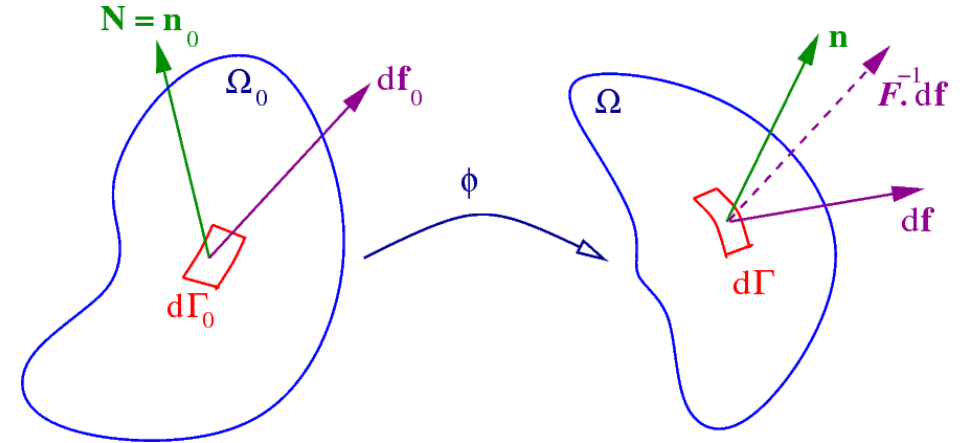
τ_{13} Shear stress in the 1–3 plane

τ_{23} Shear stress in the 2–3 plane

FEA Using Abaqus: Conventions

Stress Measurements

σ : Cauchy Stress (True Stress) is defined to be the **current** force per unit **deformed area**.



\mathbf{P} : First Piola-Kirchhoff stress tensor (known as the Lagrangian stress tensor or transpose of Nominal stress) is defined to be the **current** force per unit **undeformed area**.

\mathbf{S} : Second Piola-Kirchhoff stress is defined to be the **initial** (transformed current) force per unit **undeformed area**.

FEA Using Abaqus: Conventions

Stress Measurements

Conversion formulae

[hide]

	σ	P	S	τ	T	M
$\sigma =$	σ	$J^{-1} P F^T$	$J^{-1} F S F^T$	$J^{-1} \tau$	$J^{-1} R T F^T$	$J^{-1} F^{-T} M F^T$ (non isotropy)
$P =$	$J \sigma F^{-T}$	P	$F S$	τF^{-T}	$R T$	$F^{-T} M$
$S =$	$J F^{-1} \sigma F^{-T}$	$F^{-1} P$	S	$F^{-1} \tau F^{-T}$	$U^{-1} T$	$C^{-1} M$
$\tau =$	$J \sigma$	$P F^T$	$F S F^T$	τ	$R T F^T$	$F^{-T} M F^T$ (non isotropy)
$T =$	$J R^T \sigma F^{-T}$	$R^T P$	$U S$	$R^T \tau F^{-T}$	T	$U^{-1} M$
$M =$	$J F^T \sigma F^{-T}$ (non isotropy)	$F^T P$	$C S$	$F^T \tau R^{-T}$ (non isotropy)	$U T$	M

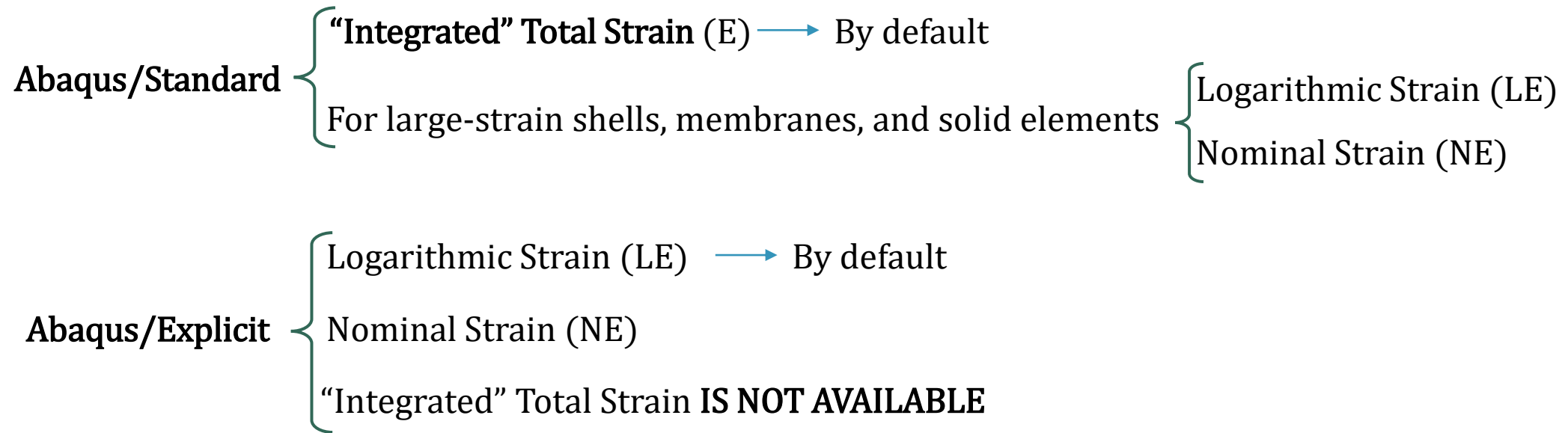
$$J = \det(F), \quad C = F^T F = U^2, \quad F = R U, \quad R^T = R^{-1}$$

$$P = J \sigma F^{-T}, \quad \tau = J \sigma, \quad S = J F^{-1} \sigma F^{-T}, \quad T = R^T P, \quad M = C S$$

FEA Using Abaqus: Conventions

Strain Measurements

For geometrically nonlinear analysis, a large number of different strain measures exist. Unlike “true” stress, there is no clearly preferred “true” strain.



FEA Using Abaqus: Conventions

Strain Measurements: Total (integrated) Strain

Is obtained by integrating the strain rate numerically in a **material frame of reference**:

Incremental rotation tensor

$$\boldsymbol{\epsilon}^{n+1} = \Delta \mathbf{R} \cdot \boldsymbol{\epsilon}^n \cdot \Delta \mathbf{R}^T + \Delta \boldsymbol{\epsilon}$$

Rate of deformation

$$\Delta \boldsymbol{\epsilon} = \int_{t^n}^{t^{n+1}} \mathbf{D} dt.$$

Element with corotational coordinate system
(finite-strain shells, membranes, and solid elements with user-defined orientations)


$$\boldsymbol{\epsilon}^{n+1} = \boldsymbol{\epsilon}^n + \Delta \boldsymbol{\epsilon}.$$

FEA Using Abaqus: Conventions

Strain Measurements: Green's strain

For small-strain shells and beams in Abaqus/Standard, the default strain measure, E, is Green's strain:

Deformation gradient

$$\boldsymbol{\epsilon}^G = \frac{1}{2} (\mathbf{F}^T \cdot \mathbf{F} - \mathbf{I})$$


This strain measure is appropriate for the small-strain, large-rotation approximation used in these elements.

FEA Using Abaqus: Conventions

Strain Measurements

Nominal strain

$$\epsilon^N = \mathbf{V} - \mathbf{I} = \sum_{i=1}^3 (\lambda_i - 1) \mathbf{n}_i \mathbf{n}_i^T,$$

The Principal Stretches

The principal stretch directions

Logarithmic strain

$$\epsilon^L = \ln \mathbf{V} = \sum_{i=1}^3 \ln \lambda_i \mathbf{n}_i \mathbf{n}_i^T,$$

Strain output for hyperelastic materials.

For a hyper-viscoelastic material, the logarithmic elastic strain EE is computed from the current (relaxed) stress state, and the viscoelastic strain CE is computed as LE – EE.

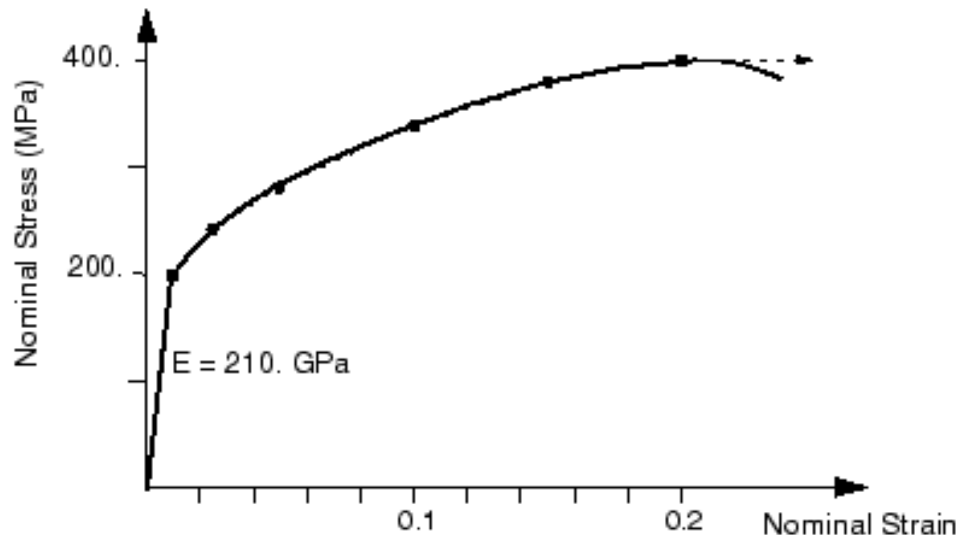
FEA Using Abaqus: Conventions

Defining Plasticity in Abaqus

True Stress and True Strain

$$\begin{cases} \varepsilon = \ln(1 + \varepsilon_{nom}) \\ \sigma = \sigma_{nom} (1 + \varepsilon_{nom}) \end{cases}$$

$$\begin{array}{c} \text{True elastic strain} \quad \text{Young's modulus} \\ \varepsilon^{pl} = \varepsilon^t - \varepsilon^{el} = \varepsilon^t - \sigma/E \\ \text{True plastic strain} \quad \text{True total strain} \quad \text{True stress} \end{array}$$

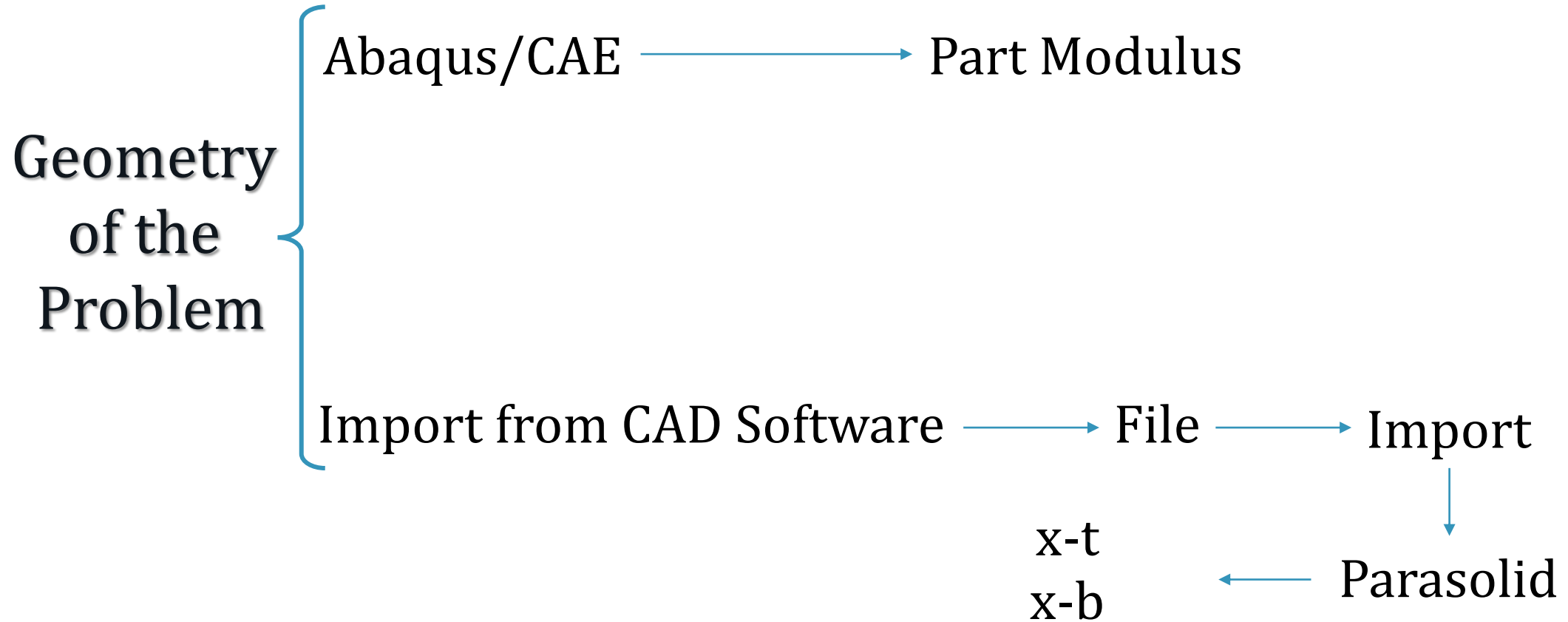


Nominal Stress (Pa)	Nominal Strain	True Stress (Pa)	True Strain	Plastic Strain
200E6	0.00095	200.2E6	0.00095	0.0
240E6	0.025	246E6	0.0247	0.0235
280E6	0.050	294E6	0.0488	0.0474
340E6	0.100	374E6	0.0953	0.0935
380E6	0.150	437E6	0.1398	0.1377
400E6	0.200	480E6	0.1823	0.1800

