

# Class 1: Lagrange mechanics and first steps into numerical integration

Florence Bertails-Descoubes <sup>1</sup>, Mélina Skouras <sup>2</sup>, Mickaël Ly <sup>3</sup>

*inria*



2020, September 8 - ENS Lyon

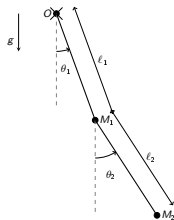
---

<sup>1</sup> [florence.descoubes@inria.fr](mailto:florence.descoubes@inria.fr)

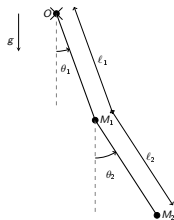
<sup>2</sup> [melina.skouras@inria.fr](mailto:melina.skouras@inria.fr)

<sup>3</sup> [mickael.ly@inria.fr](mailto:mickael.ly@inria.fr)


## Example: the double pendulum



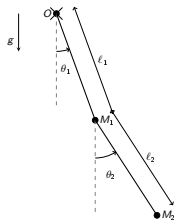
## Example: the double pendulum




## Questions

- How to formulate the equations of motion of this object?
- How to solve these equations in a “safe” way ? 

## Example: the double pendulum

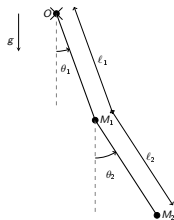


## Questions


- How to formulate the equations of motion of this object?
- How to solve these equations in a “safe” way ? 

Goal of this lecture: understand there are many possible choices and learn best practices

## Example: the double pendulum



### Questions

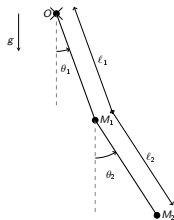
- How to formulate the equations of motion of this object?
- How to solve these equations in a “safe” way ? 

Goal of this lecture: understand there are many possible choices and learn best practices


### Outline of the class

- I. Lagrange mechanics: how to **formulate** the equations of motion
- II. Finite differences: how to **solve** the equations of motion

## Example: the double pendulum





### Questions

- How to formulate the equations of motion of this object?
- How to solve these equations in a “safe” way ? 

Goal of this lecture: understand there are many possible choices and learn best practices

### Outline of the class

- I. Lagrange mechanics: how to **formulate** the equations of motion
- II. Finite differences: how to **solve** the equations of motion

NB: Once these concepts are known, more complex systems can be considered.  

# Keywords and bibliography

## Part I: Lagrange mechanics

- **Analytical Mechanics**

Landau and Lifshitz, Mechanics Vol 1; J. Fereira, Mécanique analytique

- **Calculus of Variations**

J.-P. Bourguignon, Calcul variationnel (“Variational calculus”)

## Part II: Finite differences

- **Numerical Analysis**

G. Allaire, Analyse numérique et optimisation (“Numerical analysis and optimization”)

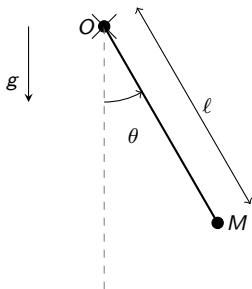
- **Energy conservation properties**

E. Hairer et al., Geometric numerical integration

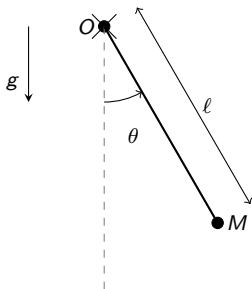
# Part I: Lagrange mechanics



## The simple pendulum



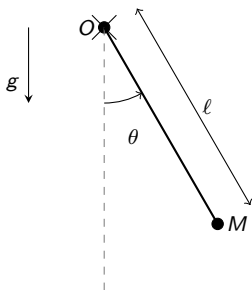
# The simple pendulum



## Exercise

- Write the equations of motion of the simple pendulum...

