Simulation and Analysis of 1D Wave Propagation under Various Physical Models

Dario Liotta

Università degli Studi di Padova

Dipartimento di Fisica e Astronomia Galileo Galilei

September 6th 2025

Course of **Quantum Information and Computing**

Academic Year 2024/2025

Numerical methods for differential equations

Ordinary Differential Equations

- Euler Method
- Crank-Nicolson Method
- Trotter-Suzuki Formula

Runge-Kutta

Partial Differential Equations

- Finite Difference Method
- Finite Volume Method
- Galerkin methods
  - Spectral Method
  - Finite Element Method

Numerical methods for differential equations

Ordinary **D**ifferential **E**quations

- Euler Method
- Crank-Nicolson Method
- Trotter-Suzuki Formula

$\underbrace{\text{Euler Method} \quad \text{Crank-Nicolson Method}}_{\text{Runge-Kutta}}$

Partial **D**ifferential **E**quations

- **F**inite **D**ifference **M**ethod
- **F**inite **V**olume **M**ethod
- Galerkin methods
  - Spectral Method
  - $\boxed{\text{Finite Element Method}}$

Introduction to the problem

Solving a **PDE** means to find a function u such that

$$\mathcal{L}u = f$$

where $\mathcal{L}$ is a differential operator and f is a source term.

The equation holds in a domain Ω and is completed by prescribing **boundary conditions** on $\partial\Omega$.

In most physical applications $\mathcal{L}$ is a second-order operator

- **Poisson** equation: $\mathcal{L} = -\Delta$
- **Heat** equation: $\mathcal{L} = \frac{\partial}{\partial t} - \Delta$
- **Wave** equation: $\mathcal{L} = \frac{\partial^2}{\partial t^2} - c^2\Delta$

Introduction to the problem

Solving a **PDE** means to find a function u such that

$$\mathcal{L}u = f$$

where $\mathcal{L}$ is a differential operator and f is a source term.

The equation holds in a domain Ω and is completed by prescribing **boundary conditions** on $\partial\Omega$.

In most physical applications $\mathcal{L}$ is a second-order operator

- **Poisson** equation: $\mathcal{L} = -\Delta$
- **Heat** equation: $\mathcal{L} = \frac{\partial}{\partial t} - \Delta$
- **Wave** equation: $\mathcal{L} = \frac{\partial^2}{\partial t^2} - c^2\Delta$

Weak formulation

Galerkin methods rely on a **weak formulation**

- Multiply by a test function v and integrate over the entire domain

$$-\int_{\Omega}(\Delta u)v d\Omega = \int_{\Omega} f v d\Omega$$

- Integrate by parts the left hand side

$$-\int_{\Omega}(\Delta u)v d\Omega = \int_{\Omega} \nabla u \cdot \nabla v d\Omega - \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

- Substitute and get the new expression

$$\int_{\Omega} \nabla u \cdot \nabla v d\Omega = \int_{\Omega} f v d\Omega + \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

Weak formulation

Galerkin methods rely on a **weak formulation**

- Multiply by a test function v and integrate over the entire domain

$$-\int_\Omega (\Delta u) v d\Omega = \int_\Omega f v d\Omega$$

- Integrate by parts the left hand side

$$-\int_\Omega (\Delta u) v d\Omega = \int_\Omega \nabla u \cdot \nabla v d\Omega - \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

- Substitute and get the new expression

$$\int_\Omega \nabla u \cdot \nabla v d\Omega = \int_\Omega f v d\Omega + \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

Weak formulation

Galerkin methods rely on a **weak formulation**

- Multiply by a test function v and integrate over the entire domain

$$-\int_{\Omega} (\Delta u) v d\Omega = \int_{\Omega} f v d\Omega$$

- Integrate by parts the left hand side

$$-\int_{\Omega} (\Delta u) v d\Omega = \int_{\Omega} \nabla u \cdot \nabla v d\Omega - \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

- Substitute and get the new expression

$$\int_{\Omega} \nabla u \cdot \nabla v d\Omega = \int_{\Omega} f v d\Omega + \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

Weak formulation

Galerkin methods rely on a **weak formulation**

- Multiply by a test function v and integrate over the entire domain

$$-\int_\Omega (\Delta u) v d\Omega = \int_\Omega f v d\Omega$$

- Integrate by parts the left hand side

$$-\int_\Omega (\Delta u) v d\Omega = \int_\Omega \nabla u \cdot \nabla v d\Omega - \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

- Substitute and get the new expression

$$\int_\Omega \nabla u \cdot \nabla v d\Omega = \int_\Omega f v d\Omega + \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds$$

About the test function

The test function v is introduced to check whether the PDE is satisfied *on average* throughout the domain.