FEA Using Abaqus

Part

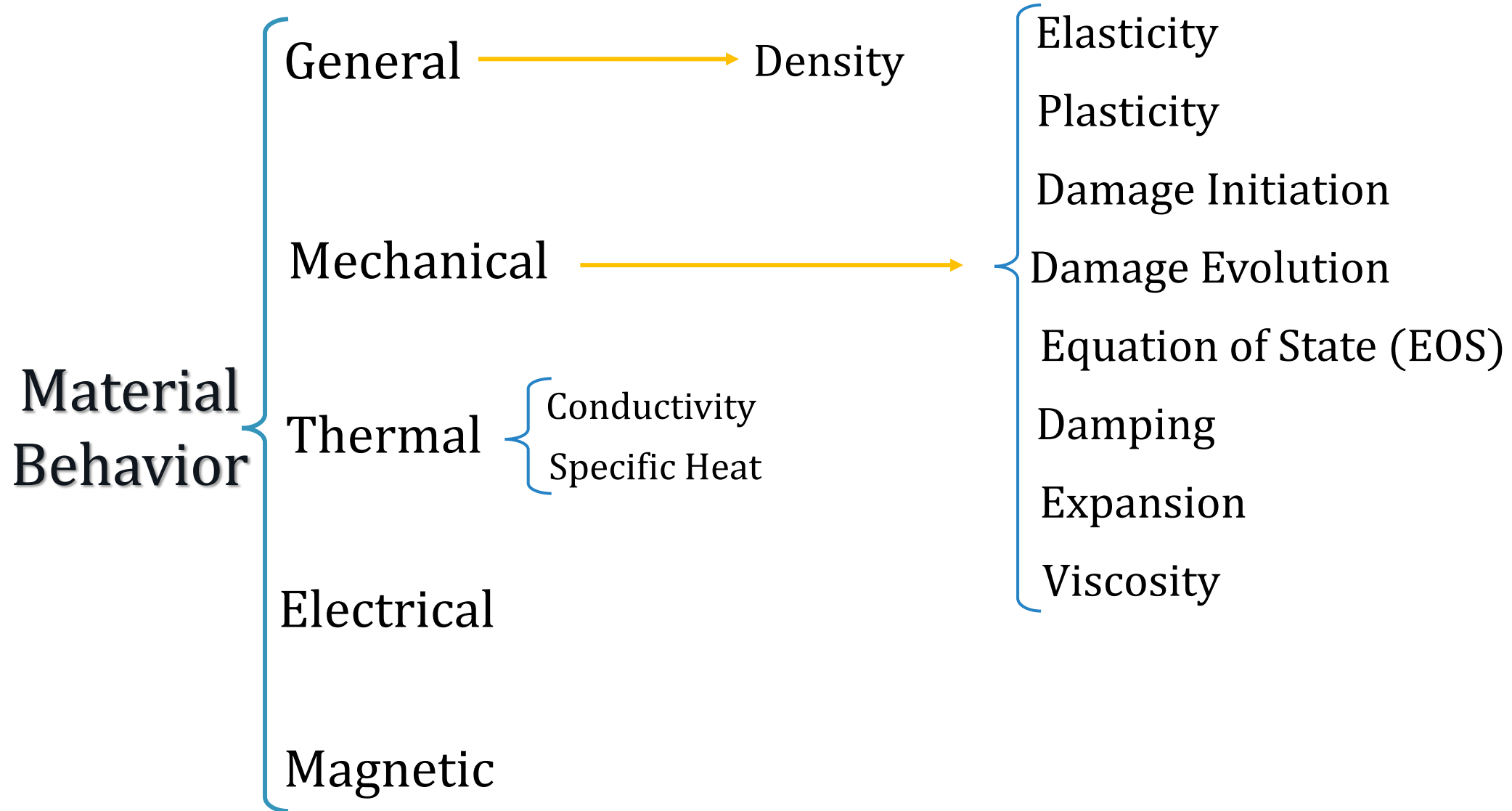
FEA Using Abaqus: Part



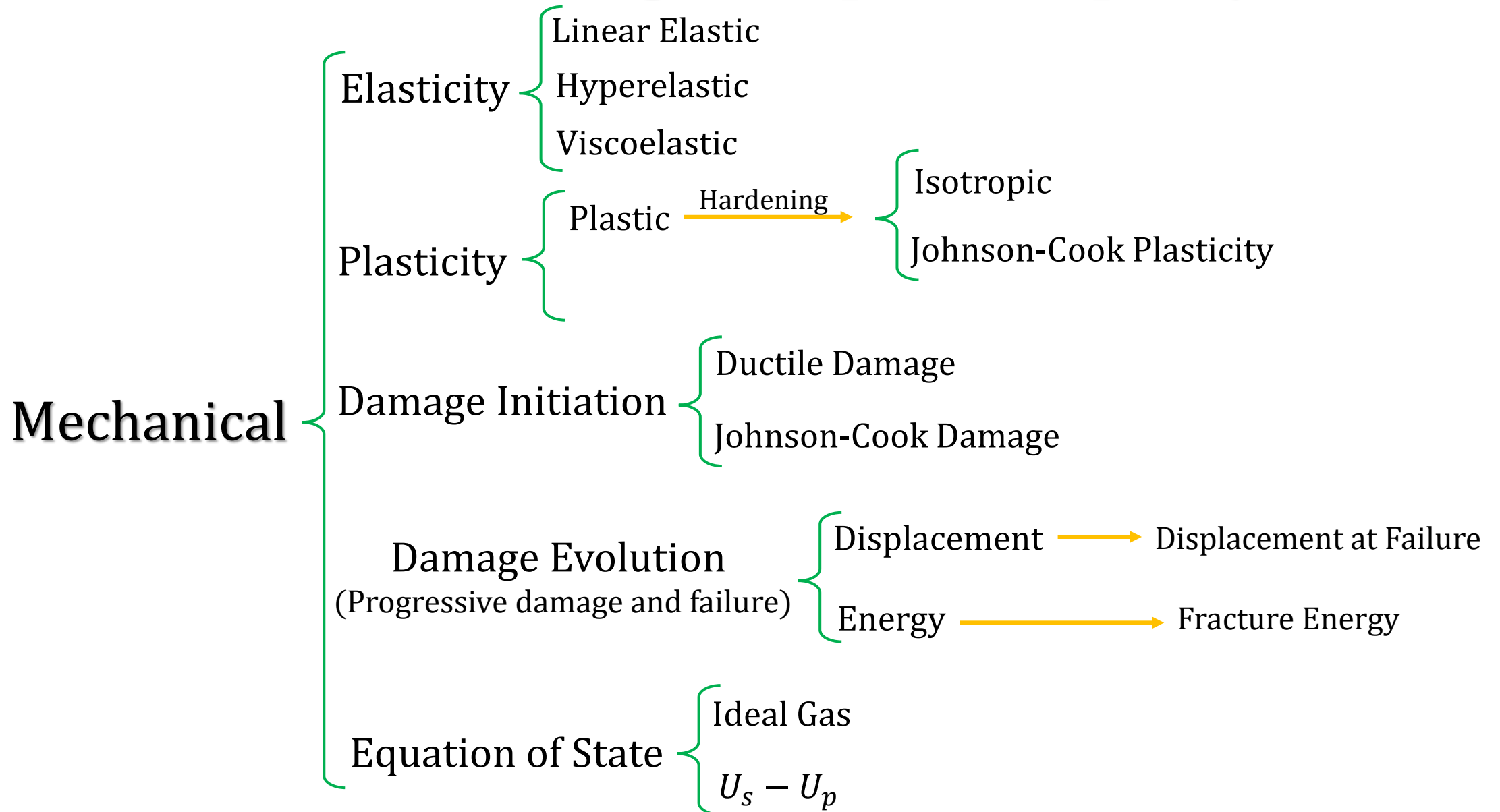
FEA Using Abaqus

Property

FEA Using Abaqus: Property



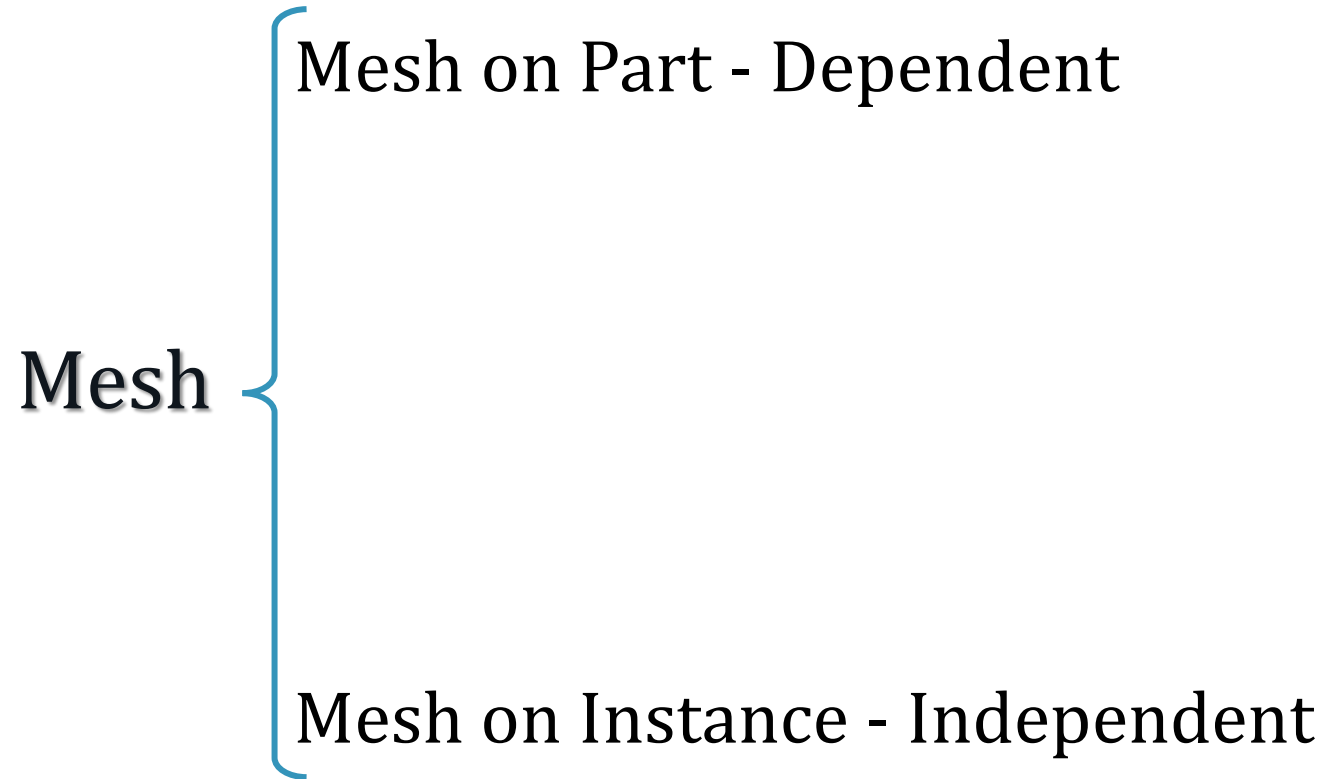
FEA Using Abaqus: Property



FEA Using Abaqus

Assembly

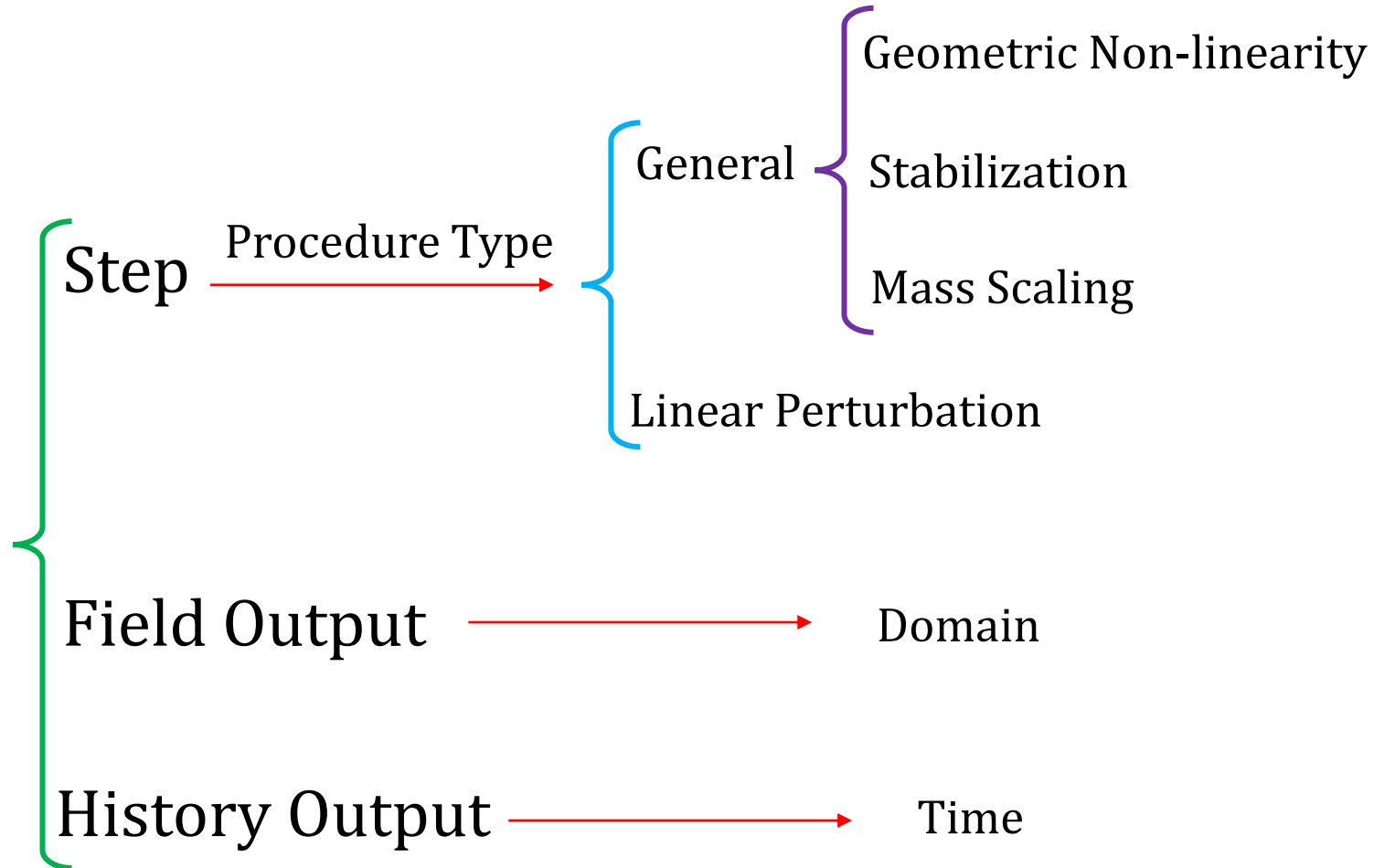
FEA Using Abaqus: Assembly



FEA Using Abaqus

Step

FEA Using Abaqus: Step



FEA Using Abaqus

Interaction

FEA Using Abaqus

Continuum coupling constraint couples motion of the reference node to the average translation of the coupling nodes. It distributes the forces and moments at the reference node as the coupling nodes forces only.

Structural coupling constraint couples the motion of the reference node to the translation and rotation of the coupling nodes. It distributes the forces and moments at the reference node as the coupling nodes forces and moments.

FEA Using Abaqus: Interaction

Contact
Model

Mesh on Part - Dependent

Mesh on Instance - Independent

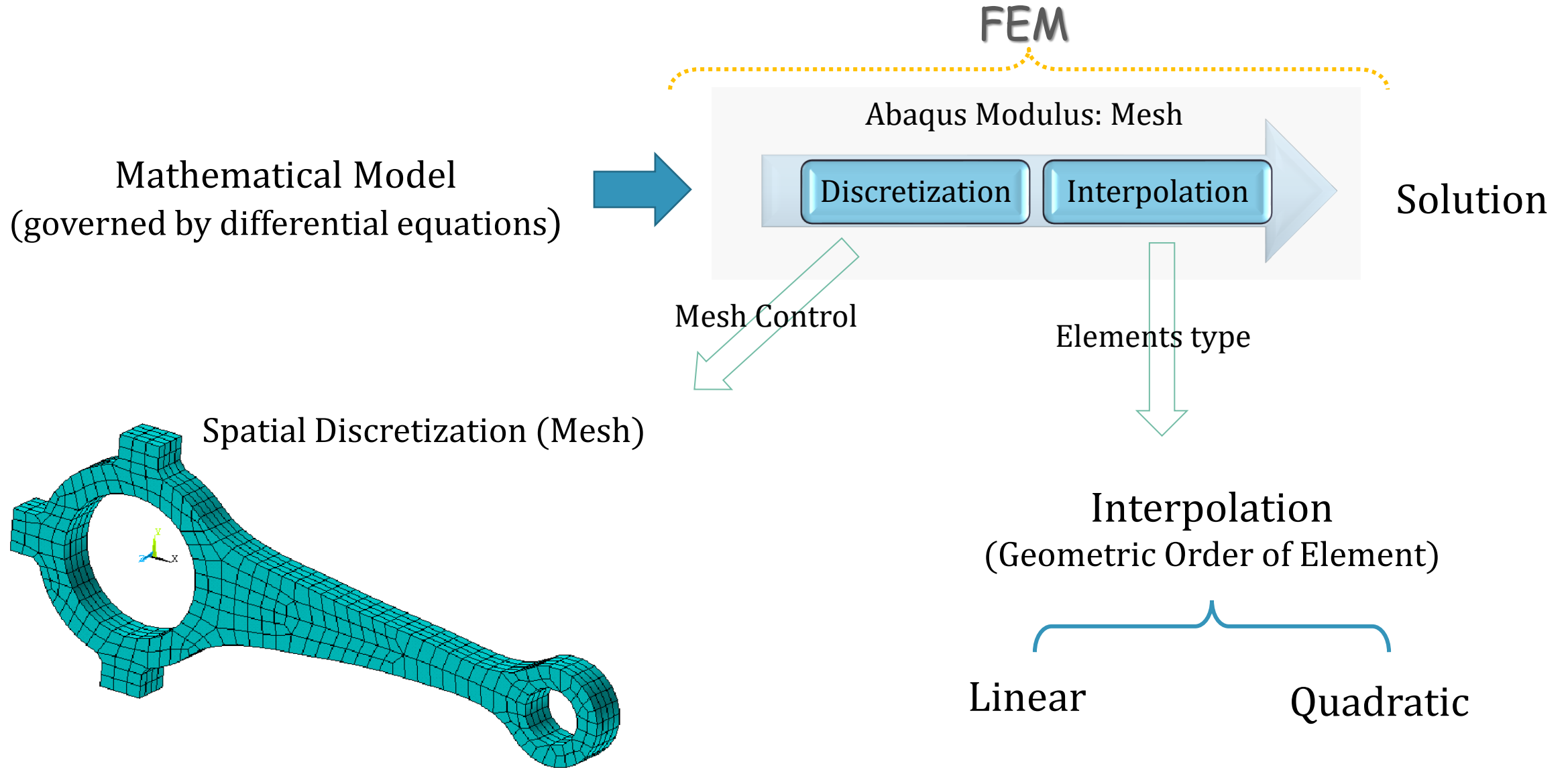
FEA Using Abaqus

Load

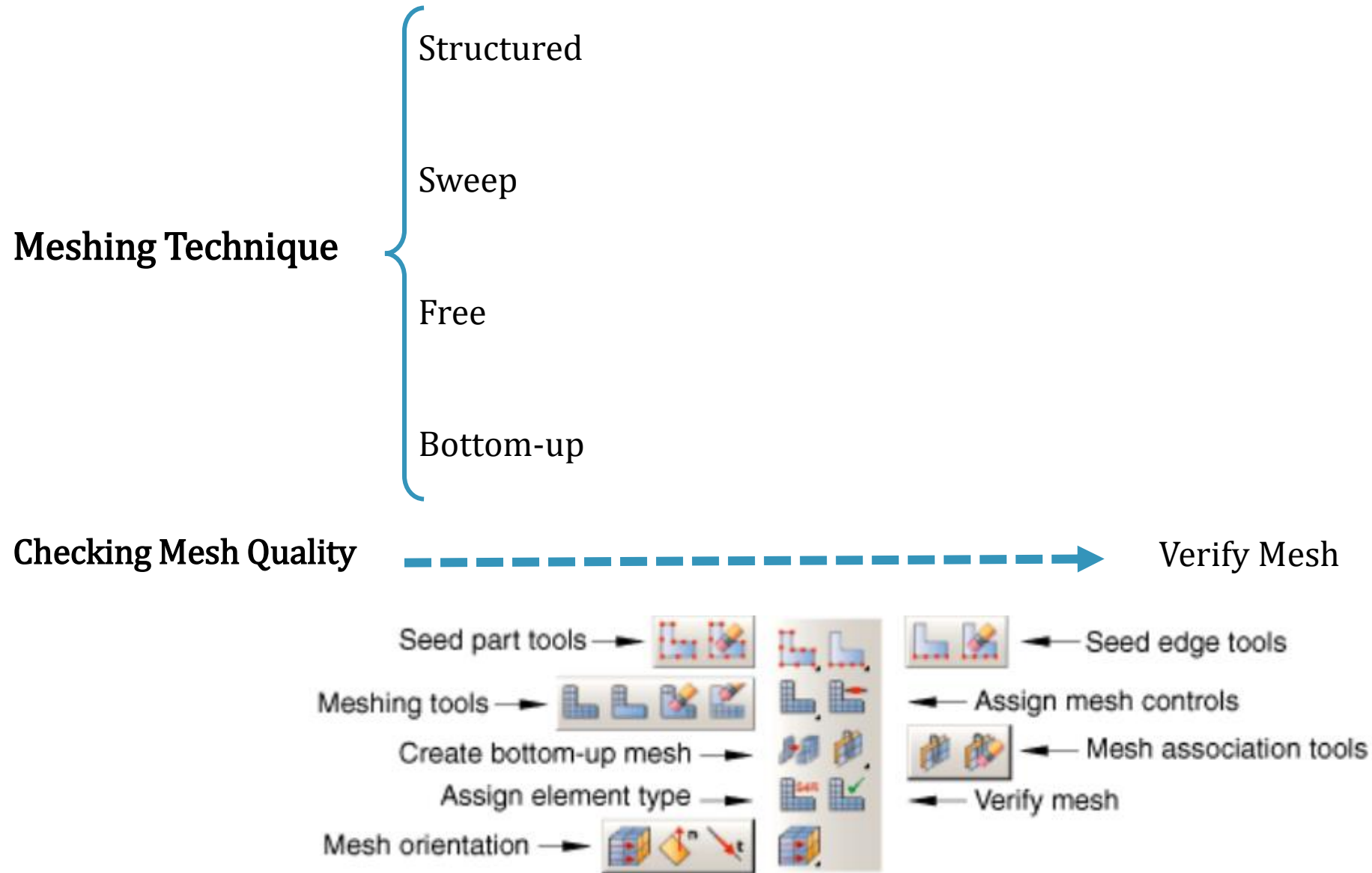
FEA Using Abaqus

Mesh

FEA Using Abaqus: Mesh



FEA Using Abaqus: Mesh Controls



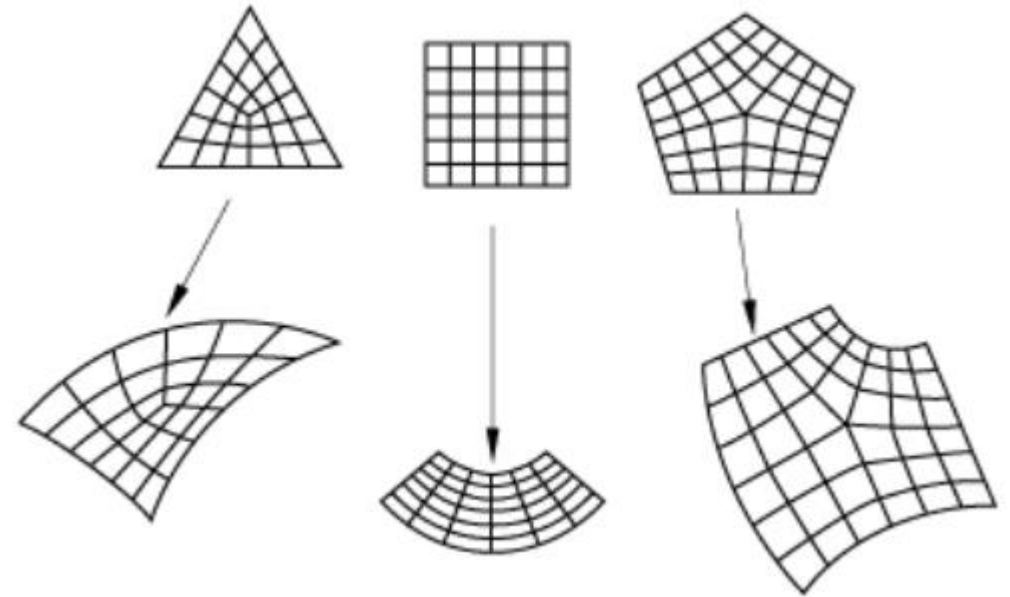
FEA Using Abaqus: Mesh Controls

Structured: generates structured meshes using simple predefined mesh topologies. Abaqus/CAE transforms the mesh of a regularly shaped region, such as a square or a cube, onto the geometry of the region you want to mesh.

Sweep

Free

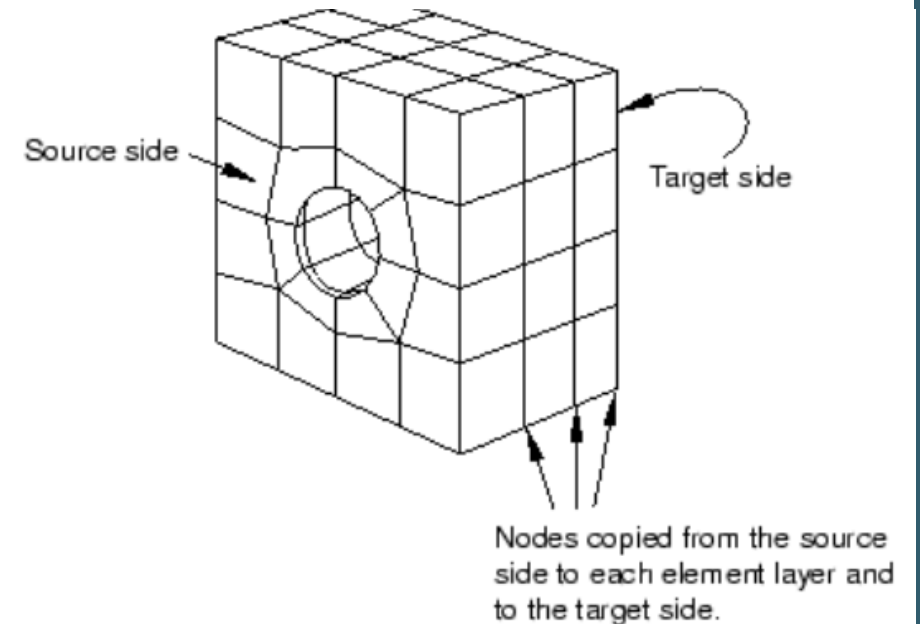
Bottom-up



FEA Using Abaqus: Mesh Controls

Structured

Sweep: Abaqus/CAE creates a mesh on one side of the region, known as the **source side**. And copies the nodes of that mesh, one element layer at a time, until the final side, known as the target side, is reached. Abaqus/CAE copies the nodes along an edge, and this edge is called the sweep path. The sweep path can be any type of edge—a straight edge, a circular edge, or a spline. If the sweep path is a straight edge or a spline, the resulting mesh is called an extruded swept mesh. If the sweep path is a circular edge, the resulting mesh is called a revolved swept mesh.



Free

Bottom-up

FEA Using Abaqus: Mesh Controls

Structured

Sweep

Free: free meshing uses no preestablished mesh patterns as a result, it is impossible to predict a free mesh pattern before creating the mesh.

Bottom-up

FEA Using Abaqus: Mesh Controls

Structured

Sweep

Free

Bottom-up: is a **manual**, incremental meshing process that allows you to build a hexahedral mesh in any solid region.

FEA Using Abaqus: Mesh Recommendations

Abaqus/Standard
&
Abaqus/Explicit

Make all elements as “well shaped” as possible → Verify Mesh

If an automatic tetrahedral mesh generator is used, use the second-order elements C3D10 (in Abaqus/Standard) or C3D10 and C3D10M (in Abaqus/Explicit).

Use the modified tetrahedral element C3D10M in Abaqus/Standard in analyses with large amounts of plastic deformation.

Use hexahedral elements in three-dimensional analyses since they give the best results for the minimum cost

FEA Using Abaqus: Mesh Recommendations

Abaqus/Standard

For linear and “smooth” nonlinear problems use **reduced-integration, second-order elements**

Use second-order, fully integrated elements close to stress concentrations to capture the severe gradients in these regions. However, avoid these elements in regions of finite strain if the material response is nearly incompressible.

Use first-order quadrilateral or hexahedral elements or the modified triangular and tetrahedral elements for problems involving large distortions.

If the mesh distortion is severe, use reduced-integration, first-order elements.

If the problem involves bending and large distortions, use a fine mesh of first-order, reduced-integration elements.

Incompatible mode elements can give very accurate results in problems dominated by bending

Hybrid elements must be used if the material is fully incompressible (except when using plane stress elements). Hybrid elements should also be used in some cases with nearly incompressible materials.

FEA Using Abaqus: Element Types

Five aspects of an element characterize its behavior:

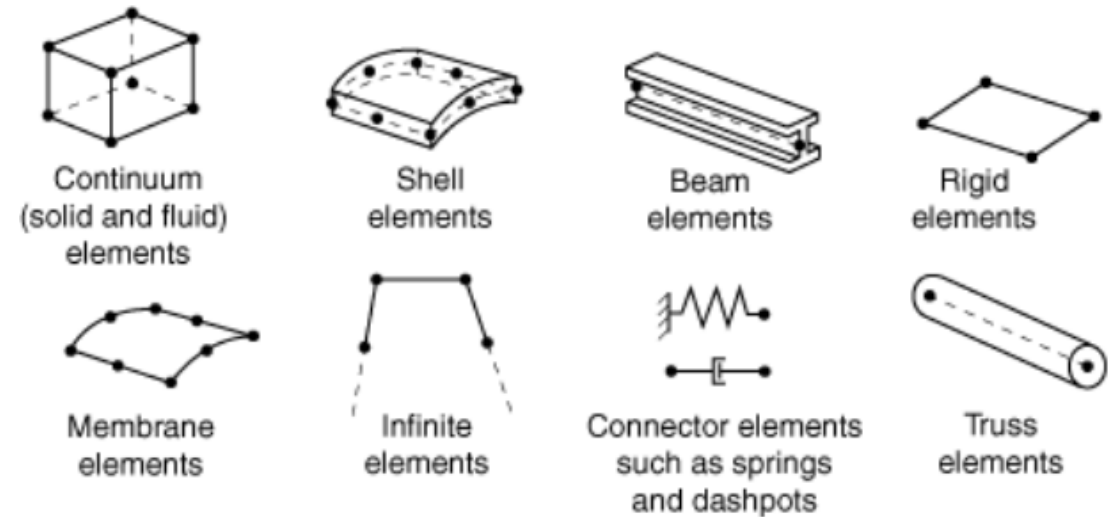
Family

Degrees of freedom Number of nodes

Number of nodes and order of interpolation

Formulation

Integration



The **first** letter or letters of an element's name indicate to which **family** the element belongs. For example, S4R is a shell element and C3D8I is a continuum element.

FEA Using Abaqus: Element Types

Element Family

Continuum Elements

Structural Elements

Inertial, Rigid, and Capacitance Elements

Connector Elements

Particle Elements

Special-Purpose Elements

Truss Element

Beam Element

Frame Element

Elbow Element

Membrane Element

Shell Element

Shear Panel Element

FEA Using Abaqus: Element Types

Continuum Elements

FEA Using Abaqus: Element Types

Structural Elements



Truss Element

Beam Element

Frame Element

Elbow Element

Membrane Element

Shell Element

Shear Panel Element

FEA Using Abaqus: Element Types

Beam Element

First-order, shear-deformable (Timoshenko) beam elements (B21, B31) should be used in any simulation that includes contact.

If the transverse shear deformation is important, use Timoshenko beam elements (B21, B22, B31, B32).

If the structure is either very rigid or very flexible, the hybrid beam elements (B21H, B32H, etc.) available in Abaqus/Standard should be used in geometrically nonlinear simulations.

The Euler-Bernoulli (cubic) beams (B23, B33) available in Abaqus/Standard are very accurate for simulations that include distributed loading, such as dynamic vibration analyses.

Structures with open, thin-walled cross-sections should be modeled with the elements that use open-section warping theory (B31OS, B32OS) available in Abaqus/Standard

FEA Using Abaqus: Element Types

Five aspects of an element characterize its behavior:

Family

Degrees of freedom Number of nodes: the translations and, for shell, pipe, and beam elements, the rotations at each node.

Number of nodes and order of interpolation

Formulation

Integration

FEA Using Abaqus: Element Types

Five aspects of an element characterize its behavior:

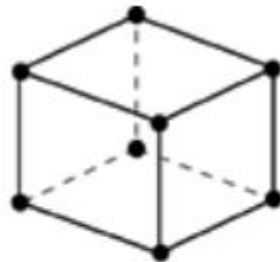
Family

Degrees of freedom Number of nodes

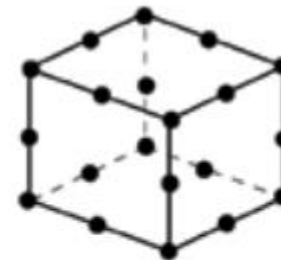
Number of nodes and order of interpolation

Formulation

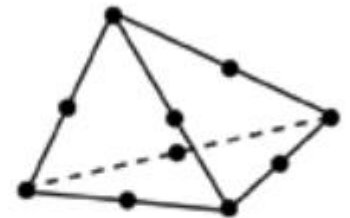
Integration



(a) Linear element
(8-node brick, C3D8)



(b) Quadratic element
(20-node brick, C3D20)



(c) Modified second-order element
(10-node tetrahedron, C3D10M)

FEA Using Abaqus: Element Types

Five aspects of an element characterize its behavior:

Family

Degrees of freedom Number of nodes

Number of nodes and order of interpolation

Formulation: mathematical theory used to define the element's behavior (Lagrangian or Eulerian/shell element: 1-general-purpose shell analysis, 2-thin shells, 3-for thick shells.)

Integration

Plane strain

Plane stress

Hybrid elements

Incompatible-mode elements

Small-strain shells

Finite-strain shells

Thick shells

Thin shells

Hybrid Element Formulation

Fully Incompressible

When the material response is **incompressible**, the solution to a problem cannot be obtained in terms of the displacement history only, since a purely hydrostatic pressure can be added without changing the displacements.