# The simple pendulum



## Exercise

- Write the equations of motion of the simple pendulum...
- ...using Newton's second law

## White board 1/6

## White board 2/6

## White board 3/6

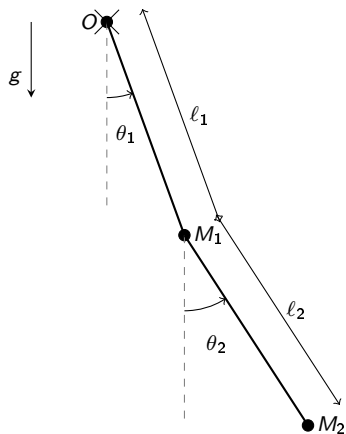
## White board 4/6

## White board 5/6

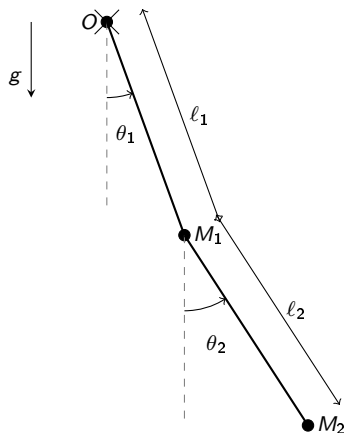


## White board 6/6

## The double pendulum



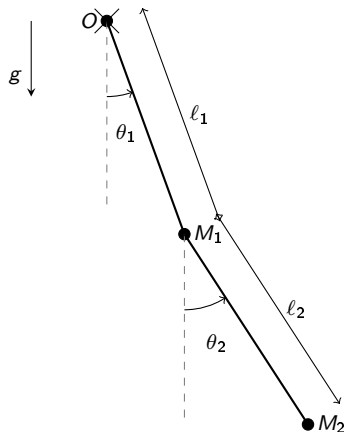
## The double pendulum



## Exercise

- Write the equations of motion of the double pendulum...

## The double pendulum



## Exercise

- Write the equations of motion of the double pendulum...
- ...using Newton's second law

## White board 1/1

## Generalized coordinates

### Definition

**Generalized coordinates**, denoted  $q_i$ , are  $n$  independent variables (functions of time) which allow to characterize the configuration of a system possessing  $n$  degrees of freedom. The **generalized velocities** of the system are defined by  $\frac{d}{dt}q_i = \dot{q}_i$ .

## Generalized coordinates

### Definition

**Generalized coordinates**, denoted  $q_i$ , are  $n$  independent variables (functions of time) which allow to characterize the configuration of a system possessing  $n$  degrees of freedom. The **generalized velocities** of the system are defined by  $\frac{d}{dt}q_i = \dot{q}_i$ .

*Example:* For the simple pendulum, we can take  $q = \theta$  ( $n = 1$ ).

## Generalized coordinates

### Definition

**Generalized coordinates**, denoted  $q_i$ , are  $n$  independent variables (functions of time) which allow to characterize the configuration of a system possessing  $n$  degrees of freedom. The **generalized velocities** of the system are defined by  $\frac{d}{dt}q_i = \dot{q}_i$ .

*Example:* For the simple pendulum, we can take  $q = \theta$  ( $n = 1$ ).

*Remark:* A generalized velocity  $\dot{q}_i$  does not necessarily correspond to the velocity  $\mathbf{v}_M$  of a given material point  $M$ . Reconstructing the configuration and the material velocities of the system corresponds to writing the **kinematics** of the system.



## Generalized coordinates

### Definition

**Generalized coordinates**, denoted  $q_i$ , are  $n$  independent variables (functions of time) which allow to characterize the configuration of a system possessing  $n$  degrees of freedom. The **generalized velocities** of the system are defined by  $\frac{d}{dt}q_i = \dot{q}_i$ .

*Example:* For the simple pendulum, we can take  $q = \theta$  ( $n = 1$ ).

*Remark:* A generalized velocity  $\dot{q}_i$  does not necessarily correspond to the velocity  $\mathbf{v}_M$  of a given material point  $M$ . Reconstructing the configuration and the material velocities of the system corresponds to writing the **kinematics** of the system.

*Example:* For the simple pendulum, the position of the mass  $M$  can be computed as  $\mathbf{OM} = \ell \mathbf{e}_r$  and its velocity  $\mathbf{v}_M$  as  $\mathbf{v}_M = \ell \dot{\theta} \mathbf{e}_\theta$  (in the basis  $(\mathbf{e}_r, \mathbf{e}_\theta)$  defined by  $\theta$ ).