The problem becomes to find u such that

$$a(u,v) = F(v) \qquad \forall v \in V$$

where

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v d\Omega \qquad \text{is a bilinear form}$$

$$F(v) = \int_\Omega f v d\Omega + \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds \qquad \text{is a linear functional}$$

About the test function

The test function v is introduced to check whether the PDE is satisfied *on average* throughout the domain.

The problem becomes to find u such that

$$a(u,v) = F(v) \qquad \forall v \in V$$

where

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v d\Omega \qquad \text{is a bilinear form}$$

$$F(v) = \int_\Omega f v d\Omega + \int_{\partial\Omega} \frac{\partial u}{\partial n} v ds \qquad \text{is a linear functional}$$

Benefits of the weak formulation

Strong formulation	Weak formulation
$u \in C^2(\Omega)$	$u, v \in H^1(\Omega)$*
Holds pointwise in Ω	Holds on average on Ω
Derivatives exist classically	Derivatives exist in the distributional sense

In short: weak formulation requires **less regularity**

* $H^1(\Omega)$ is a **Sobolev space** of functions with square-integrable first derivatives:

$$w \in H^1(\Omega) = \left\{ w \in L^2(\Omega) \mid \nabla w \in L^2(\Omega)^d \right\}$$

Benefits of the weak formulation

Strong formulation	Weak formulation
$u \in C^2(\Omega)$	$u, v \in H^1(\Omega)$*
Holds pointwise in Ω	Holds on average on Ω
Derivatives exist classically	Derivatives exist in the distributional sense

In short: weak formulation requires **less regularity**

* $H^1(\Omega)$ is a **Sobolev space** of functions with square-integrable first derivatives:

$$w \in H^1(\Omega) = \left\{ w \in L^2(\Omega) \mid \nabla w \in L^2(\Omega)^d \right\}$$

On boundary conditions

Another difference lies in the boundary condition prescription.

In strong formulation

→ **Dirichlet**: $u = g$ on $\partial\Omega$

→ **Neumann**: $\frac{\partial u}{\partial n} = h$ on $\partial\Omega$

In weak formulation

Dirichlet becomes **essential condition**	Neumann becomes **natural condition**
$\boldsymbol{v = 0}$ on $\partial\Omega \Rightarrow$ cancels boundary term (no information available on $\frac{\partial u}{\partial n}$)	$\boldsymbol{v}$ **free** on $\partial\Omega$
$\boldsymbol{u = g}$ enforced on $\partial\Omega$ (final solution)	$\boldsymbol{\frac{\partial u}{\partial n} = h}$ naturally enters weak form

On boundary conditions

Another difference lies in the boundary condition prescription.

In strong formulation

- **Dirichlet**: $u = g \quad \text{on } \partial\Omega$
- **Neumann**: $\frac{\partial u}{\partial n} = h \quad \text{on } \partial\Omega$

In weak formulation

Dirichlet becomes **essential condition**	Neumann becomes **natural condition**
$\boldsymbol{v} = \mathbf{0}$ on $\partial\Omega \Rightarrow$ cancels boundary term (no information available on $\frac{\partial u}{\partial n}$)	$\boldsymbol{v}$ **free** on $\partial\Omega$
$\boldsymbol{u} = \boldsymbol{g}$ enforced on $\partial\Omega$ (final solution)	$\frac{\partial u}{\partial n} = \boldsymbol{h}$ naturally enters weak form

On boundary conditions

Another difference lies in the boundary condition prescription.

In strong formulation

- **Dirichlet**: $u = g \quad \text{on } \partial\Omega$
- **Neumann**: $\frac{\partial u}{\partial n} = h \quad \text{on } \partial\Omega$

In weak formulation

Dirichlet becomes **essential condition**	Neumann becomes **natural condition**
$\boldsymbol{v} = \mathbf{0}$ on $\partial\Omega \Rightarrow$ cancels boundary term (no information available on $\frac{\partial u}{\partial n}$)	$\boldsymbol{v}$ **free** on $\partial\Omega$
$\boldsymbol{u} = \boldsymbol{g}$ enforced on $\partial\Omega$ (final solution)	$\boldsymbol{\frac{\partial u}{\partial n} = h}$ naturally enters weak form

Shape functions

Galerkin methods allow to find an approximate solution

$$u_h \in V_h \subset H^1(\Omega) \qquad \text{where } V_h \text{ is a } \textbf{finite-dimensional} \text{ space}$$