Almost Incompressible

The nearly incompressible case exhibits behavior approaching this limit, in that a very small change in displacement produces extremely large changes in pressure, so that a purely displacement-based solution is **too sensitive** to be useful numerically



We remove this singular behavior in the system by treating the **pressure stress** as an independently interpolated basic solution variable, coupled to the displacement solution through the constitutive theory and the compatibility condition, with this coupling implemented by a **Lagrange multiplier**.

Hybrid Element Formulation

FEA Using Abaqus: Element Types


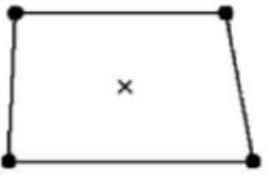
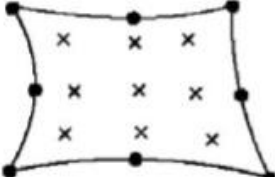
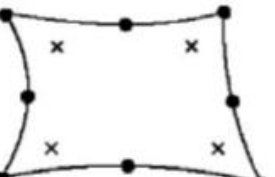
Five aspects of an element characterize its behavior:

Family

Degrees of freedom Number of nodes

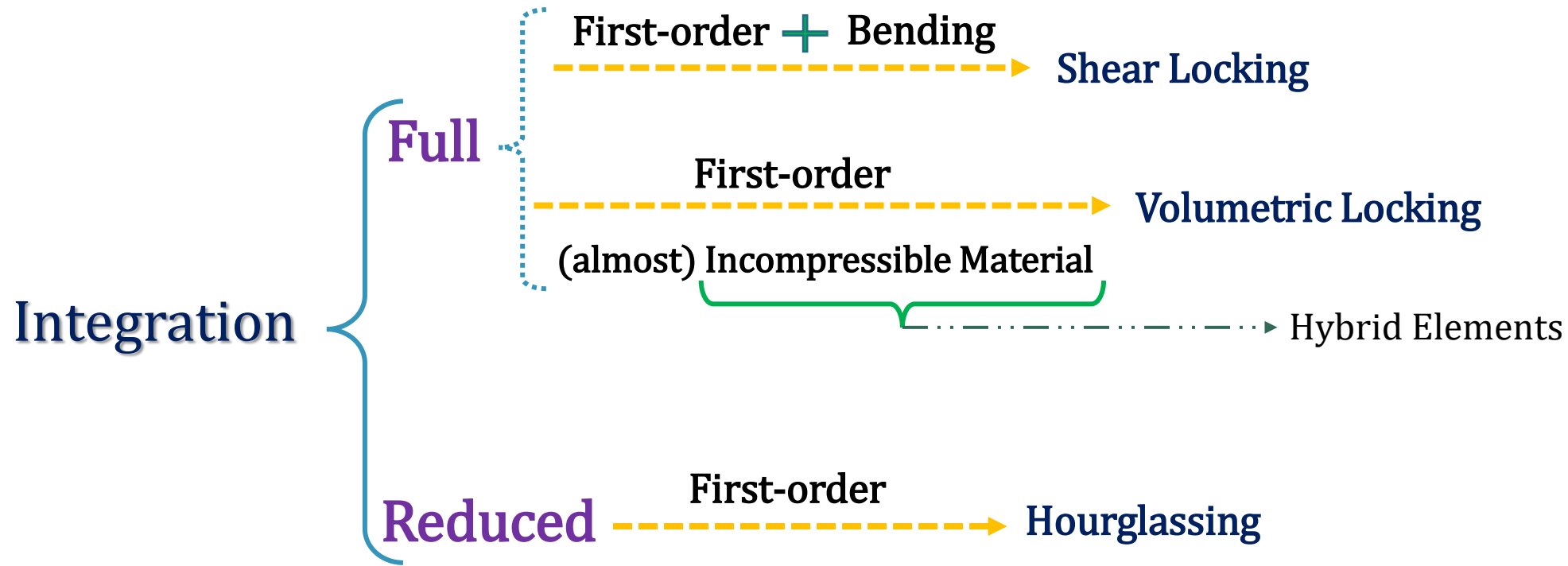
Number of nodes and order of interpolation

Formulation

	Full integration	Reduced integration
First-order interpolation		
Second-order interpolation		

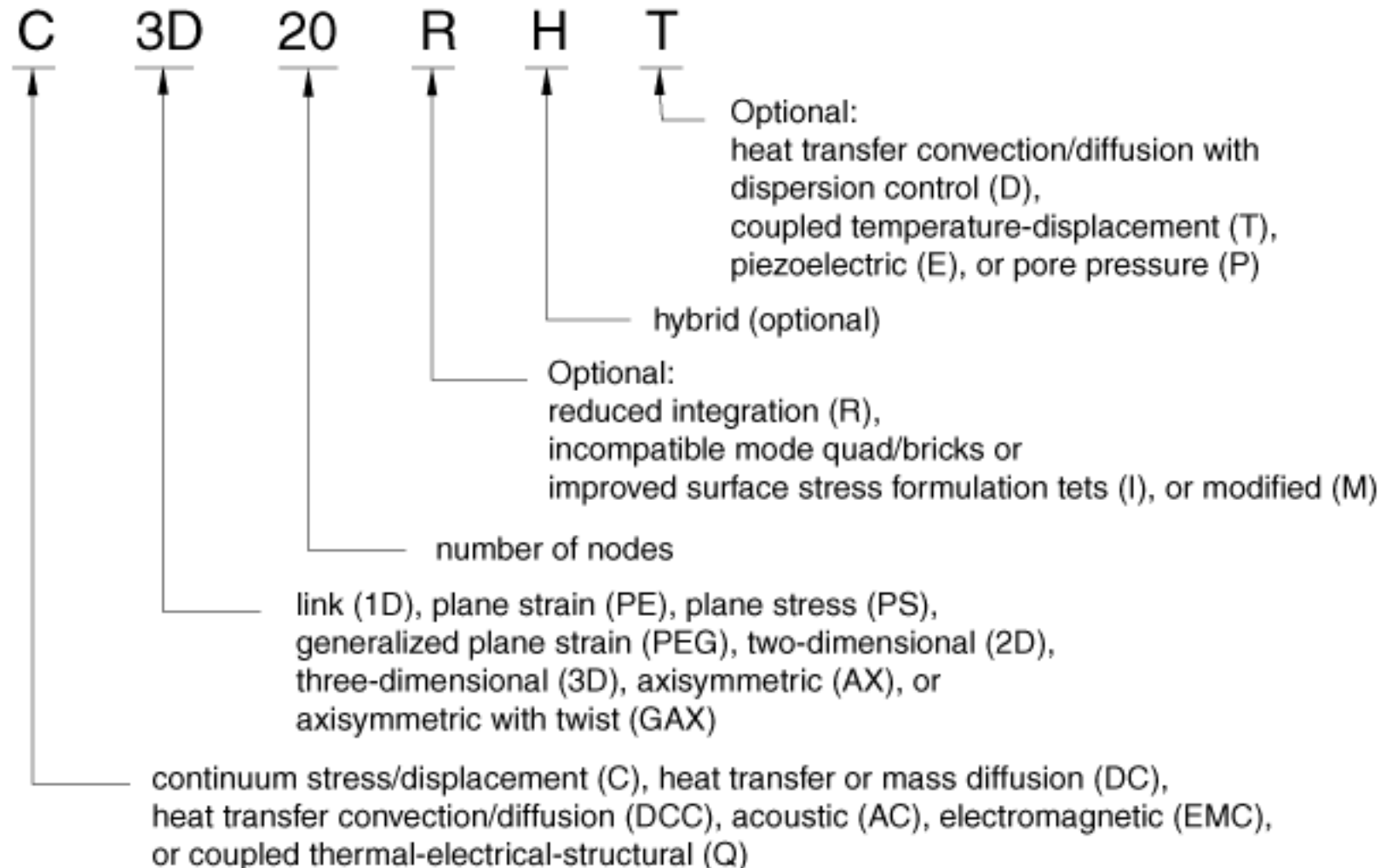
Integration: Using Gaussian quadrature for most elements (full or reduced integration)

FEA Using Abaqus: Element Types



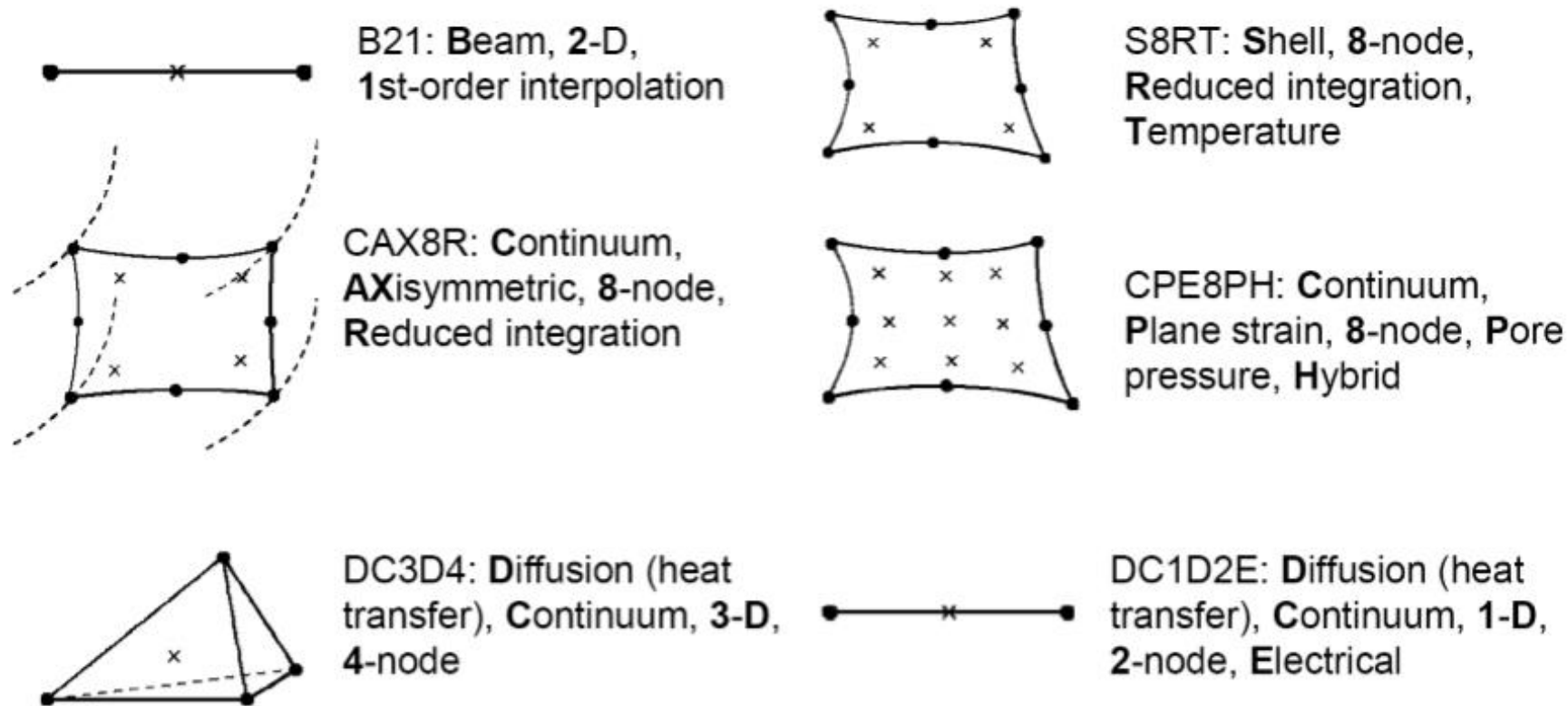
FEA Using Abaqus: Element Types

Element naming convention



FEA Using Abaqus: Element Types

Element naming convention: Example



FEA Using Abaqus

Optimization

FEA Using Abaqus

Job

FEA Using Abaqus

Visualization

FEA Using Abaqus: Abaqus Solvers

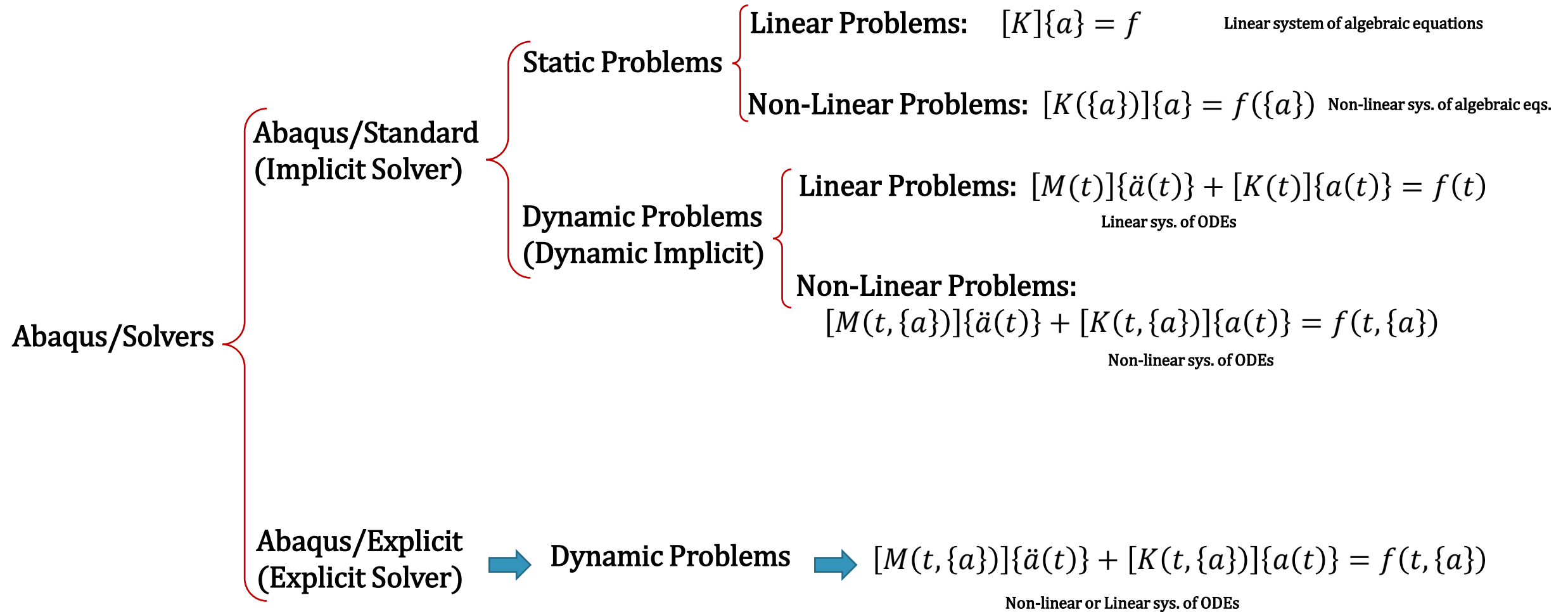
Non-linear Structural Problems

Material Nonlinearity: Due to non-linear constitutive law(e.g., polymer materials)

Geometric Nonlinearity: Due to Large displacements or large rotations

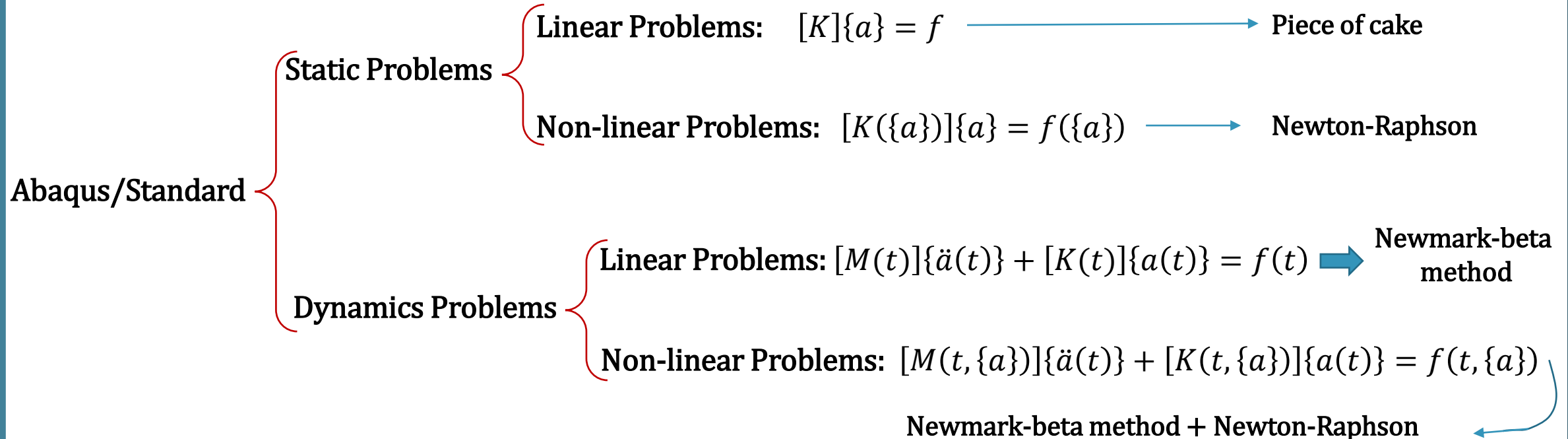
Boundary Nonlinearity: Due to the non-linearity of boundary conditions (i.e., contact problems)

FEA Using Abaqus: Abaqus Solvers



FEA Using Abaqus: Abaqus Solvers

Abaqus/Standard: is a **general-purpose** analysis product (solver) that can solve a wide range of linear and nonlinear problems involving the **static, dynamic, thermal, electrical, and electromagnetic** response of components. Abaqus/Standard solves a system of equations implicitly at each solution “increment.” In contrast, Abaqus/Explicit marches a solution forward through time in small time increments without solving a coupled system of equations at each increment (or even forming a global stiffness matrix). Abaqus/Standard is ideal for static and low-speed dynamic events where highly accurate stress solutions are important.



FEA Using Abaqus: Abaqus Solvers

Abaqus/Explicit: is a **special-purpose** analysis product that uses an **explicit dynamic finite element** formulation. It is suitable for modeling **brief, transient dynamic** events, such as **impact and blast** problems, and is also very efficient for highly nonlinear problems involving changing **contact conditions**, such as **forming** simulations. Abaqus/Explicit is ideal for analyses where high-speed, non-linear, transient response dominates the solution.

Abaqus/Explicit
(Explicit Solver) → Dynamic Problems → $[M(t, \{a\})]\{\ddot{a}(t)\} + [K(t, \{a\})]\{a(t)\} = f(t, \{a\})$

Non-linear or Linear sys. of ODEs

FEA Using Abaqus: Abaqus Solvers

Abaqus/Standard (Implicit Method)

Implicit Method involves the solution for the current step is based on the solution from the previous step.

Mathematically, if $a(t_i)$ is the current system state and $a(t_{i+1})$ is the state at the later time, then, for an implicit method:

$$[M(t, \{a\})]\{\ddot{a}(t)\} + [K(t, \{a\})]\{a(t)\} = f(t, \{a\})$$

$$a(t_{i+1}) = G(a(t_{i+1}), a(t_i)) \longrightarrow a(t)$$

These solutions are **unconditionally stable** and facilitate larger time steps. Despite this advantage, the implicit methods can be extremely time-consuming when solving dynamic and nonlinear problems.

vs.

Abaqus/Explicit (Explicit Method)

Explicit Method aim to solve acceleration and in most cases, the mass matrix is considered as “lumped” and consequently mass matrix is diagonal matrix.

Explicit methods calculate the state of a system at a later time from the state of the system at the current time.

$$[M(t, \{a\})]\{\ddot{a}(t)\} + [K(t, \{a\})]\{a(t)\} = f(t, \{a\})$$

$$\ddot{a}(t_{i+1}) = F(a(t_i)) \longrightarrow \ddot{a}(t_{i+1})$$

Once the accelerations are calculated at the nth step, the velocity at n+1/2 step and displacement at n+1 step are calculated accordingly. In these calculations, the scheme is **not unconditionally stable** and thus smaller time steps are required.

FEA Using Abaqus: Abaqus Solvers

Abaqus/Standard (Implicit Method) **vs.** Abaqus/Explicit (Explicit Method)

$$[M]\{x''\} + [C]\{x'\} + [K]\{x\} = \{f\}$$

$$[K]\{x\} = \{f\} - ([M]\{x''\} + [C]\{x'\})$$

$$[K]^{-1}[K]\{x\} = [K]^{-1}(\{f\} - ([M]\{x''\} + [C]\{x'\}))$$

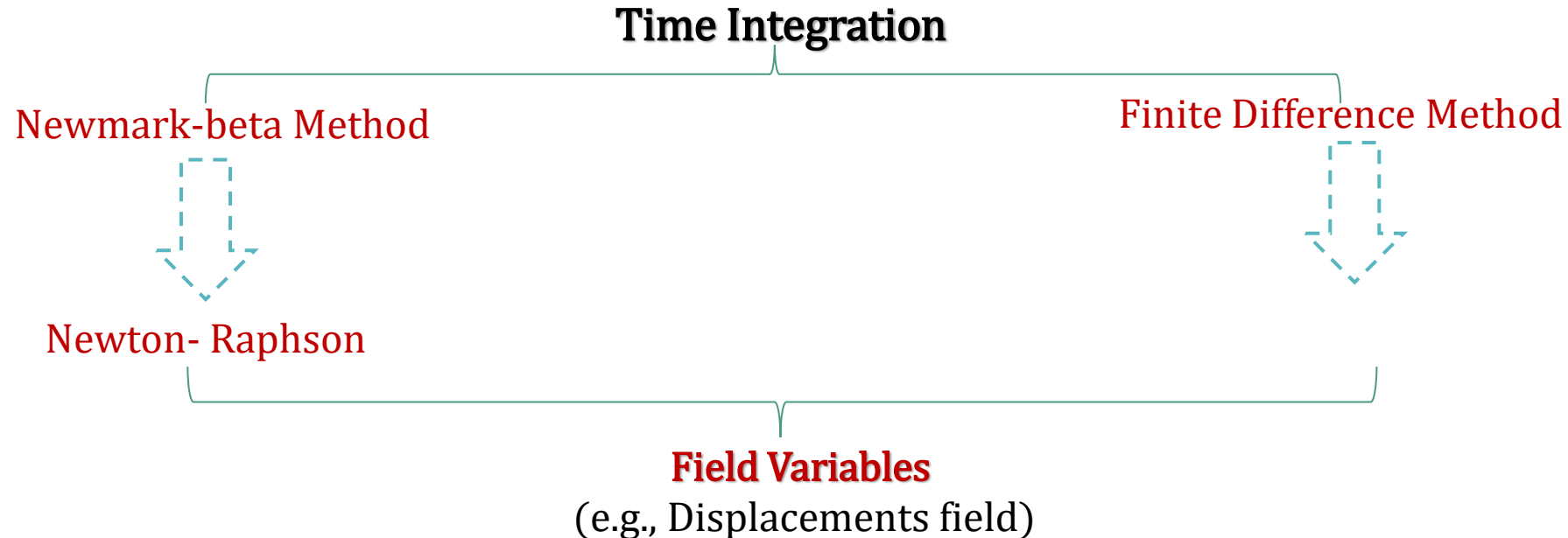
$$\{x\} = [K]^{-1}(\{f\} - ([M]\{x''\} + [C]\{x'\}))$$

$$[M]\{x'\} + [C]\{x'\} + [K]\{x\} = \{f\}$$

$$[M]\{x'\} = \{f\} - ([C]\{x'\} + [K]\{x\})$$

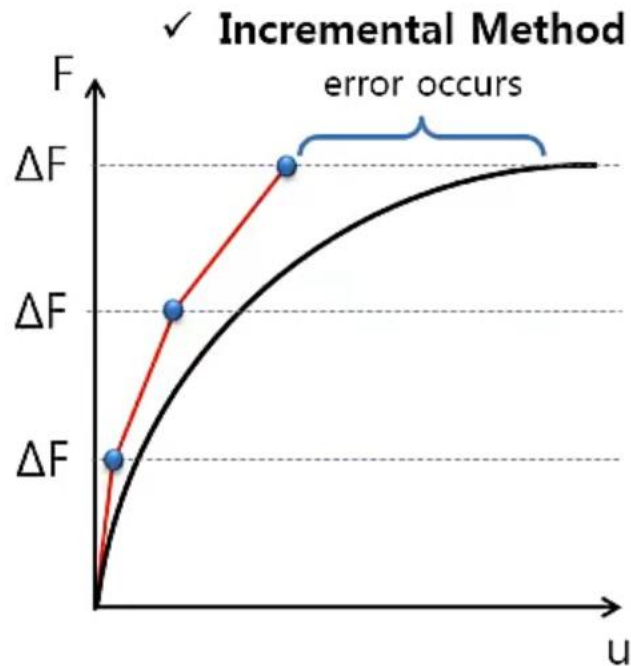
$$[M]^{-1}[M]\{x'\} = [M]^{-1}(\{f\} - ([C]\{x'\} + [K]\{x\}))$$

$$\{x'\} = [M]^{-1}(\{f\} - ([C]\{x'\} + [K]\{x\}))$$

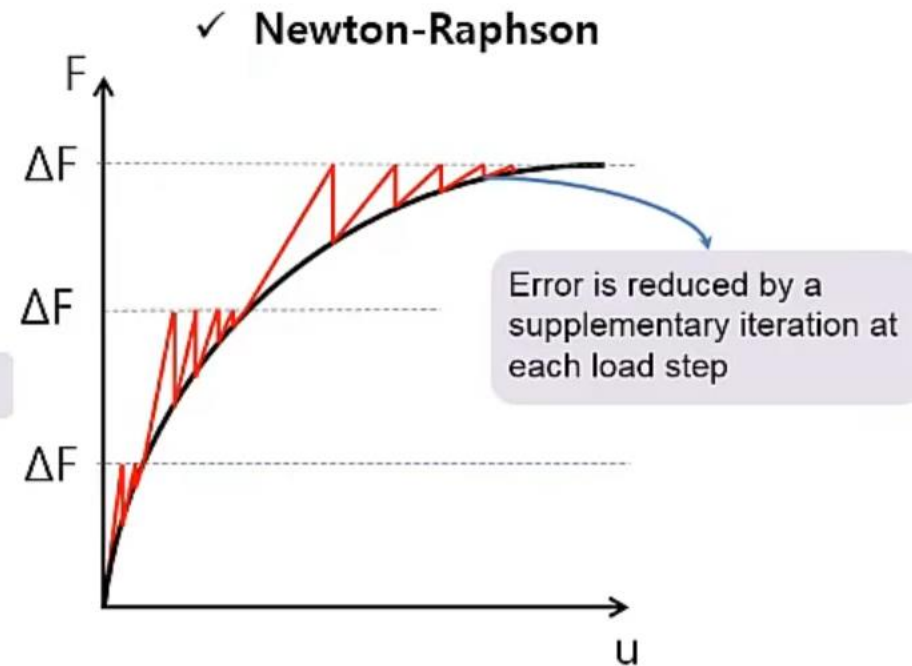


FEA Using Abaqus: Abaqus Solvers

Nonlinear Solution Scheme



Improvement



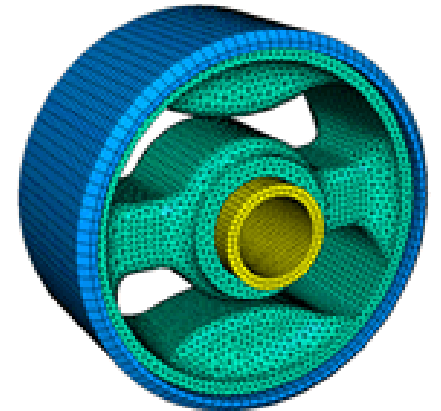
- Stiffness is updated at each load step
- Problem: error accumulated at each load step will create some big error at the end of the analysis.