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

The equations relating the  $q_i$  and the  $\dot{q}_i$  to the accelerations  $\ddot{q}_i$  are called the **equations of motion** of the system. They take the form of  $n$  independent second-order differential equations in the functions  $q_i$ .

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

The equations relating the  $q_i$  and the  $\dot{q}_i$  to the accelerations  $\ddot{q}_i$  are called the **equations of motion** of the system. They take the form of  $n$  independent second-order differential equations in the functions  $q_i$ .

Question: how to find a systematic way to compute these equations?

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

The equations relating the  $q_i$  and the  $\dot{q}_i$  to the accelerations  $\ddot{q}_i$  are called the **equations of motion** of the system. They take the form of  $n$  independent second-order differential equations in the functions  $q_i$ .

Question: how to find a systematic way to compute these equations?

- Principle of **virtual work** (see next exercise)

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

The equations relating the  $q_i$  and the  $\dot{q}_i$  to the accelerations  $\ddot{q}_i$  are called the **equations of motion** of the system. They take the form of  $n$  independent second-order differential equations in the functions  $q_i$ .

Question: how to find a systematic way to compute these equations?

- Principle of **virtual work** (see next exercise)
- Principle of **least action**: more general settings, beyond Newtonian dynamics!

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

The equations relating the  $q_i$  and the  $\dot{q}_i$  to the accelerations  $\ddot{q}_i$  are called the **equations of motion** of the system. They take the form of  $n$  independent second-order differential equations in the functions  $q_i$ .

Question: how to find a systematic way to compute these equations?

- Principle of **virtual work** (see next exercise)
- Principle of **least action**: more general settings, beyond Newtonian dynamics!

→ Both principles lead to the so-called **Euler-Lagrange** equations

## Equations of motion

### Equations of motion

At instant  $t$ , having both the  $q_i(t)$  and the  $\dot{q}_i(t)$  is necessary, and also sufficient, to determine the accelerations  $\ddot{q}_i(t)$  of the system at  $t$ , and thus predict the trajectory  $q_i(t_+)$  forward in time,  $t_+ > t$ .

The equations relating the  $q_i$  and the  $\dot{q}_i$  to the accelerations  $\ddot{q}_i$  are called the **equations of motion** of the system. They take the form of  $n$  independent second-order differential equations in the functions  $q_i$ .

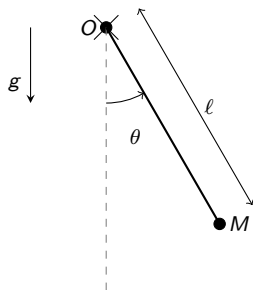
Question: how to find a systematic way to compute these equations?

- Principle of **virtual work** (see next exercise)
- Principle of **least action**: more general settings, beyond Newtonian dynamics!

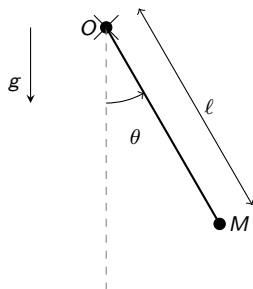
→ Both principles lead to the so-called **Euler-Lagrange** equations  
which take a specific form in Newtonian dynamics



## Back to the simple pendulum



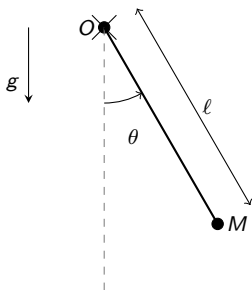
## Back to the simple pendulum



## Exercise

- Write the equations of motion of the simple pendulum...

## Back to the simple pendulum



## Exercise

- Write the equations of motion of the simple pendulum...
- ...using the principle of virtual work

## White board 1/6

## White board 2/6

## White board 3/6

## White board 4/6

## White board 5/6



## White board 6/6

## General method: Least action principle

Let  $q = \{q_0, \dots, q_i, \dots, q_{n-1}\}$  and  $\dot{q} = \frac{d}{dt}q$ .