In this framework, the goal is to find u_h such that

$$\boxed{a\left(u_h, v_h\right) = F\left(v_h\right) \qquad \forall v_h \in V_h}$$

A **basis of function** $\{\phi_i\}$ is chosen to express u_h and to use it as test:

$$u_h = \sum_{j=1}^{N} u_j \phi_j \implies a\left(\sum_{j=1}^{N} u_j \phi_j, \phi_i\right) = F\left(\phi_i\right) \qquad \forall i = 1, \ldots, N$$

Functions ϕ_i model the solution $\longrightarrow$ **shape functions**

Shape functions

Galerkin methods allow to find an approximate solution

$$u_h \in V_h \subset H^1(\Omega) \qquad \text{where } V_h \text{ is a } \textbf{finite-dimensional} \text{ space}$$

In this framework, the goal is to find u_h such that

$$\boxed{a\left(u_h, v_h\right) = F\left(v_h\right) \qquad \forall v_h \in V_h}$$

A **basis of function** $\{\phi_i\}$ is chosen to express u_h and to use it as test:

$$u_h = \sum_{j=1}^{N} u_j \phi_j \implies a\left(\sum_{j=1}^{N} u_j \phi_j, \phi_i\right) = F\left(\phi_i\right) \qquad \forall i = 1, \ldots, N$$

Functions ϕ_i model the solution $\longrightarrow$ **shape functions**

Shape functions

Galerkin methods allow to find an approximate solution

$$u_h \in V_h \subset H^1(\Omega) \qquad \text{where } V_h \text{ is a } \textbf{finite-dimensional} \text{ space}$$

In this framework, the goal is to find u_h such that

$$\boxed{a\left(u_h, v_h\right) = F\left(v_h\right) \qquad \forall v_h \in V_h}$$

A **basis of function** $\{\phi_i\}$ is chosen to express u_h and to use it as test:

$$u_h = \sum_{j=1}^{N} u_j \phi_j \implies a\left(\sum_{j=1}^{N} u_j \phi_j, \phi_i\right) = F\left(\phi_i\right) \qquad \forall i = 1, \ldots, N$$

Functions ϕ_i model the solution $\longrightarrow$ **shape functions**

Final expression

By linearity of $a(\cdot,\cdot)$, the problem reduces to a **finite linear system**:

$$\sum_{j=1}^{N} u_j a\left(\phi_j, \phi_i\right) = F\left(\phi_i\right) \qquad \forall i = 1, \ldots, N$$

$$\Downarrow$$

$$\boxed{A\boldsymbol{u} = \boldsymbol{F}}$$

where

$A_{i,j} = a\left(\phi_j, \phi_i\right)$ form the **stiffness matrix**

$\boldsymbol{u} = \left(u_1, \ldots, u_N\right)^T$ is the **vector of unknowns**

$\boldsymbol{F} = \left(F\left(\phi_1\right), \ldots, F\left(\phi_N\right)\right)^T$ is the **load vector**

Final expression

By linearity of $a(\cdot,\cdot)$, the problem reduces to a **finite linear system**:

$$\sum_{j=1}^{N} u_j a\left(\phi_j, \phi_i\right) = F\left(\phi_i\right) \qquad \forall i = 1, \dots, N$$

$$\Downarrow$$

$$\boxed{A\boldsymbol{u} = \boldsymbol{F}}$$

where

$$A_{i,j} = a\left(\phi_j, \phi_i\right) \qquad \text{form the } \textbf{stiffness matrix}$$

$$\boldsymbol{u} = \left(u_1, \dots, u_N\right)^T \qquad \text{is the } \textbf{vector of unknowns}$$

$$\boldsymbol{F} = \left(F\left(\phi_1\right), \dots, F\left(\phi_N\right)\right)^T \qquad \text{is the } \textbf{load vector}$$

Mesh discretization

FEM approach consists in the subdivision of the domain in a so-called **mesh**

This choice brings several advantages:

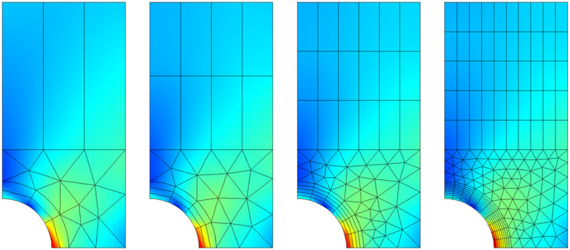
- Good approximation of **complex geometries**
- Better capture of **local effects**
- Possibility of **adaptive refinement**
- Natural construction of a **global solution**