- Stiffness is updated at each load step
- Error is reduced by adding an internal stiffness iteration for each load step.

FEA Using Abaqus: Abaqus Solvers

Abaqus/Standard (Implicit):

Solution technology for **static**, **quasi-static**, and **low-speed dynamic** events



Abaqus/Explicit:

Solution technology that is particularly well-suited to simulate brief transient **dynamic** events such as automotive crashworthiness.

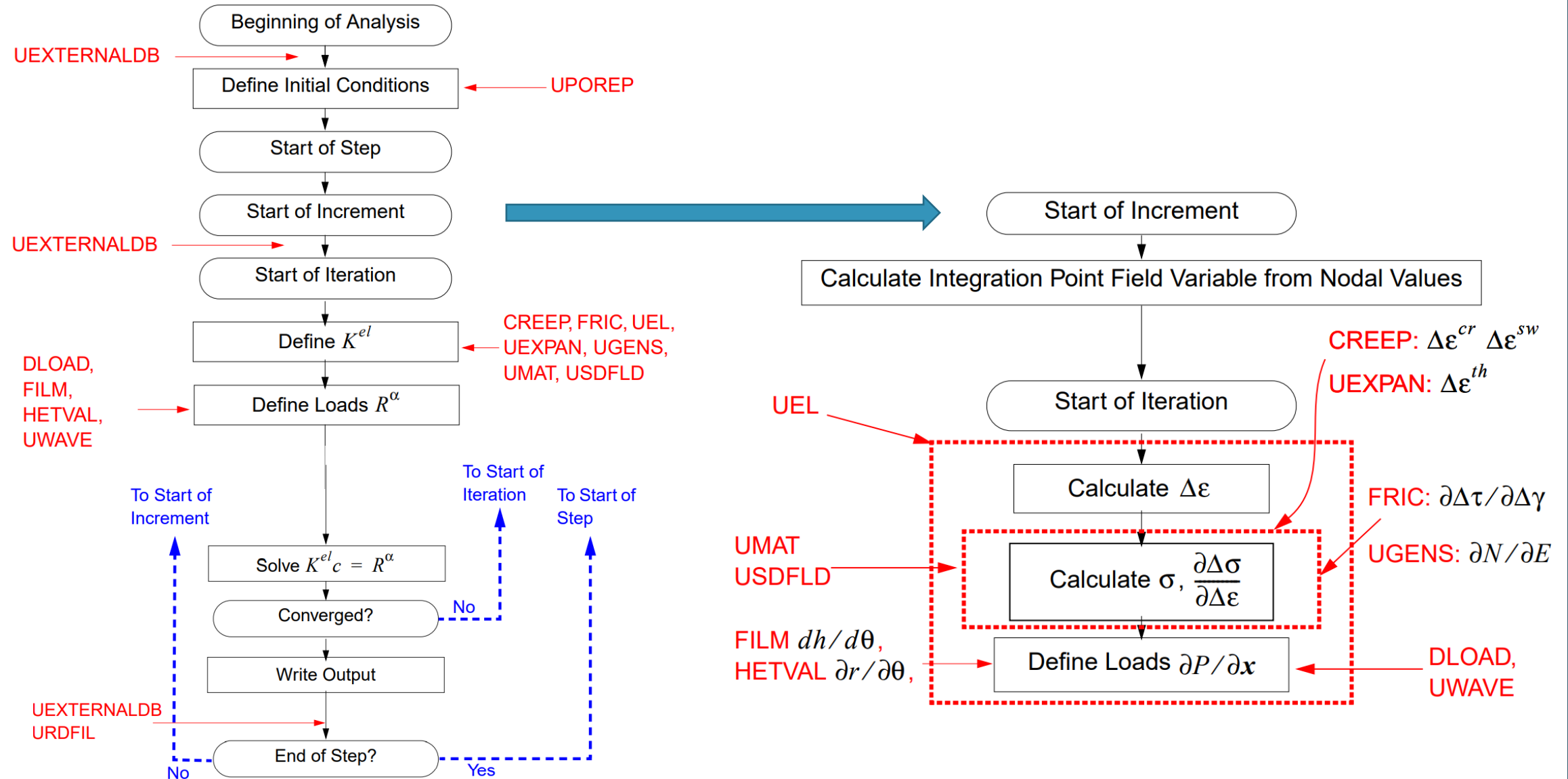
Effectively handle severely nonlinear behavior the simulation of many **quasi-static** events

Comparison

Abaqus/Explicit (The flexibility provided by Explicit integration): well-suited to **high-speed dynamic**, **nonlinear**, **transient** analyses

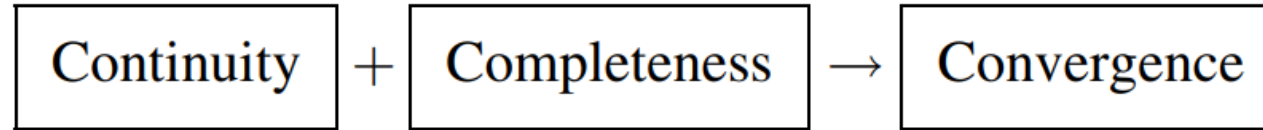
Abaqus/Standard (Implicit integration): well-suited to **static**, **low-speed dynamic**, or **steady-state** transport analyses.

FEA Using Abaqus: User Subroutine



FEA Using Abaqus: User Subroutine

FEA Using Abaqus: Solution Convergence



› Convergence problems (e.g. excessive element distortion, ...)  **Abort**



Adaptivity Techniques

› **Mesh Independency:** field variable (e.g. displacement) must be mesh independent

FEA Using Abaqus: Solution Convergence

Adaptivity Techniques

The finite element discretization that results from suboptimal meshing of models can limit your ability to obtain adequate analysis results at a reasonable CPU cost. This section provides an overview of the adaptivity techniques available in Abaqus that help you optimize a mesh and, therefore, obtain quality solutions while controlling the cost of your analysis. The term “adaptivity” reflects the adaptive, or solution-dependent, processes that Abaqus uses to adapt your mesh to meet your analysis goals.

	Accuracy	Distortion control	Single mesh	Multiple meshes	Adaptivity occurs
ALE adaptive meshing		✓	✓		Throughout a step
Adaptive remeshing	✓			✓	Separately from analysis steps
Mesh-to-mesh solution mapping		✓		✓	Between analysis steps

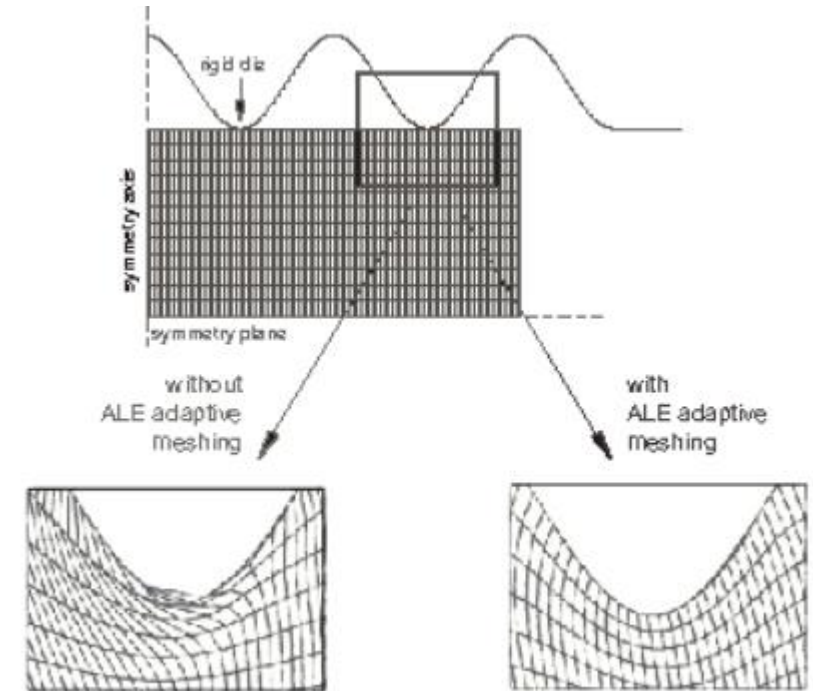
FEA Using Abaqus: Solution Convergence

Adaptivity Techniques

- ✓ **ALE Adaptive Meshing:** Arbitrary Lagrangian-Eulerian (ALE) adaptive meshing provides control of mesh distortion. ALE adaptive meshing uses a single mesh definition that is gradually smoothed within analysis steps. ALE adaptive meshing is available for **limited applications in Abaqus/Standard** and is more generally applicable in Abaqus/Explicit.

- ✓ Adaptive Remeshing

- ✓ Mesh-to-mesh Solution Mapping

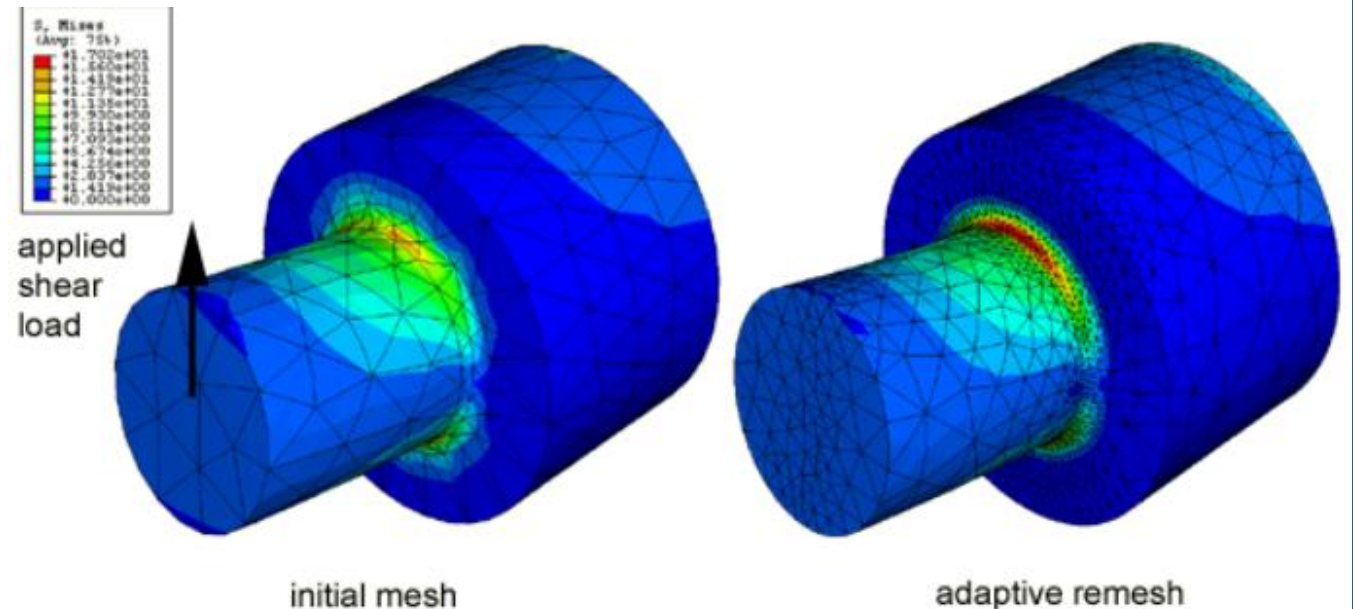


FEA Using Abaqus: Solution Convergence

Adaptivity Techniques

- ✓ ALE Adaptive Meshing
- ✓ **Adaptive Remeshing:** is typically used for **accuracy** control, although it can also be used for distortion control in some situations. The adaptive remeshing process involves the iterative generation of multiple dissimilar meshes to determine a single, **optimized mesh that is used throughout an analysis**. Adaptive remeshing is available only for Abaqus/Standard analyses submitted from Abaqus/CAE. The goal of adaptive remeshing is to obtain a solution that satisfies mesh discretization error indicator targets that you set, while minimizing the number of elements and, hence, the cost of your analysis.

- ✓ Mesh-to-mesh Solution Mapping

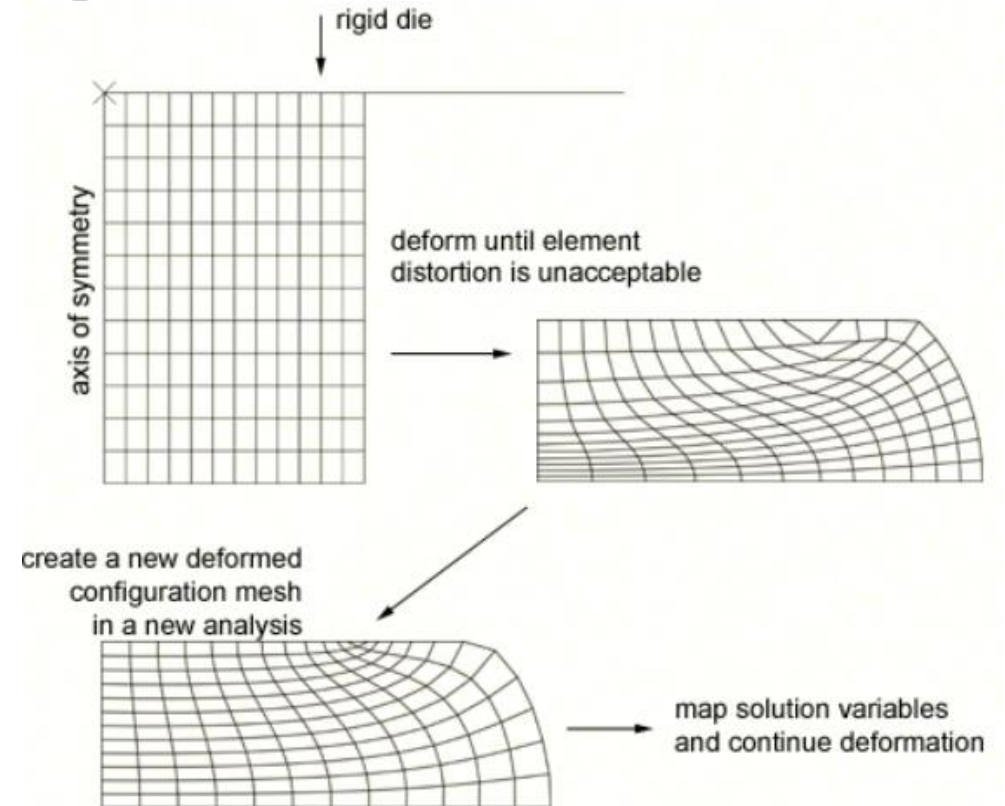


FEA Using Abaqus: Solution Convergence

Adaptivity Techniques

✓ ALE Adaptive Meshing

✓ Adaptive Remeshing



✓ **Mesh-to-mesh Solution Mapping:** is available only in Abaqus/Standard. You can use this technique to control element distortion in cases where **large deformation** occurs by replacing the mesh and continuing the analysis. The figure illustrates a case where solution mapping is used in conjunction with a new mesh to overcome difficulties associated with **element distortion**.

FEA Using Abaqus: Solution Convergence

Mesh-to-mesh Solution Mapping

Mapping a solution from one mesh to another is a step in a remeshing analysis technique, where a mesh that has deformed significantly from its original configuration is replaced by a mesh of better quality and the analysis **continues**. The solution mapping technique:

- ☐ Is used when elements become so **severely distorted** during an analysis that they no longer provide a good discretization of the problem
- ☐ Maps the solution from an old, deformed mesh to a new mesh so that the analysis can continue
- ☐ Can be used only with **continuum elements**

FEA Using Abaqus: Generated Files

When a session has been created and begun defining your model, Abaqus/CAE generates the following file:

1. The **replay file** (**abaqus.rpy**): The replay file contains Abaqus/CAE commands that record almost every modeling operation you perform during a session.
2. The **model database file** (**abaqus.cae**): contains models and analysis jobs and created when the model has been saved
3. The **journal file** (**model_database_ name.jnl**): contains the Abaqus/CAE commands that will replicate the model database that was saved to disk.
4. The **recover file** (**model_database_ name.rec**): contains the Abaqus/CAE commands that will replicate the version of the model database in memory. The model database recovery file contains only the commands that changed the model database since you last saved it. For more information, see Recreating an unsaved model database.

When a job is submitted for analysis, Abaqus/Standard and Abaqus/Explicit create a set of files. The following list describes some of the files that Abaqus/Standard and Abaqus/Explicit create and their relationship to Abaqus/CAE:

1. **Input files (job_name.inp):** Abaqus/CAE generates an input file that is read by Abaqus/Standard or Abaqus/Explicit when you submit a job for analysis.
2. **Output database files (job_name.odb):** Output database files contain the results from your analysis. You use the Step module's output request managers to choose which variables are written to the output database during the analysis and at what rate. An output database is associated with the job you submit from the Job module; for example, if you named your job FrictionLoad, the analysis creates an output database called *FrictionLoad.odb*. When you open an output database, Abaqus/CAE loads the Visualization module and allows you to view a graphical representation of the contents. **You can also import a part from an output database as a mesh.** You can save X-Y data objects to an output database file if you open the file with write permission; otherwise, you cannot modify the contents of the output database once it has been created.
3. **The output database lock file (job_name.lck):** The lock file (job_name.lck) is written whenever an output database file is opened with write access, including when an analysis is running and writing output to an output database file. The lock file prevents you from having simultaneous write permission to the output database from multiple sources. It is deleted automatically when the output database file is closed or when the analysis that creates it ends.
4. **The restart file (job_name.res):** The restart file is used to continue an analysis that stopped before it was complete. You use the Step module to specify which analysis steps should write restart information and how often. If you are using Abaqus/Explicit, the restart information you supply in the Step module controls the data written to the state file (job_name.abq).
5. **The data file (job_name.dat):** The data file contains printed output from the analysis input file processor, as well as printed output of selected results written during the analysis. Abaqus/CAE automatically requests that the default printed output for the current analysis procedure be generated at the end of each step; you cannot use Abaqus/CAE to exert any additional control over the contents of the data file.
6. **The message file (job_name.msg):** The message file contains diagnostic or informative messages about the progress of the solution. You can control the diagnostic information that is output to the message file using the Step module.
7. **The status file (job_name.sta):** The status file (job_name.sta) contains information about the progress of the analysis. In addition, you use the Step module to request that the value of a single degree of freedom at a single node be output to the status file. For more information, see Degree of freedom monitor requests.
8. **The results file (job_name.fil):** The results file contains selected results from the analysis in a format that can be read by other applications, such as postprocessing programs. A submodel analysis can read the global model results from either an output database or a results file. By default, an analysis from Abaqus/CAE does not create a results file. For more information, see Submodeling, and Submodeling."

Note:

The errors and warnings that Abaqus/Standard and Abaqus/Explicit write to the **data**, **message**, and **status** files while analyzing a job can be monitored by the Job module.

When you open an output database file in the Visualization module and create new field output variables, Abaqus/CAE generates the following file:

- The scratch output database file (job_name.ods): contains a “session step” in which field output variables that you create (by operating on either fields or frames) are saved. This file is deleted automatically when the original output database file (from which the field output originates) is closed or when the Abaqus/CAE session ends.
In most cases the files generated by Abaqus/CAE are written to the work directory. The work directory is the directory from which you started the Abaqus/CAE session unless you changed the directory by selecting File “Set Work Directory” from the main menu bar.

FEA Using Abaqus: Abaqus Automation Possibility

Programing in Abaqus

Abaqus Keywords: has been implemented for **pre-processing** (i.e., creation or modification of Simulation)



Abaqus has considered particular syntax the same as .inp



Abaqus
Documentation

Abaqus Scripts: has been implemented for pre-processing as well as post-processing (specially for repetitive task)



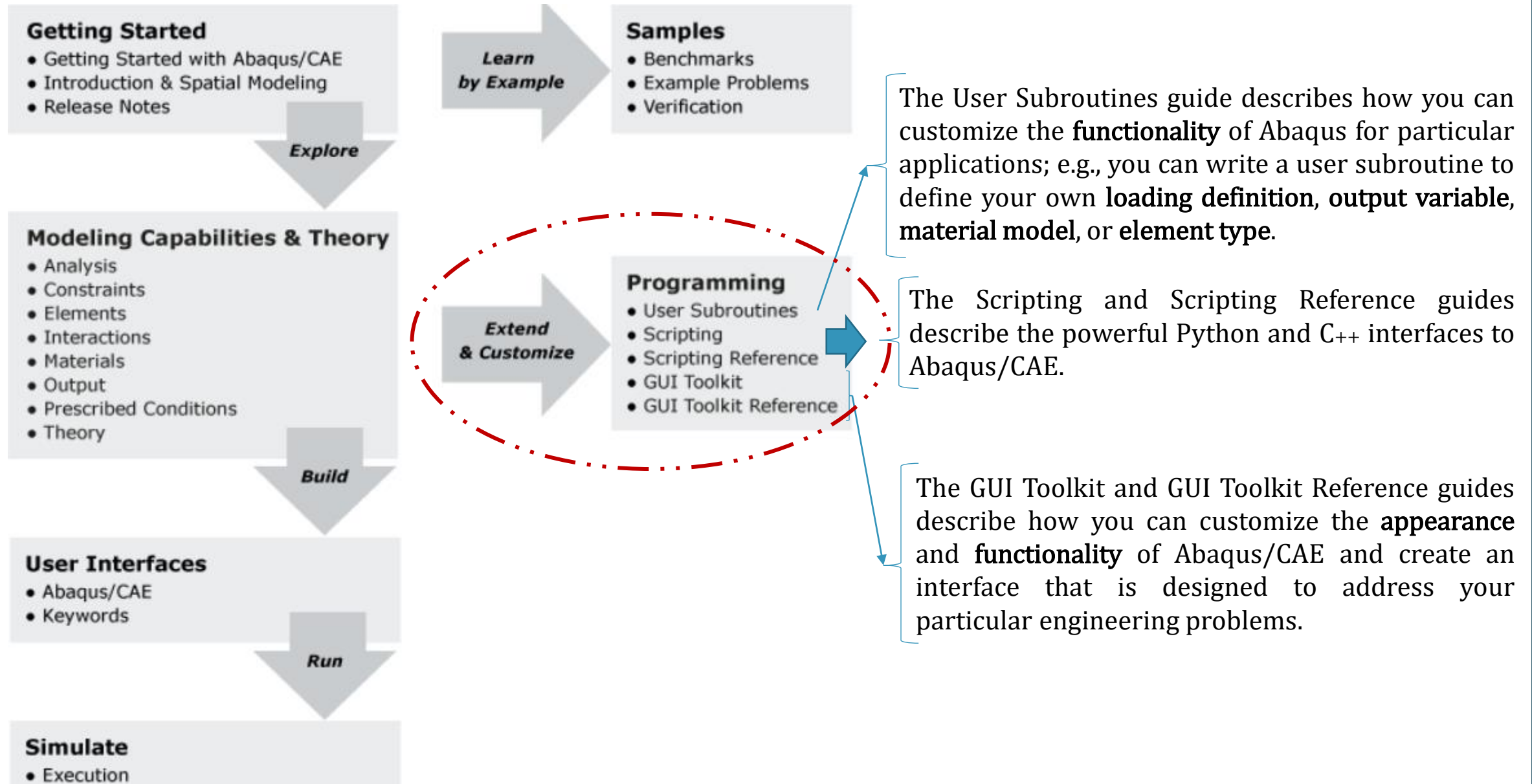
Python Programming Language

Subroutines: has been implemented for enhancing capability of Abaqus Solvers

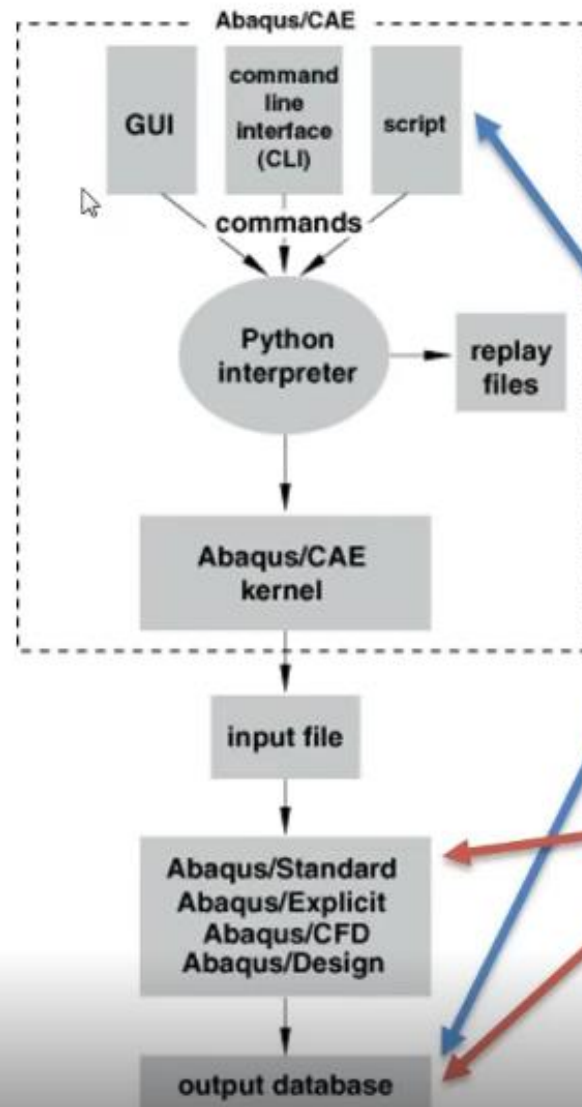


Fortran and **C++** Programming Language

FEA Using Abaqus: Abaqus Automation Possibility



FEA Using Abaqus: Abaqus Automation Possibility



1. Python script:
 1. Pre-process
 2. Execute
 3. Post-process



2. Subroutines: Fortran/C
 1. Solver expansion
 2. Post-process

FEA Using Abaqus: Abaqus Automation Possibility

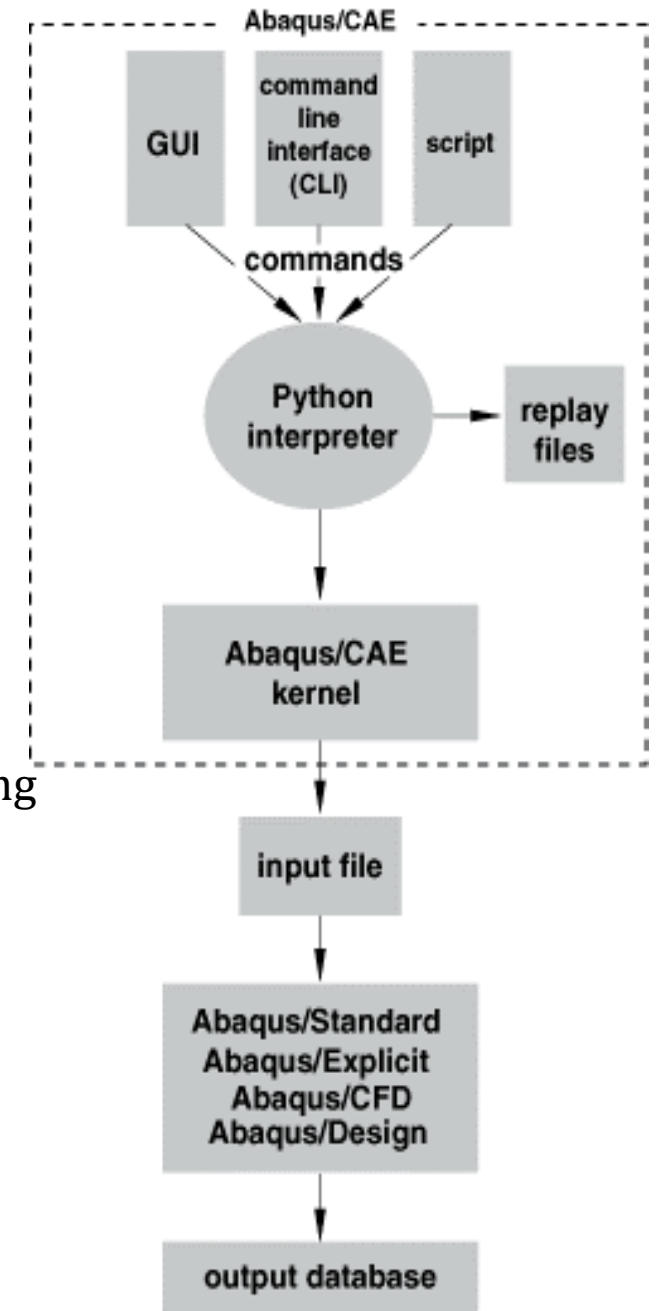
How does the Abaqus Scripting Interface interact with Abaqus/CAE ? →

Switch to script

If there are more than a few commands to execute or if the same commands are execute **repeatedly**, it may be more convenient to store the set of statements in a file called a script. A script contains a sequence of **Python statements** stored in plain ASCII format.

You can run a script when you start an Abaqus/CAE session by typing the following command: `abacus cae script=myscript.py` where `myscript.py` is the name of the file containing the script. The **Abaqus.jnl** contains commands of modeling that will be sent into Python interpreter

The **Abaqus.rpy** contains commands of Visualization that will be sent into Python interpreter



FEA Using Abaqus: Abaqus Keyword

The following rules apply when entering a keyword line:

The first non-blank character of each keyword line must be a star (*).

The keyword must be followed by a comma (,) if any parameters are given.

Parameters must be separated by commas.

Blanks on a keyword line are ignored.

A line can include no more than 256 characters, including blanks. There are additional limitations when encrypting an input file (see Encrypting and decrypting Abaqus input data).

Keywords and parameters are not case sensitive.

Parameter values usually are not case sensitive. The only exceptions to this rule are those imposed externally to Abaqus, such as file names on case-sensitive operating systems.

Keywords, parameters, and, in most cases, parameter values need not be spelled out completely, but there must be enough characters given to distinguish them from other keywords, parameters, and parameter values that begin in the same way. Abaqus first searches each associated text string for an exact match. If an exact match is not found, Abaqus then searches based upon the minimum number of unique characters in each keyword, parameter, or parameter value, as the case may be. Embedded blanks can be omitted from any item in a keyword line. If a parameter value is used to provide a number or a file name, the complete value should be provided.

If a parameter has a value, the equal sign (=) is used. The value can be an integer, a floating point number, or a character string, depending on the context. For example,

ELASTIC, TYPE=ISOTROPIC, DEPENDENCIES=1

FEA Using Abaqus: Abaqus Keyword

When the parameter value is a character string that represents the name of an item, you should not use case as a method of distinguishing values unless the values are enclosed within quotation marks. For example, Abaqus does not distinguish between the following definitions:

```
MATERIAL, NAME=STEEL
```

```
MATERIAL, NAME=Steel
```

The same parameter should not appear more than once on a single keyword line. If a parameter has multiple settings on a single keyword line, Abaqus ignores all but one of the settings.

Continuation of a keyword line is sometimes necessary; for example, because of a large number of parameters. If the last character on a keyword line is a comma, the next line is interpreted as a continuation of the line. For example, the ELASTIC keyword line above could also be given as

```
ELASTIC, TYPE=ISOTROPIC,
```

```
DEPENDENCIES=1
```

Certain keywords must be used in conjunction with other keywords; for example, the ELASTIC and DENSITY keywords must be used in conjunction with the MATERIAL keyword. These related keywords must be grouped in a block in the input file; unrelated keywords cannot be specified within this block.

FEA Using Abaqus: Abaqus Keyword

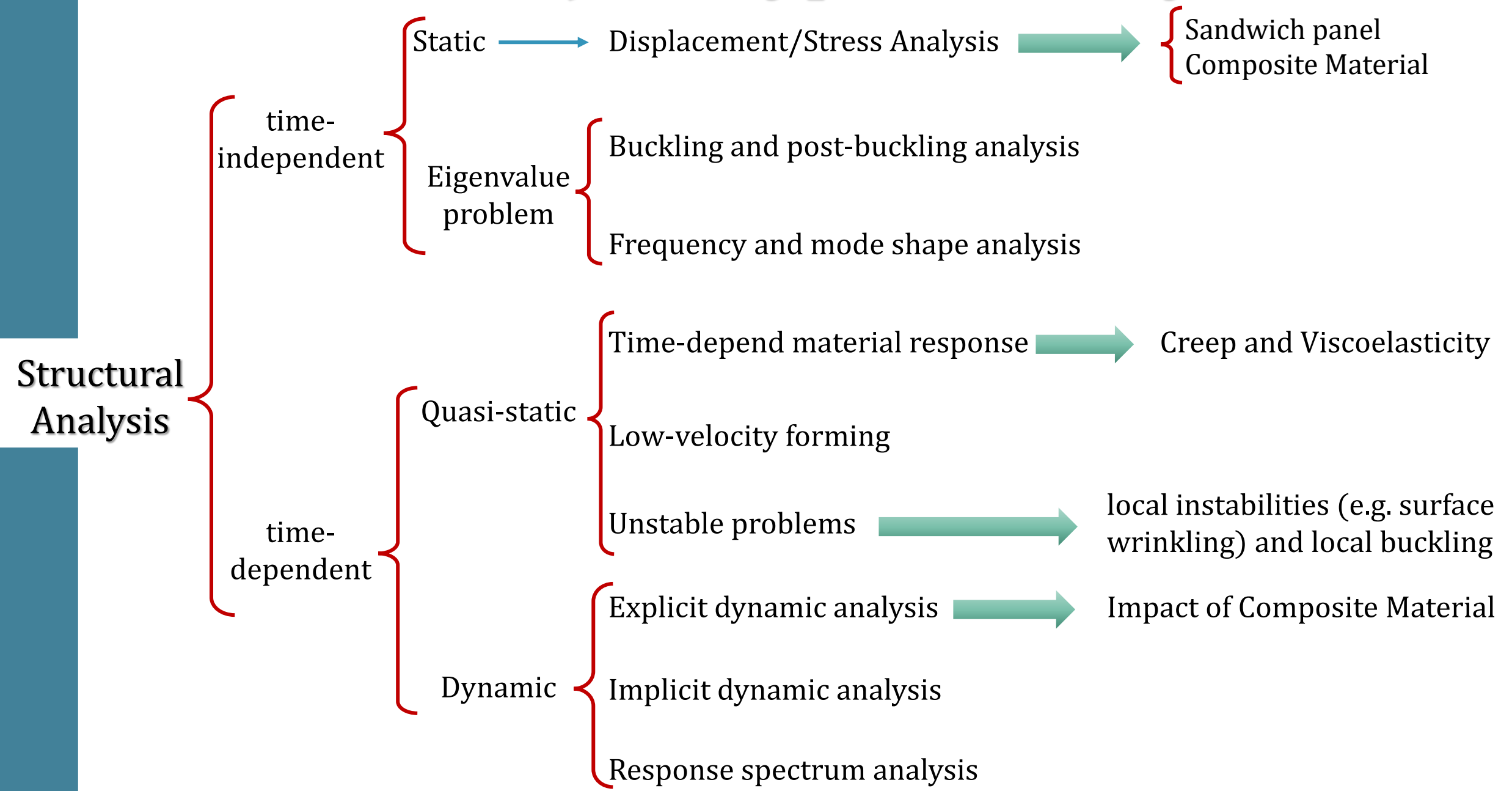
Some options allow the INPUT or FILE parameter to be set equal to the name of an alternate file. Such file names can include a full path name or a relative path name. Relative path names must be with respect to the directory from which the job was submitted. If no path is specified, the file is assumed to be in the directory from which the job was submitted. A substructure library must be in the same directory from which the job was submitted; a full path name cannot be used to specify a substructure library name.

For files referenced by the INPUT parameter, the file name must include any extension (for example, elem.inp). For files referenced by the FILE parameter, the name must be given without an extension in most cases since Abaqus assumes that the file to be read has the correct extension for the file type that is relevant to the option: .res for restart files (Restarting an analysis) and .fil for results files (About Output). However, special rules may apply when a results file (.fil) or an output database file (.odb) is relevant for the option (see Initial Conditions and Sequentially coupled thermal-stress analysis for details).

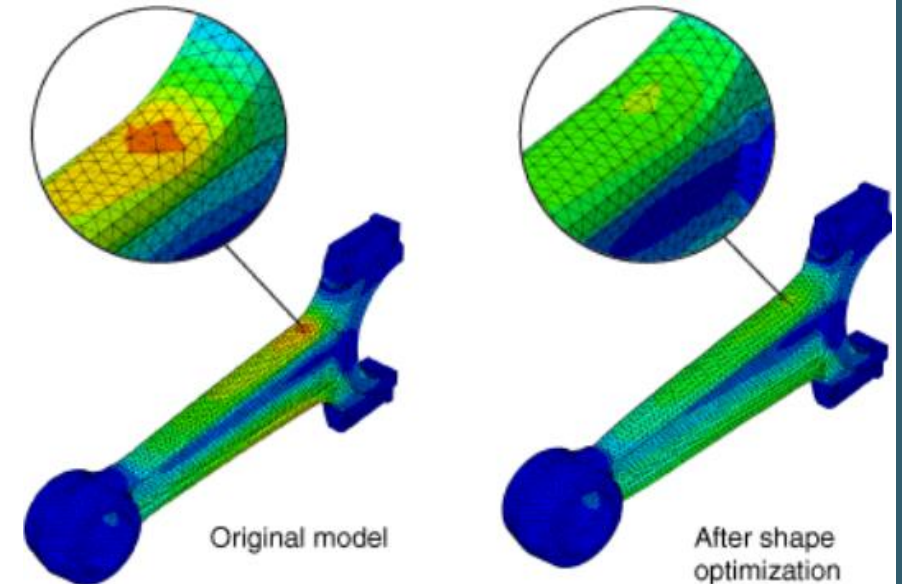
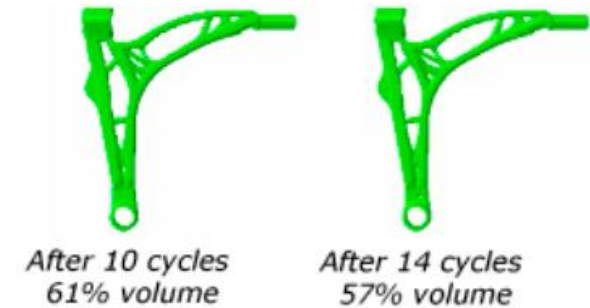
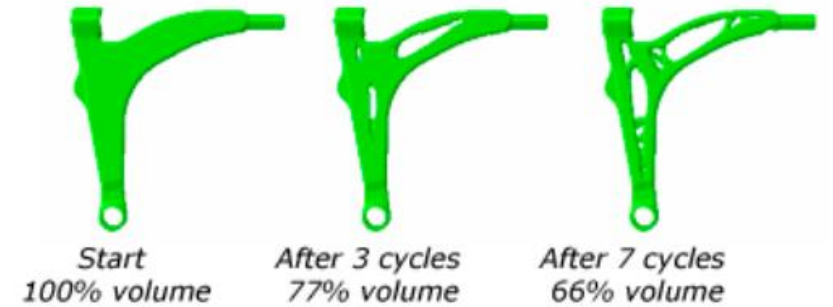
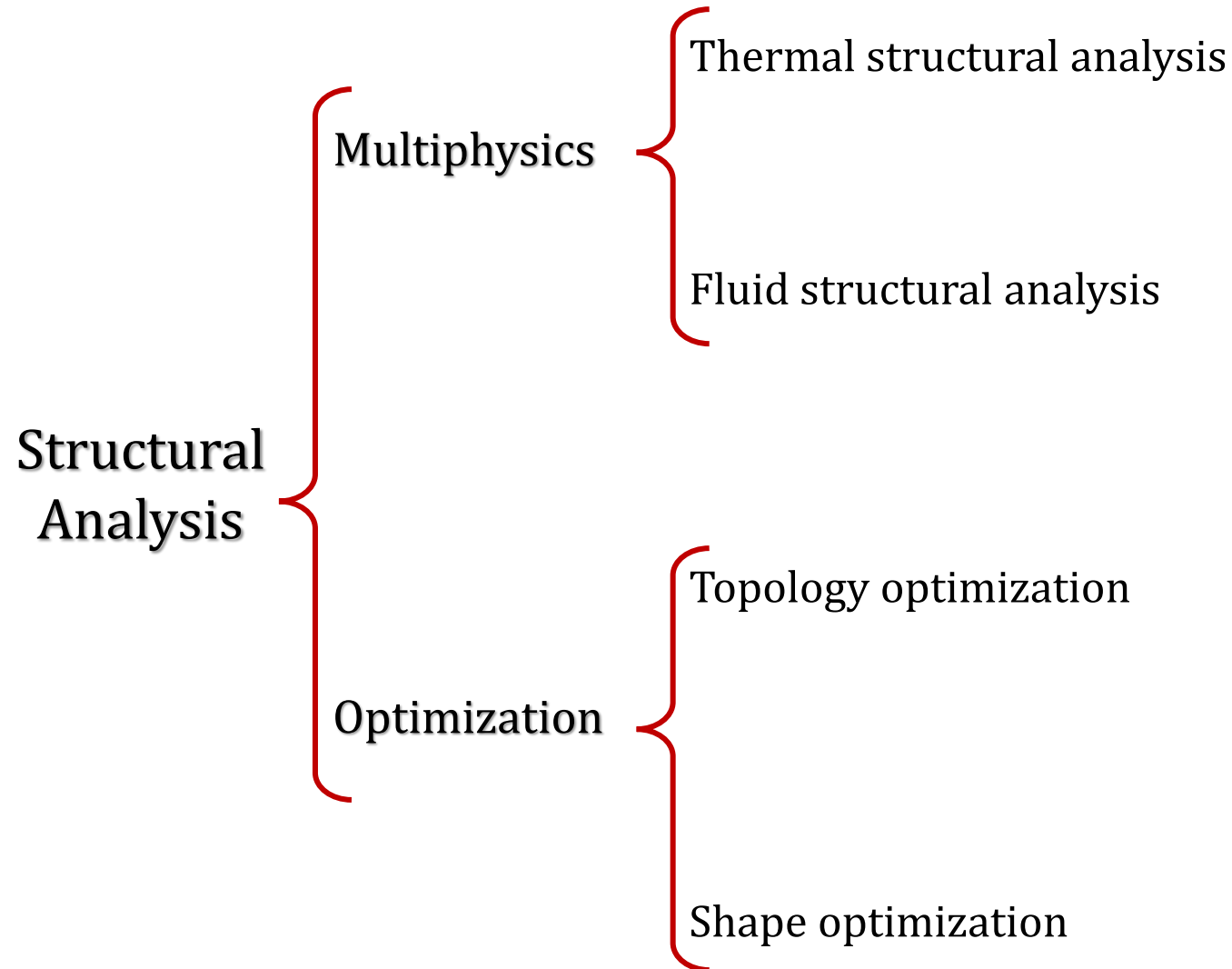
The import and part instance options allow the LIBRARY parameter to be set to a value that specifies the previous analysis from which to import the element sets or instance (see IMPORT and INSTANCE).

The file or library name must have the correct case on computers with case-sensitive operating systems. Regardless of whether the user specifies only a file name, a relative path name, or a full path name, the complete name including the path can have a maximum of 256 characters. All spaces within a file name, a relative path name, or a full path name are ignored unless the name is enclosed in quotation marks, in which case all spaces within the name are maintained.