Mesh discretization

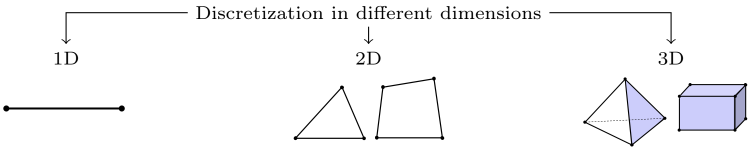
FEM approach consists in the subdivision of the domain in a so-called **mesh**

This choice brings several advantages:

- Good approximation of **complex geometries**
- Better capture of **local effects**
- Possibility of **adaptive refinement**
- Natural construction of a **global solution**



Elements

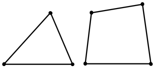


Elements

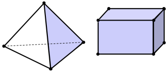
Discretization in different dimensions

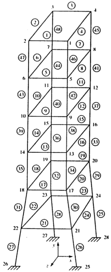
1D

2D

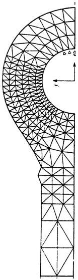


3D

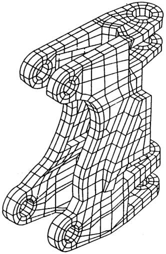




Frame elements



Triangular elements

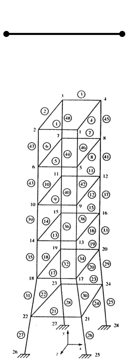


Brick elements

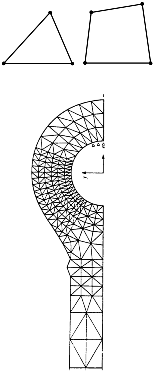
Elements

Discretization in different dimensions

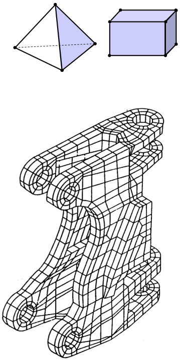
1D | 2D | 3D



Frame elements



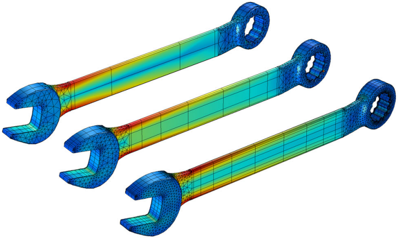
Triangular elements



Brick elements

Finite Element Method

Application examples

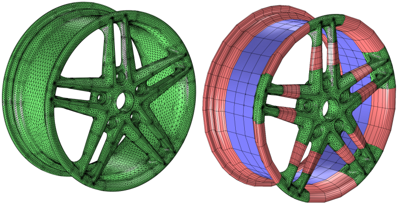


Manual mesh refinement of a wrench using different element types

`Image from COMSOL Multiplysics Cyclopedia, ‘‘Finite Element Mesh Refinement’’, 21st of February 2017`

Mesh of a wheel rim composed of tetrahedrons in green, bricks in blue and prisms in pink

`Image from COMSOL Multiplysics Blog, ‘‘Meshing Your Geometry: When to Use the Various Element Types’’, Walter Frei, 4th of November 2013`



Choise of the base

Mesh division into sub-domains

↓

Choice of a **local basis functions**

↓

Exploitation of **compact support**

- Leads to **sparse matrices**
- Allows **local interpolation**
- Enhances **numerical stability**
- Enables **efficient parallelization**

Choise of the base

Mesh division into sub-domains

↓

Choice of a **local basis functions**

↓

Exploitation of **compact support**

- Leads to **sparse matrices**
- Allows **local interpolation**
- Enhances **numerical stability**
- Enables **efficient parallelization**

Choise of the base

Mesh division into sub-domains

↓

Choice of a **local basis functions**

↓

Exploitation of **compact support**

- Leads to **sparse matrices**
- Allows **local interpolation**
- Enhances **numerical stability**
- Enables **efficient parallelization**

Choise of the base

Mesh division into sub-domains

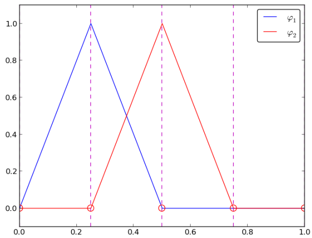
↓

Choice of a **local basis functions**

↓

Exploitation of **compact support**

- Leads to **sparse matrices**
- Allows **local interpolation**
- Enhances **numerical stability**
- Enables **efficient parallelization**



FEniCS library

A leading software platform for finite element computations is **FEniCS**.