Final Project: Types of Analysis



Final Project: Types of Analysis



Final Project: Suggested Subject

Static, Quasi-static, and Dynamic Analysis



Co-Simulation

Forming Process Analysis

- Forging
- Extrusion
- Rolling
- Sheet metal forming
- Rotary swaging
- Thread rolling
- Explosive forming
- Electromagnetic forming

Composite Materials
&
Sandwich Panel

- Impact
 - Low-velocity
 - High-velocity
- Delamination
- Forming
- Optimization

Buckling and Post-buckling Analysis

Vibration Analysis

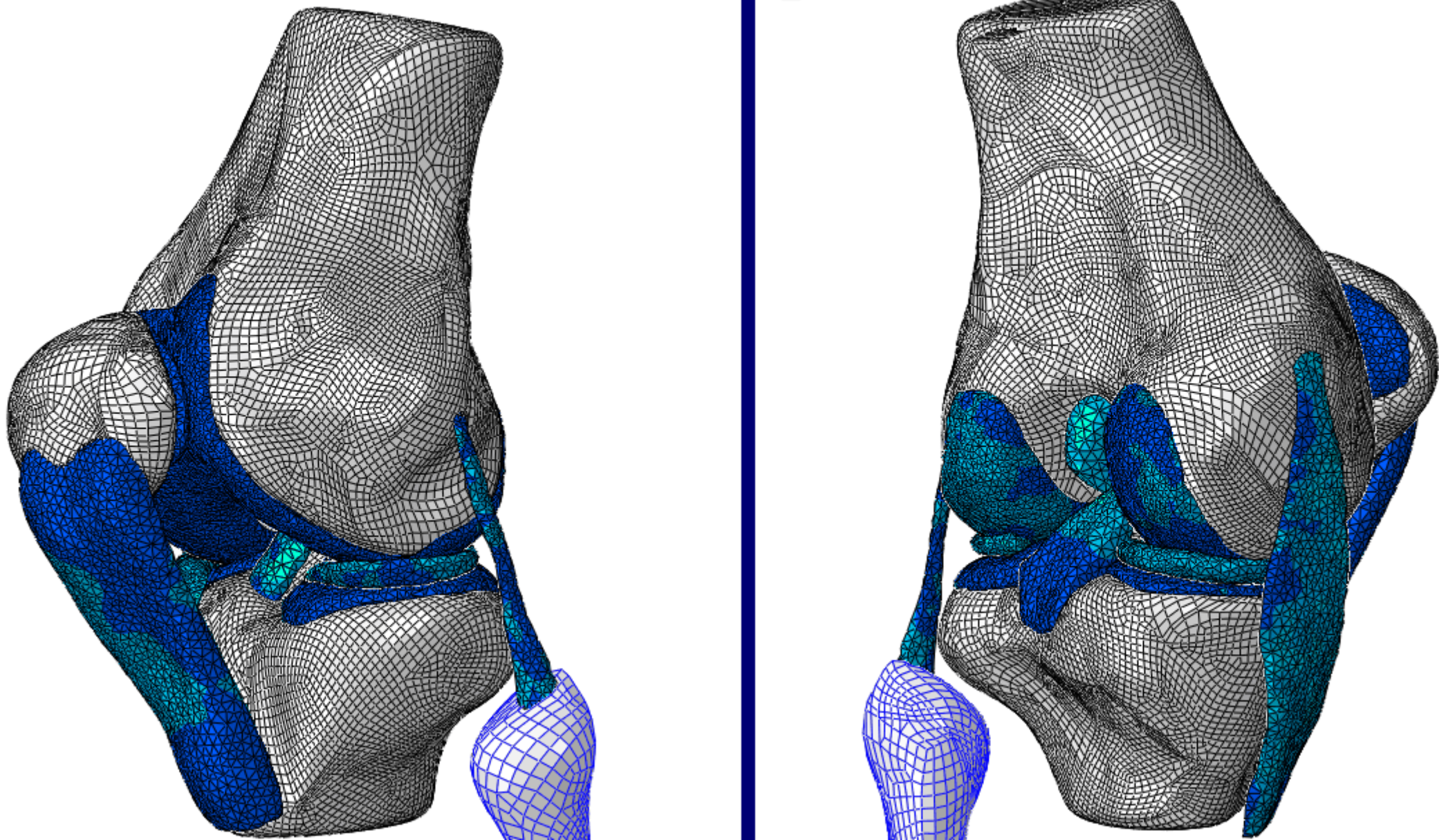
Thermal Structural Analysis

Fluid Structural Analysis

Impact Analysis

Topology/Shape Optimization Analysis

Example

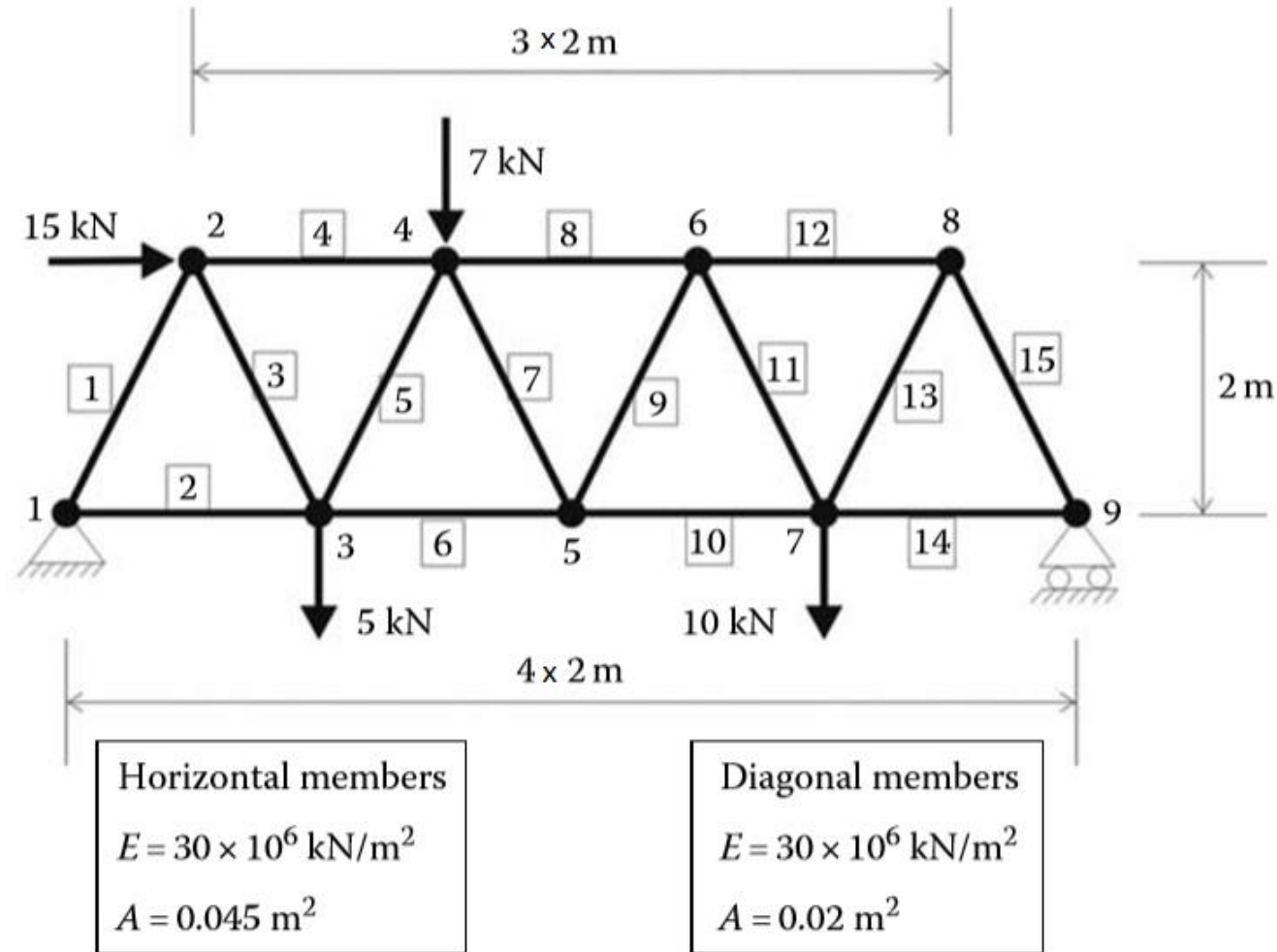


Exercise

1- It is desired to determine the definition and application of the below items.

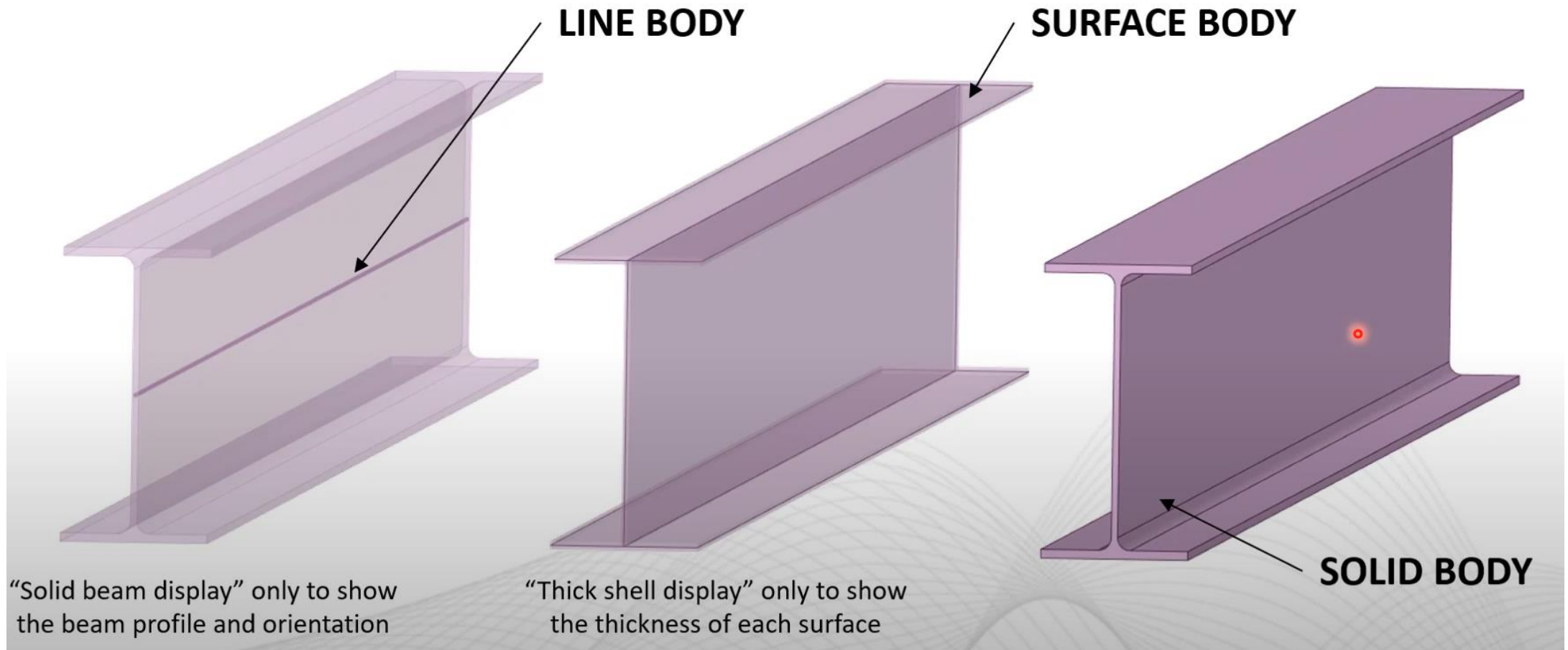
- Shear locking and membrane locking
- hourglass modes
- Comparative Full Integration and Reduced Integration
(i.e., Plus points and negative points of both of them)

Truss Problem

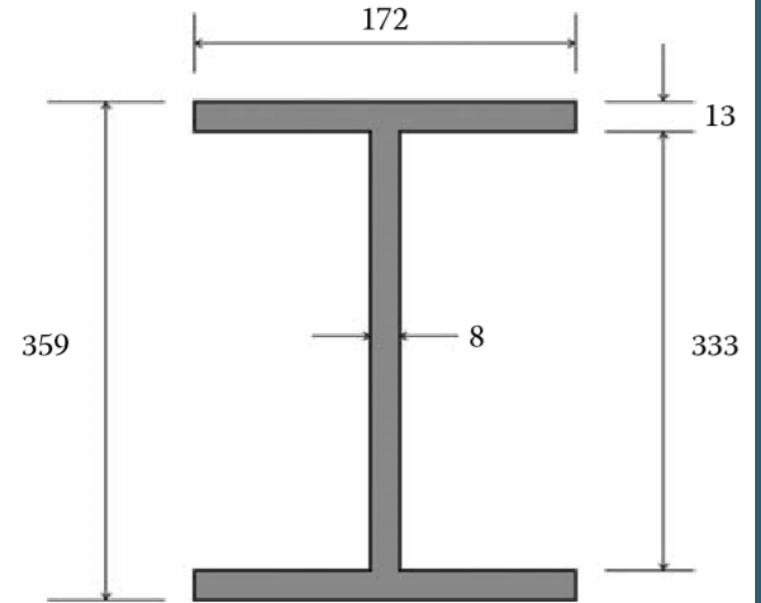
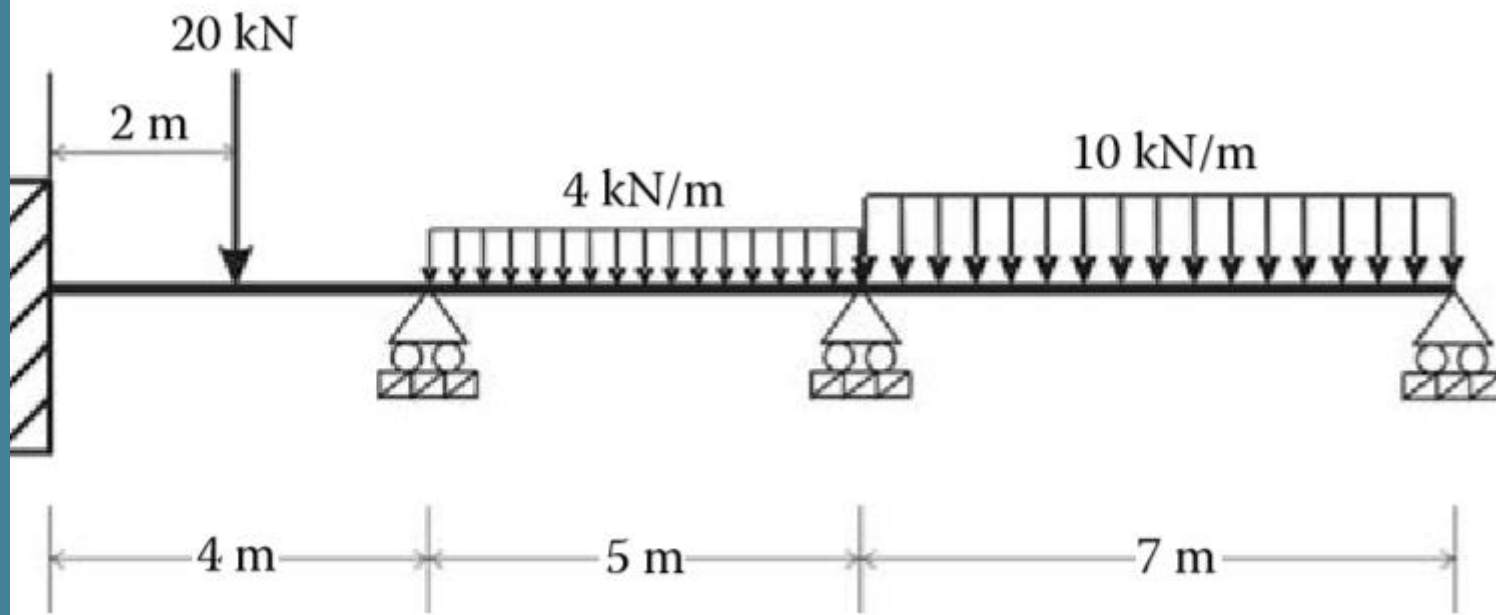


Beam Problem

Different types of modeling and associated assumptions



Beam Problem



$$E = 200 \text{ GPa} \quad \nu = 0.3$$

Exercise

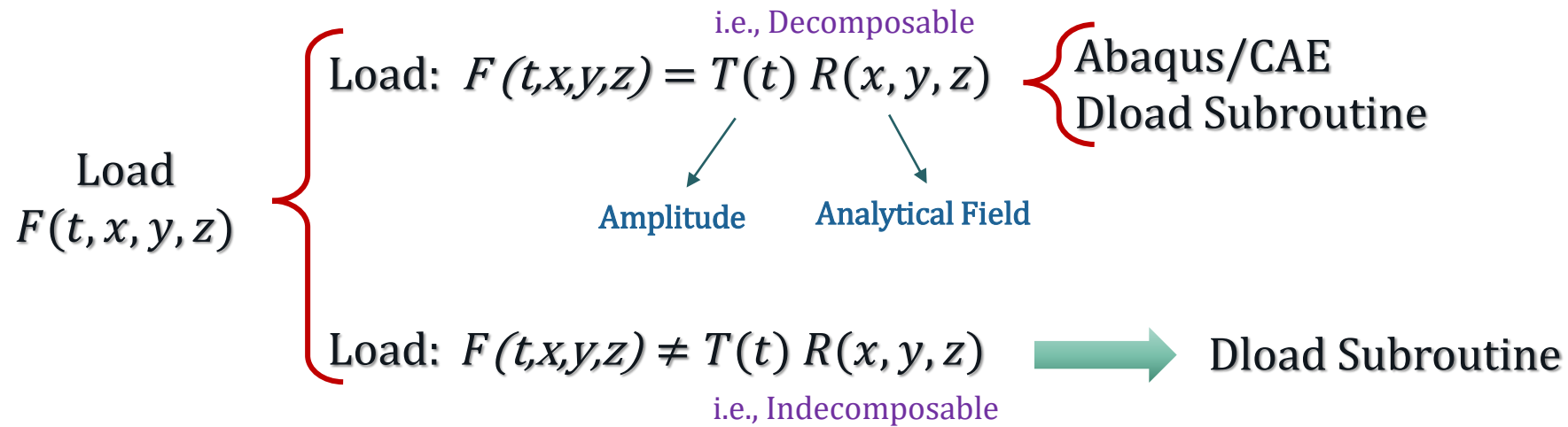
1- Determine the strain and stress components in Problems 1 & 2.

The desired files must be received from Abaqus result with .rpt format.

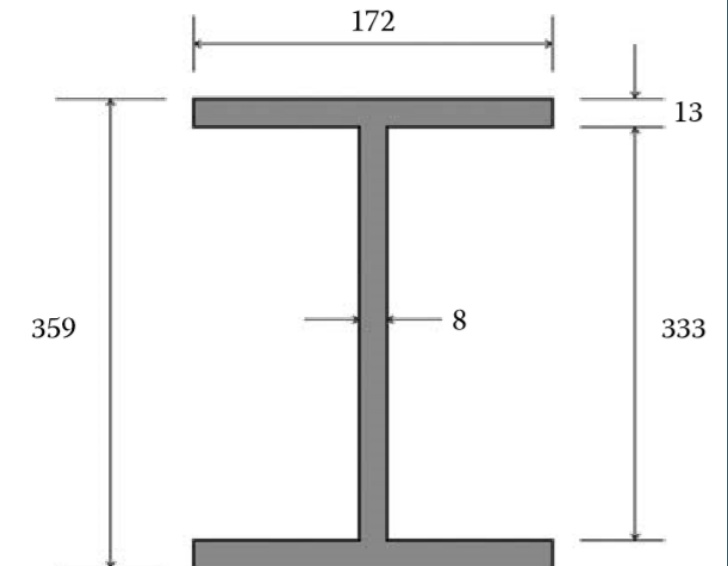
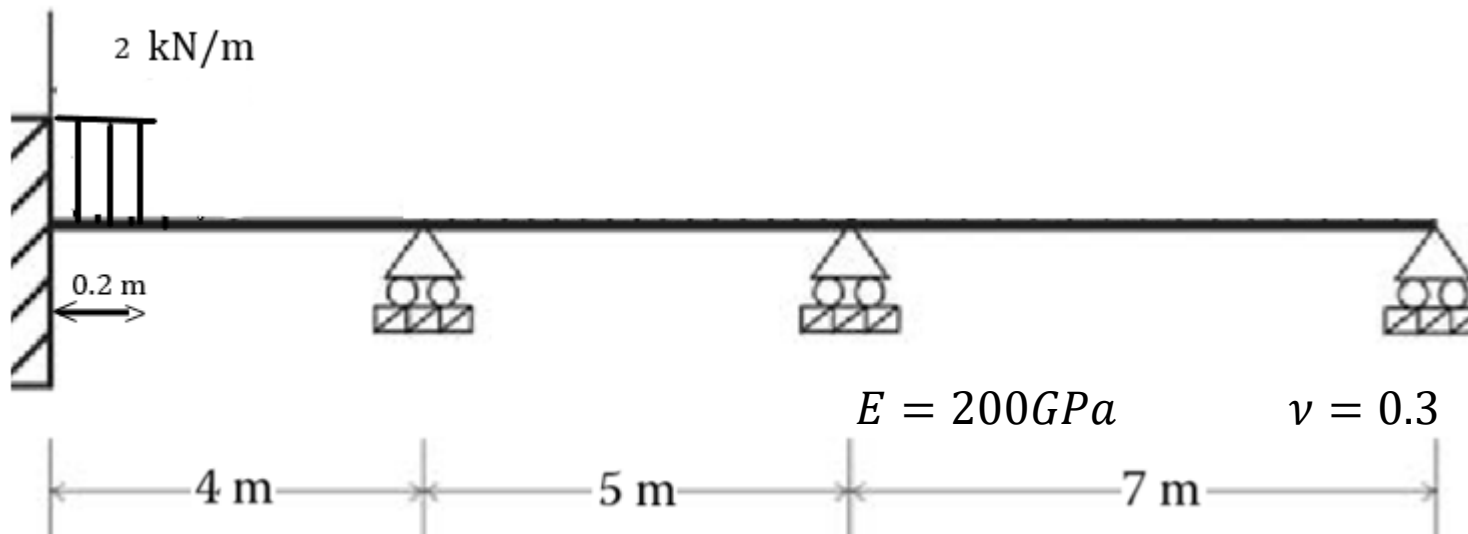
2- Determine the definition and applications of different strain and stress measurements in Abaqus.

(Conspicuously, searching in Abaqus documentation along with on the Internet would be helpful.)

Moving Load: Dload Subroutine



The load is monitored by writing output to the printed output (.dat) file



DLOAD Subroutine

```
      SUBROUTINE DLOAD(F,KSTEP,KINC,TIME,NOEL,NPT,LAYER,KSPT,  
1 COORDS,JLTYP,SNAME)  
C  
      INCLUDE 'ABA_PARAM.INC'  
C  
      DIMENSION TIME(2), COORDS (3)  
      CHARACTER*80 SNAME  
  
      user coding to define F  
  
      RETURN  
      END
```

Variables to be defined (i.e., F): FL^{-2} for surface loads and FL^{-3} for body forces.

KSTEP: Step number.

KINC: Increment number.

TIME(1): Current value of step time or current value of the load proportionality factor

TIME(2): Current value of total time.

NOEL: Element number.

NPT: Load integration point number within the element or on the element's surface, depending on the load type.

LAYER: Layer number (for body forces in layered solids).

KSPT: Section point number within the current layer.

COORDS: An array containing the coordinates of the load integration point. These are the current coordinates if geometric nonlinearity is accounted for during the step; otherwise, the array contains the original coordinates of the point.

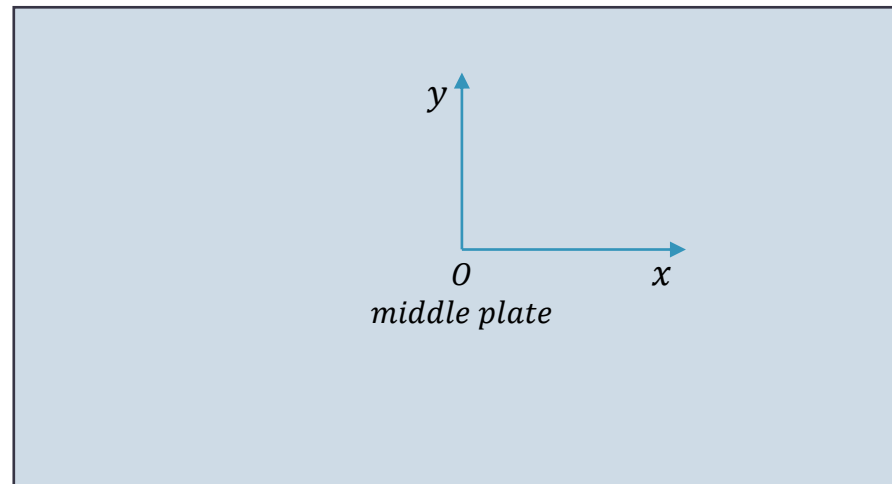
JLTYP :Surface name for a surface-based load definition (JLTYP=0). For a body force or an element-based surface load the surface name is passed in as blank.

Exercise: DLOAD

Simulation time: 10 (s)

Hint: check the JLTYP in DLOAD subroutine out

Loads { **Surface Force (Pressure):** exert on entire plate $P(x, y, t) = 2\sin\left(\frac{\pi x}{300}\right)\cos\left(\frac{\pi y}{200}\right) + 5\cos(\pi t)$
Body force: exert on whole plate $F_b(x, y, t) = \cos\left(\frac{\pi x}{300}\right)\sin\left(\frac{\pi y}{200}\right) + 2\cos(\pi t)$



All edge has been pinned

Plate's dimensions: 300x200 (mm), thickness: 2 (mm)

Material properties: $E=200 \text{ GPa}$ $\nu = 0.3$

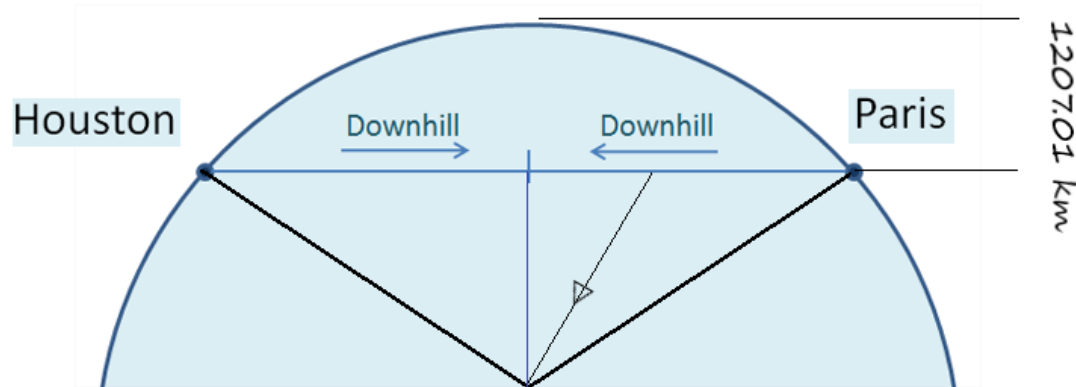
Exercise: Gravity Train

A **gravity train** is a theoretical means of transportation for purposes of commuting between two points on the surface of a sphere, by following a straight tunnel connecting the two points through the interior of the sphere.

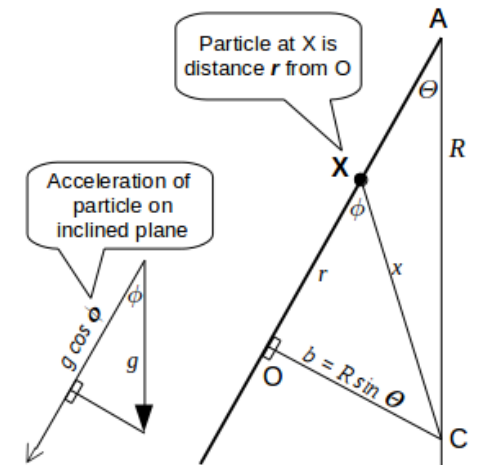
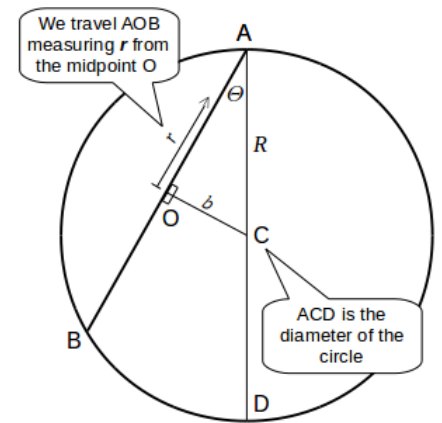
Straight path between two arbitrary points

$$F = G \frac{Mm}{x^2} \stackrel{\text{On the surface}}{=} G \frac{Mm}{R_e^2}$$

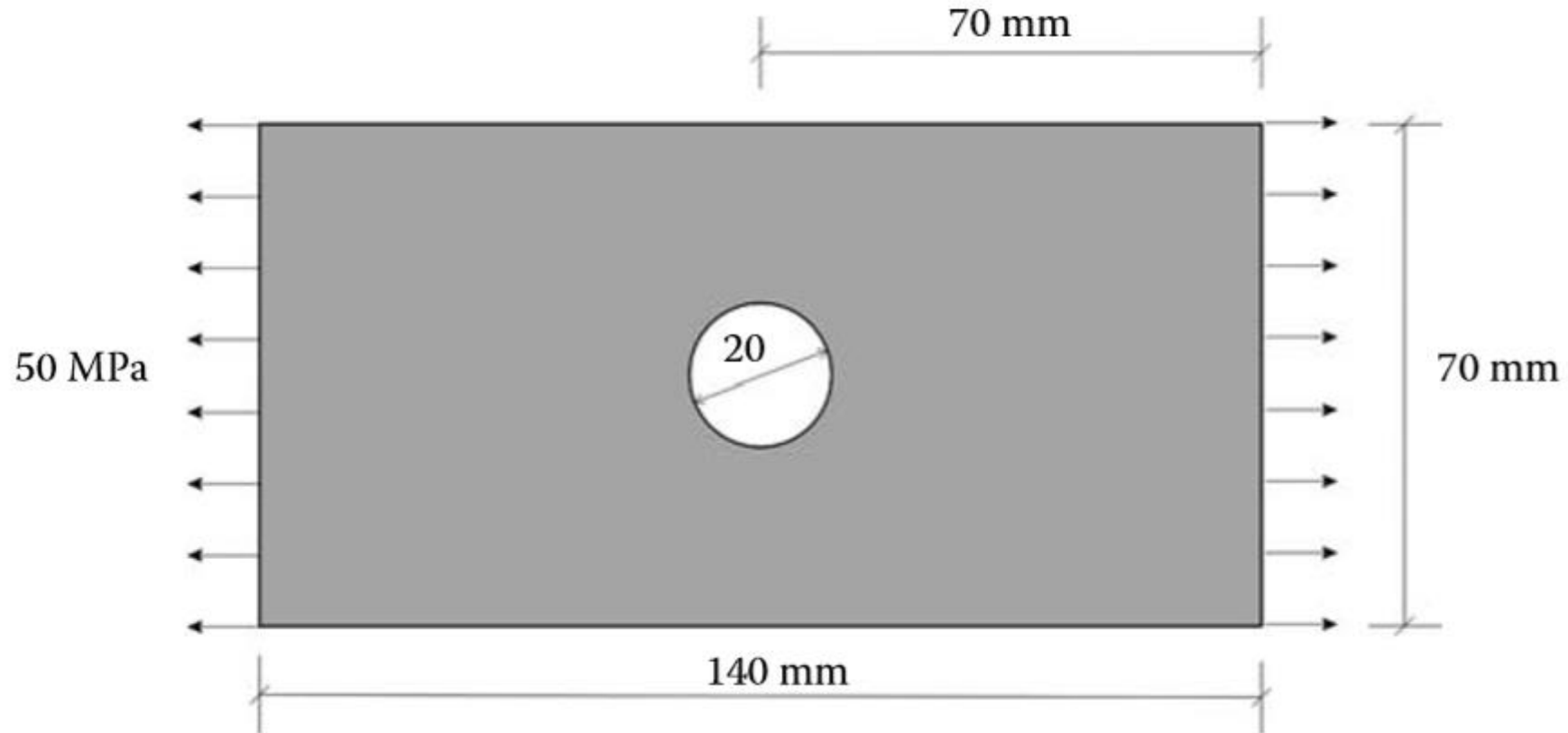
$$F = G \frac{Mm}{x^2} = G\rho \frac{4}{3}\pi x m \Rightarrow F = mg \frac{x}{R_e} = mg \frac{\sqrt{r^2 + b^2}}{R_e}$$



$$R_e = 6371 \text{ km}$$



Plane Stress Problem



$$\varepsilon = \ln(1 + \varepsilon_{nom})$$

$$E = 70 \text{ GPa}$$

$$\nu = 0.33$$

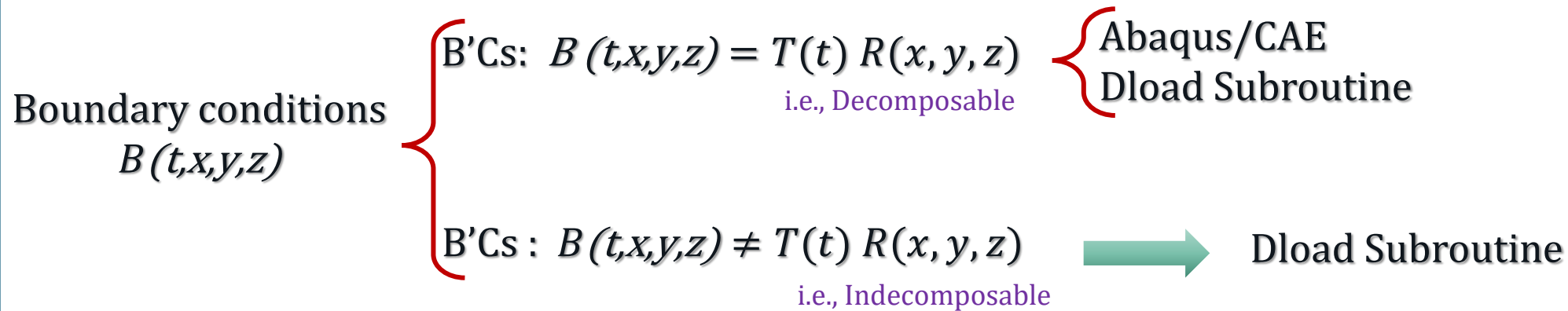
$$\text{Thickness} = 2 \text{ mm}$$

$$\sigma = \sigma_{nom} (1 + \varepsilon_{nom})$$

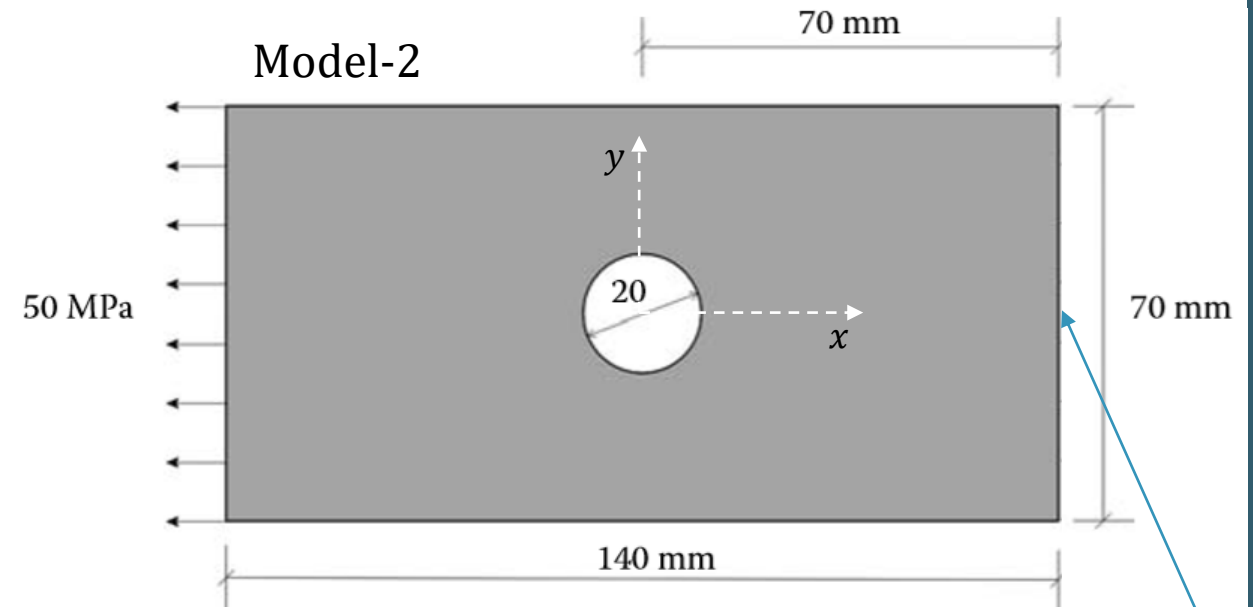
Exercise

- 1- it is desired to simulate half of plate (i.e., x-symm. & y-symm. should be considered)
- 2- it is desired to simulate entire plate (i.e., neglect the symmetry of problem)
- 3- The comparison of the these 3 simulation conditions (i.e., x-symm. & y-symm., and without symmetry) should be investigated.

Disp Subroutine Problem



Model-1



$$U_3 = 5 \cos(2\pi t) \sin\left(\frac{\pi x}{70}\right) \sin\left(\frac{\pi y}{35}\right) \quad E = 70 \text{ GPa} \quad \nu = 0.33 \quad \text{Thickness} = 1 \text{ mm}$$

$$U_1 = 2 \cos(2\pi t) \sin\left(\frac{\pi y}{35}\right)$$

Boundary condition

Exercise: Disp + Dload Subroutine

Simulation time: 1(s)

B.C's

$$@ x=70 \Rightarrow U_1 = 0, U_2 = 0, U_3 = \sin\left(\frac{\pi y}{35}\right)$$

$$@ x=-70 \Rightarrow U_1 = 0, U_2 = 0, U_3 = -\sin\left(\frac{\pi y}{35}\right)$$

$$@ y=35 \Rightarrow U_1 = 0, U_2 = 0, U_3 = \sin\left(\frac{\pi x}{70}\right)$$

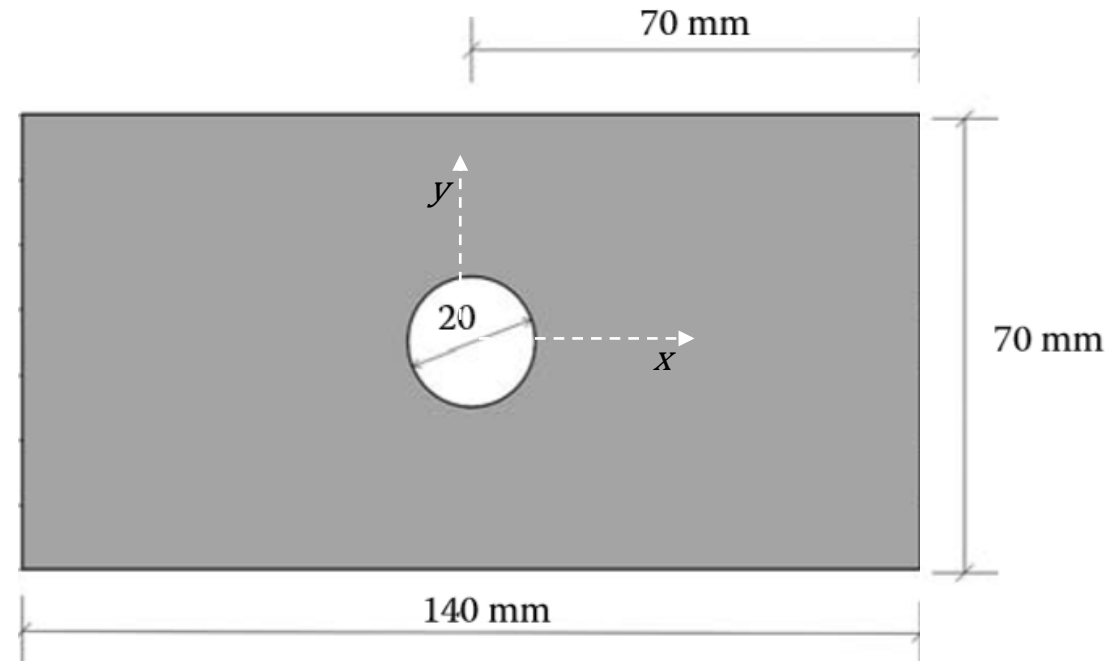
$$@ y=-35 \Rightarrow U_1 = 0, U_2 = 0, U_3 = -\sin\left(\frac{\pi x}{70}\right)$$

$$@ x^2 + y^2 = 100 \Rightarrow U_3 = e^{-0.1t}$$

Pressure

$$\text{exert on right half: } P(x, y, t) = \sin\left(\frac{\pi x}{70}\right) \sin\left(\frac{\pi y}{35}\right) \cos(\pi t)$$

$$\text{exert on left half: } P(x, y, t) = -\sin\left(\frac{\pi x}{70}\right) \sin\left(\frac{\pi y}{35}\right) \cos(\pi t)$$



Material properties: $E=210 \text{ GPa}$ $\nu = 0.3$ Thickness=2 mm

Natural Frequency Problem

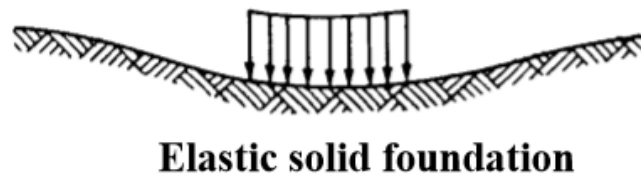
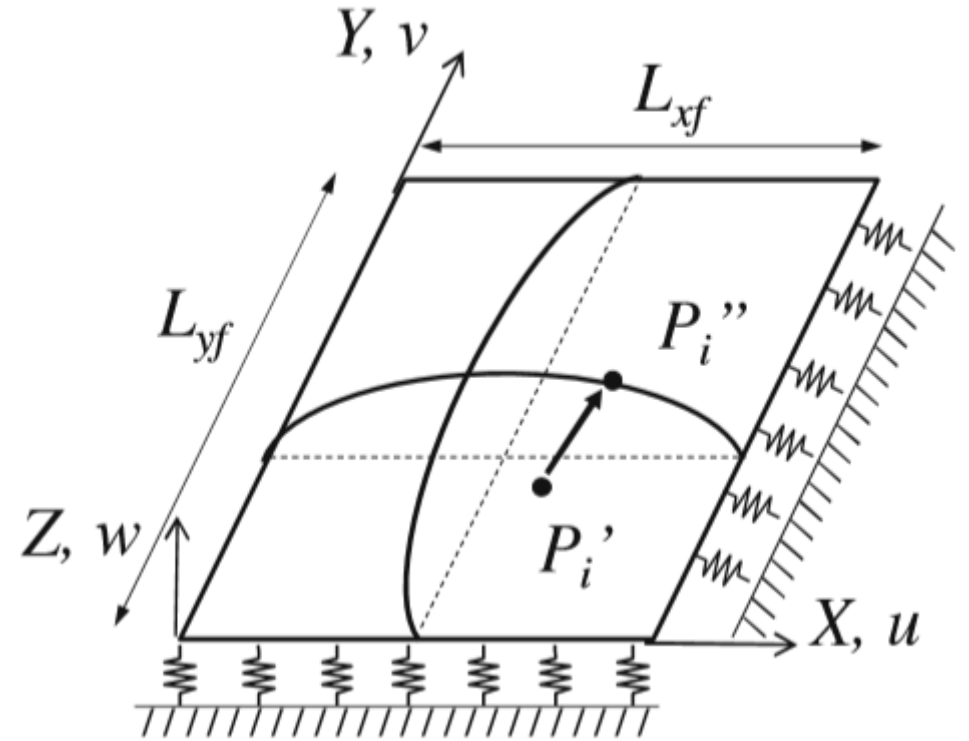
$$L_{x0} = 1 \text{ m}, \quad L_{y0} = 1 \text{ m}, \quad \text{thickness: } h = 1 \text{ mm}$$

$$\text{Mooney-Rivlin: } C_1 = 0.1361 \text{ MPa}; \quad C_2 = 0.00806 \text{ MPa}$$

$$\rho = 1200 \text{ kg/m}^3$$

$$k_1 = 10^3 \text{ N/m}^3$$

$$L_{xf} = 1.1 \text{ m}, \quad L_{yf} = 1.1 \text{ m}$$



Exercise

It is desired to simulate the last problem with non-linear Winker foundation

$$k_1 = 10^3 \left(\frac{N}{m^3} \right), \quad k_3 = 10^2 \left(\frac{N}{m^5} \right)$$

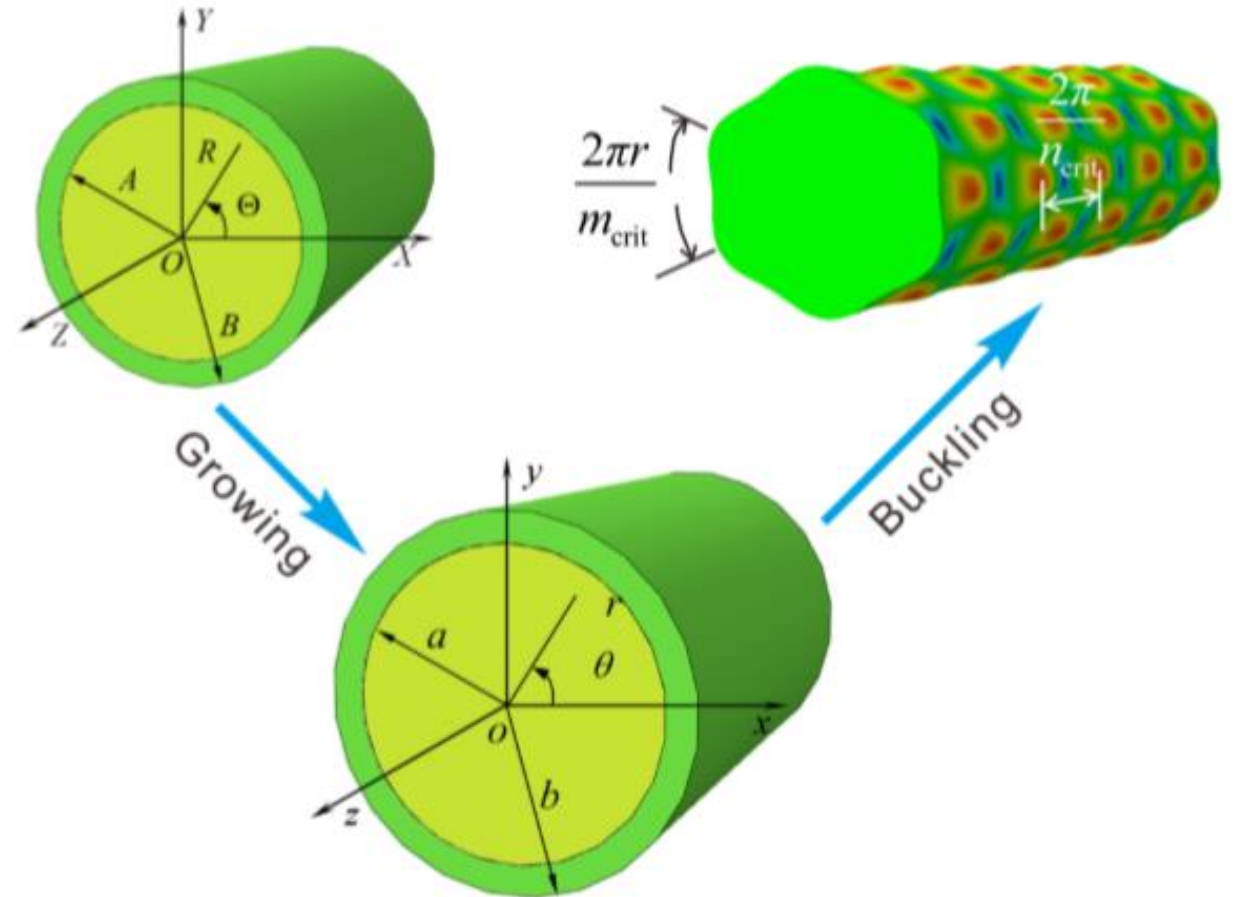
Wrinkling Patterns

$$B = 20 \text{ mm}$$

$$\text{Constitutive law: neo - Hookean} \quad \frac{C_S}{C_C} = 20$$

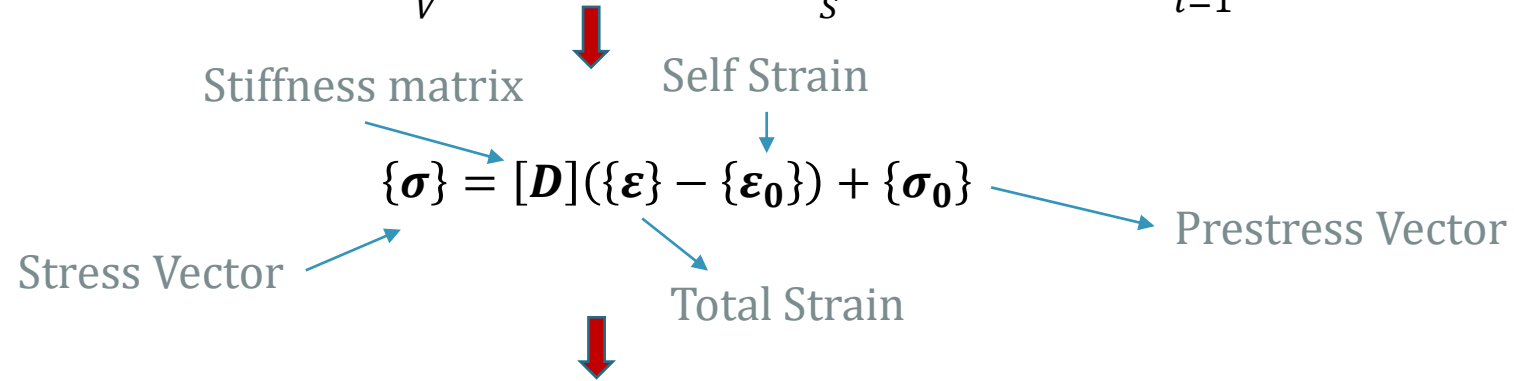
$$\text{Thickness - radius ratio: } \frac{H}{B} = 0.05 \quad (H = 1 \text{ mm})$$

$$L/B = 10$$



Exercise

$$\delta U = \delta W_{ext} \Rightarrow \iiint_V \{(\delta \epsilon)\}^T \{\sigma\} dV = \iiint_V \{\delta U\}^T \{F_b\} dV + \iint_S \{\delta U\}^T \{T\} dS + \sum_{i=1}^n \{\delta U\}^T \{F_p\}$$



Elastic strain energy

Prestress energy

Surface Traction work

$$\iiint_V \{\delta \epsilon\}^T [D] \{\epsilon\} dV - \iiint_V \{\delta \epsilon\}^T [D] \{\epsilon_0\} dV + \iiint_V \{\delta \epsilon\}^T \{\sigma_0\} dV - \iiint_V \{\delta U\}^T \{F_b\} dV - \iint_S \{\delta U\}^T \{T\} dS - \sum_{i=1}^n \{\delta U\}^T \{F_p\} = 0$$

Self strain energy

Body force work

Point Load work

$$\left(\iiint_V [B]^T [D] [B] dV \right) \{a\} = \iiint_V [B]^T [D] \{\epsilon_0\} dV - \iiint_V [B]^T \{\sigma_0\} dV + \iiint_V [N]^T \{F_b\} dV + \iint_S [N]^T \{T\} dS + \sum_{i=1}^n [N]^T \{F_p\}$$

Exercise

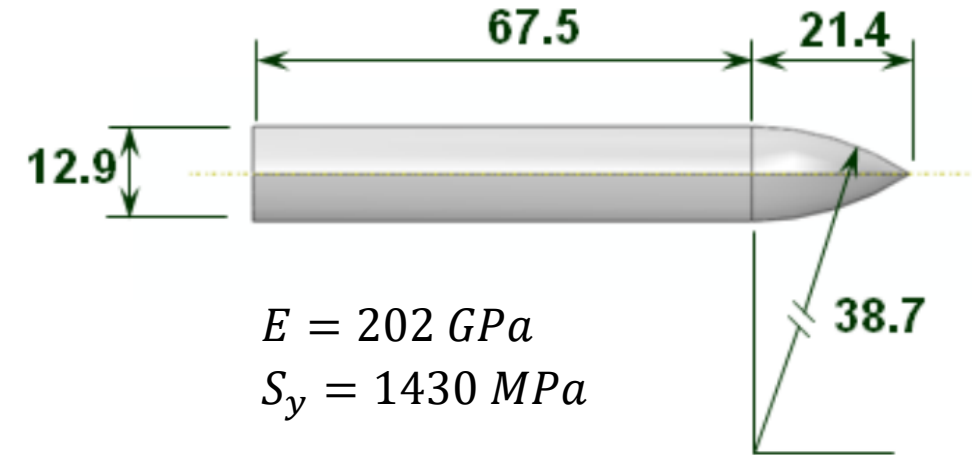
This exercise is associated with Simulating the “Wrinkling Problem” without the implementation of thermal expansion.

Ballistic Perforation

Key Abaqus/Explicit Features and Benefits

- Mie-Grüneisen equation of state to model materials at high pressure
- Johnson-Cook plasticity model that accounts for strain rate, thermal effects and compressibility
- Johnson-Cook dynamic failure model within the Abaqus ductile damage initiation criterion for metals
- Progressive damage framework

$E = 69 \text{ GPa}$ Thickness = 26.3 304 mm $\theta = 30^\circ$
 $S_y = 262 \text{ MPa}$



	Shear Modulus		Mass Density		Specific Heat
1	80100	1	0.00783	1	477

	A	B	n	m	Melting Temp	Transition Temp		C	Epsilon dot zero
1	1430	2545	0.7	1.03	1793	293.2	1	0.014	15

	c0	s	Gamma0
1	4578	1.33	1.67

d1	d2	d3	d4	d5	Melting Temperature	Transition Temperature	Reference Strain Rate
0.05	3.44	2.12	0.002	0.61	1793	293.2	15

mm	g	ms	N	MPa	mJ	m/s	1e+03 m/s ²
----	---	----	---	-----	----	-----	------------------------

Progressive Folding



Static Analysis along with Artificial Damping

Static Analysis along with Imperfection

Buckling and Post-Buckling Analysis

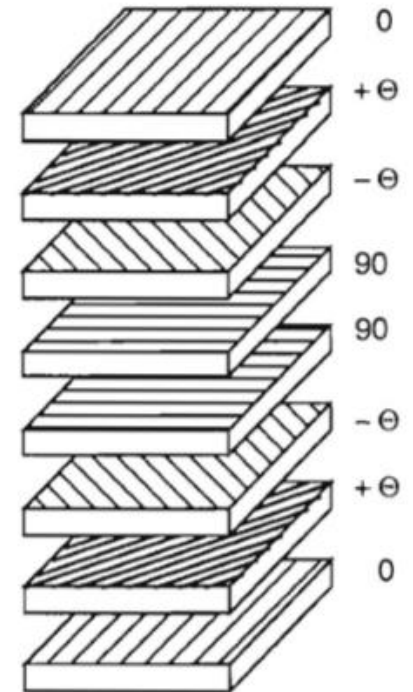
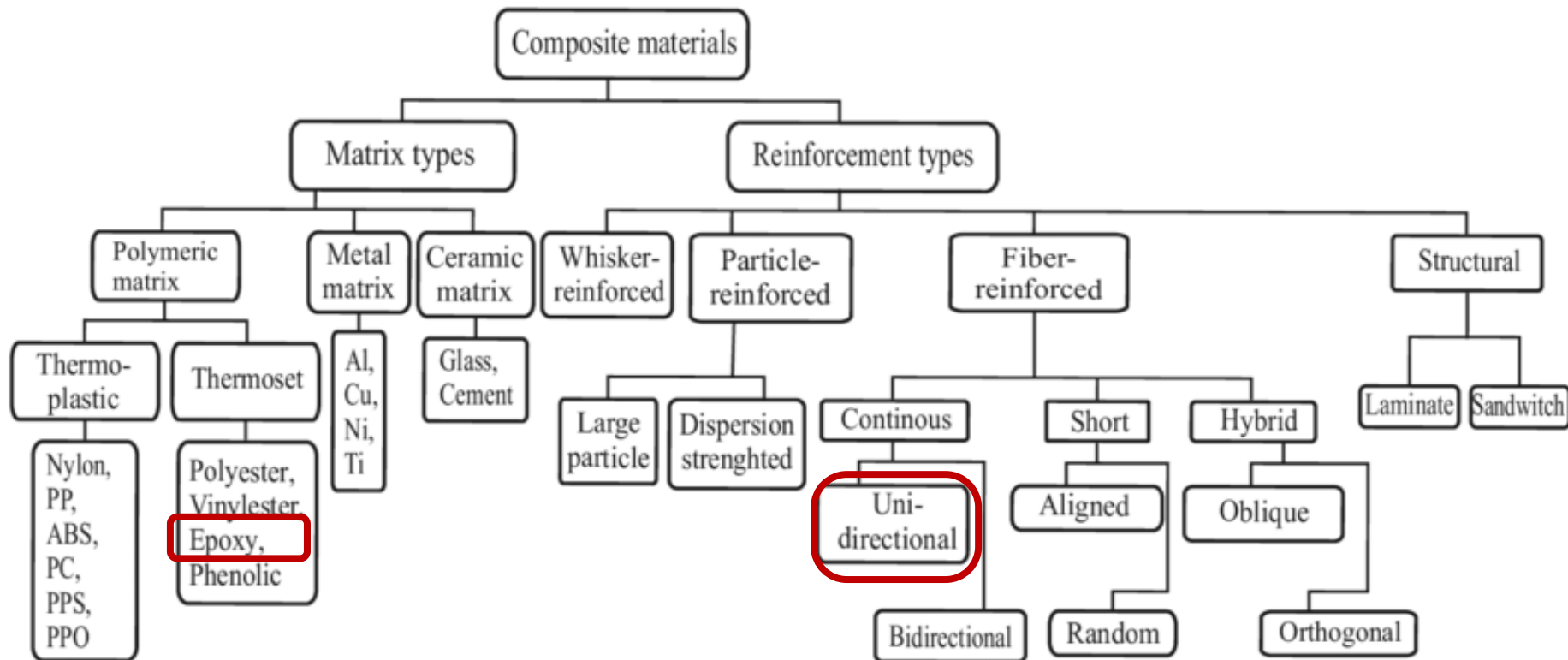
Dynamic Analysis

Exercise

Low Velocity Impact of Composite Tube

1. Composite Material:

A composite material is a material made from two or more constituent materials with significantly different physical or chemical properties that, when combined, produce a material with characteristics different from the individual components



Low Velocity Impact of Composite Tube

Isotropic elasticity

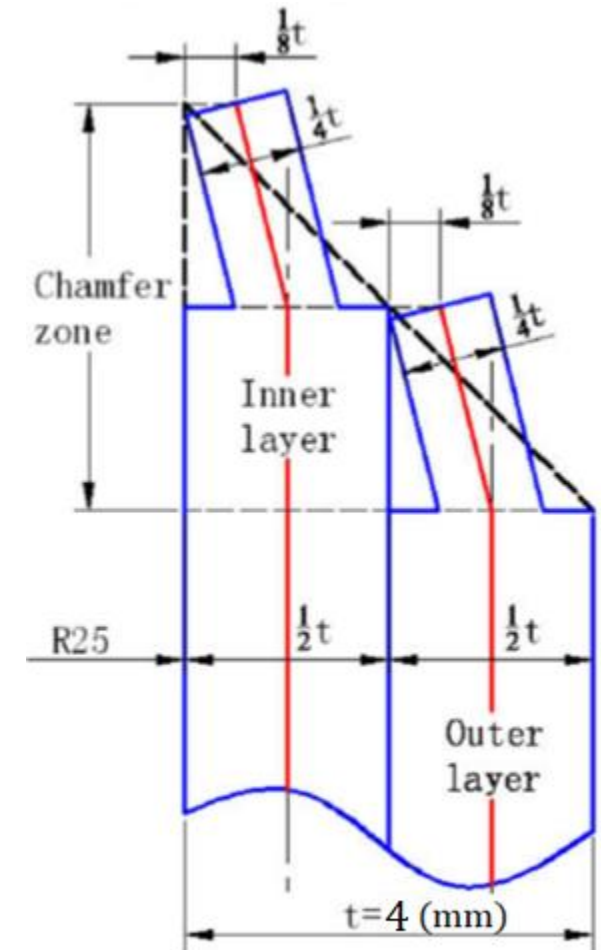
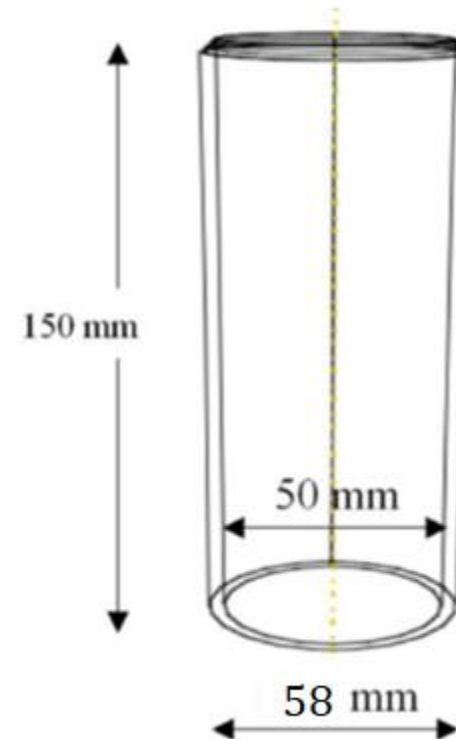
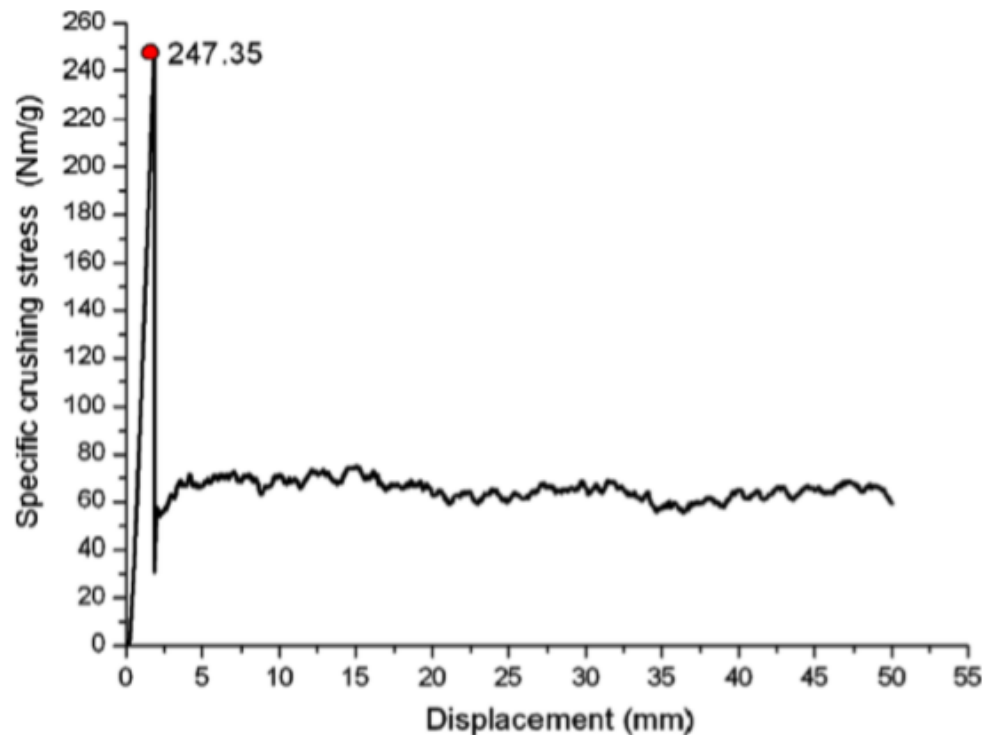
$$\begin{Bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{33} \\ \gamma_{12} \\ \gamma_{13} \\ \gamma_{23} \end{Bmatrix} = \begin{bmatrix} 1/E & -\nu/E & -\nu/E & 0 & 0 & 0 \\ -\nu/E & 1/E & -\nu/E & 0 & 0 & 0 \\ -\nu/E & -\nu/E & 1/E & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/G & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/G & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/G \end{bmatrix} \begin{Bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \\ \sigma_{13} \\ \sigma_{23} \end{Bmatrix}$$

Orthotropic elasticity by specifying the engineering constants

$$\begin{Bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{33} \\ \gamma_{12} \\ \gamma_{13} \\ \gamma_{23} \end{Bmatrix} = \begin{bmatrix} 1/E_1 & -\nu_{21}/E_2 & -\nu_{31}/E_3 & 0 & 0 & 0 \\ -\nu_{12}/E_1 & 1/E_2 & -\nu_{32}/E_3 & 0 & 0 & 0 \\ -\nu_{13}/E_1 & -\nu_{23}/E_2 & 1/E_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/G_{13} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/G_{23} \end{bmatrix} \begin{Bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \\ \sigma_{13} \\ \sigma_{23} \end{Bmatrix}$$

Low Velocity Impact of Composite Tube

- › Axial low velocity impact of two shell layers with 45 degree trigger
- › Impactor: $m = 380 \text{ kg}$ $v = 4 \text{ m/s}$



Low Velocity Impact of Composite Tube

Specific Energy Absorption:
$$SEA = \frac{E_{total}}{M_c} = \frac{\int_0^{L_c} P ds}{\rho A L_c}$$

Carbon Fiber Reinforced Polymer

Table 1

Orthotropic elastic properties of fibre-reinforced epoxy [26,27].

Materials	E_1 (MPa)	$E_2 = E_3$ (MPa)	$G_{12} = G_{13}$ (MPa)	G_{23} (MPa)	ν_{12}
CFRP	153,000	10,300	6000	3700	0.3
GFRP	55,000	9500	5500	3000	0.33

Ply orientation: $[+45/-45/90/0/0/90/0]_s$

$$\rho = 1530 \text{ kg/m}^3$$

Table 2

Orthotropic damage initiation properties of fiber-reinforced epoxy [26,27].

Materials	X^T (MPa)	X^C (MPa)	Y^T (MPa)	Y^C (MPa)	S^{12} (MPa)	S^{23} (MPa)
CFRP	2537	1580	82	293	90	40
GFRP	2500	2000	50	150	50	50

Table 3

Fracture energies for fiber-reinforced epoxy [26,27].

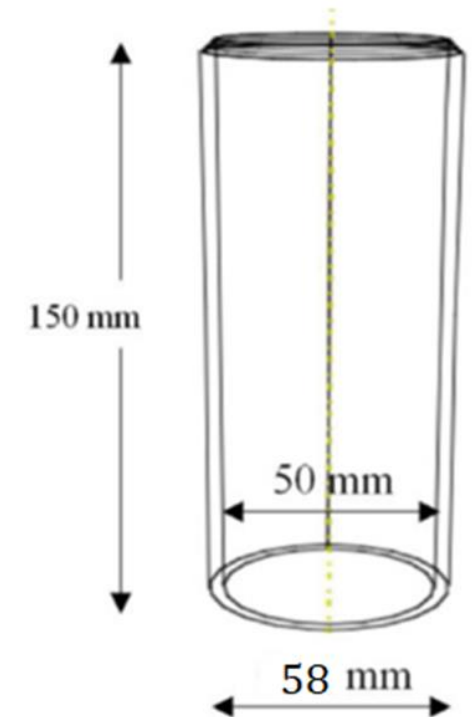
Materials	$G_{ft,c}$ (kJ/m ²)	$G_{fc,c}$ (kJ/m ²)	$G_{mt,c}$ (kJ/m ²)	$G_{mc,c}$ (kJ/m ²)
CFRP	91.6	79.9	0.22	1.1
GFRP	12.5	12.5	1.0	1.0

Exercise

It is desired to simulate the “low velocity impact of composite tube”(just 2 layers) with cohesive behavior to observe delamination. The SEA should be calculated with respect to force- displacement diagram

Material properties of the adhesive in cohesive surface [26,27].

Property	Material	Mode I	Mode II	Mode III
Normalized elastic modulus (N/m)	CFRP	9.80952e + 12	5.71429e + 12	5.71429e + 12
	GFRP	9.04762e + 12	5.2381e + 12	5.2381e + 12
Strength (MPa)	CFRP	70	70	70
	GFRP	70	70	70
Fracture toughness (J/m ²)	CFRP	700	700	700
	GFRP	4000	4000	4000



Specific Energy Absorption: $SEA = \frac{E_{total}}{M_c} = \frac{\int_0^{L_c} P ds}{\rho A L_c}$

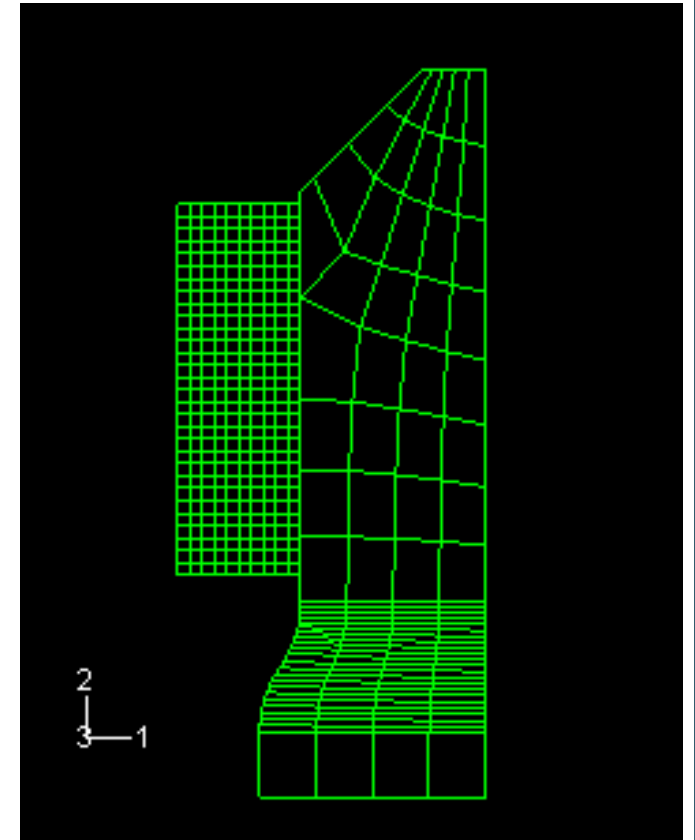
Hint

Linear elastic traction-separation behavior

$$\mathbf{t} = \begin{Bmatrix} t_n \\ t_s \\ t_t \end{Bmatrix} = \begin{bmatrix} K_{nn} & K_{ns} & K_{nt} \\ K_{ns} & K_{ss} & K_{st} \\ K_{nt} & K_{st} & K_{tt} \end{bmatrix} \begin{Bmatrix} \delta_n \\ \delta_s \\ \delta_t \end{Bmatrix} = \mathbf{K} \boldsymbol{\delta}.$$

Extrusion

Abaqus Example Problem has provided this problem



Exercise

Fluid-structure Interaction

Impact of a Drink Can

Approach {

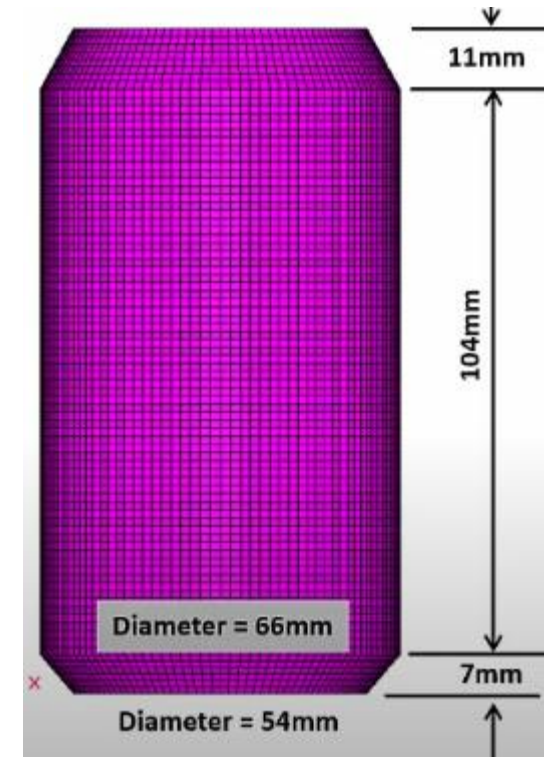
1. Coupled Eulerian-Lagrangian Analysis
2. Smoothed Particle Hydrodynamic (SPH) Analysis

Table 1. Material parameters for aluminum.

Parameter	Value
Density (ρ)	2700 kg/m ³
Young's modulus	69 GPa
Poisson's ratio	0.33
Yield stress	276 MPa

Table 2. Material parameters for water.

Parameter	Value
Density (ρ)	9.96×10^{-7} kg/mm ³
Viscosity (η)	1×10^{-8} N s/mm ²
c_0	1.45×10^6 mm/s
s	0
Γ_0	0



Impact angle: 30 degree
Height: 30 cm
Thickness: 1 mm

Exercise

$$[M(t, \{a\}, \{\dot{a}\}, \{\ddot{a}\})]\{\ddot{a}(t)\} + [C(t, \{a\}, \{\dot{a}\}, \{\ddot{a}\})]\{\dot{a}(t)\} + [K(t, \{a\}, \{\dot{a}\}, \{\ddot{a}\})]\{a(t)\} = F(t, \{a\}, \{\dot{a}\}, \{\ddot{a}\})$$