We consider a system subject to **holonomic** constraints (the  $q_i$  are independent) and **conservative** forces. Let  $T$  be the **kinetic energy** of the system, and  $U$  its **potential energy**.

## General method: Least action principle

Let  $q = \{q_0, \dots, q_i, \dots, q_{n-1}\}$  and  $\dot{q} = \frac{d}{dt}q$ .

We consider a system subject to **holonomic** constraints (the  $q_i$  are independent) and **conservative** forces. Let  $T$  be the **kinetic energy** of the system, and  $U$  its **potential energy**.

### Principle of least action (Hamilton principle)

The actual trajectory  $q(t)$  followed by the system between two instants  $a$  and  $b > a$  should be such that the **action** of the system,

$$S(q, \dot{q}) = \int_a^b \underbrace{\mathcal{L}(q(t), \dot{q}(t), t)}_{T-U} dt,$$

is minimal.

## General method: Least action principle

Let  $q = \{q_0, \dots, q_i, \dots, q_{n-1}\}$  and  $\dot{q} = \frac{d}{dt}q$ .

We consider a system subject to **holonomic** constraints (the  $q_i$  are independent) and **conservative** forces. Let  $T$  be the **kinetic energy** of the system, and  $U$  its **potential energy**.

### Principle of least action (Hamilton principle)

The actual trajectory  $q(t)$  followed by the system between two instants  $a$  and  $b > a$  should be such that the **action** of the system,

$$S(q, \dot{q}) = \int_a^b \underbrace{\mathcal{L}(q(t), \dot{q}(t), t)}_{T-U} dt,$$

is minimal.

$\mathcal{L}(q(t), \dot{q}(t), t) = T(q(t), \dot{q}(t), t) - U(q(t), t)$  is the **Lagrangian** of the system at instant  $t$  (homogeneous to an energy)

## General method: Least action principle

Let  $q = \{q_0, \dots, q_i, \dots, q_{n-1}\}$  and  $\dot{q} = \frac{d}{dt}q$ .

We consider a system subject to **holonomic** constraints (the  $q_i$  are independent) and **conservative** forces. Let  $T$  be the **kinetic energy** of the system, and  $U$  its **potential energy**.

### Principle of least action (Hamilton principle)

The actual trajectory  $q(t)$  followed by the system between two instants  $a$  and  $b > a$  should be such that the **action** of the system,

$$S(q, \dot{q}) = \int_a^b \underbrace{\mathcal{L}(q(t), \dot{q}(t), t)}_{T-U} dt,$$

is minimal.

$\mathcal{L}(q(t), \dot{q}(t), t) = T(q(t), \dot{q}(t), t) - U(q(t), t)$  is the **Lagrangian** of the system at instant  $t$  (homogeneous to an energy), with  $\mathcal{L}$  a real differentiable function

$$\mathcal{L} : \begin{array}{l} \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \longrightarrow \mathbb{R} \\ (x, u, t) \longmapsto \mathcal{L}(x, u, t), \end{array}$$

## General method: Least action principle

Let  $q = \{q_0, \dots, q_i, \dots, q_{n-1}\}$  and  $\dot{q} = \frac{d}{dt}q$ .

We consider a system subject to **holonomic** constraints (the  $q_i$  are independent) and **conservative** forces. Let  $T$  be the **kinetic energy** of the system, and  $U$  its **potential energy**.

### Principle of least action (Hamilton principle)

The actual trajectory  $q(t)$  followed by the system between two instants  $a$  and  $b > a$  should be such that the **action** of the system,

$$S(q, \dot{q}) = \int_a^b \underbrace{\mathcal{L}(q(t), \dot{q}(t), t)}_{T-U} dt,$$

is minimal.

$\mathcal{L}(q(t), \dot{q}(t), t) = T(q(t), \dot{q}(t), t) - U(q(t), t)$  is the **Lagrangian** of the system at instant  $t$  (homogeneous to an energy), with  $\mathcal{L}$  a real differentiable function

$$\mathcal{L} : \begin{array}{l} \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \longrightarrow \mathbb{R} \\ (x, u, t) \longmapsto \mathcal{L}(x, u, t), \end{array}$$

*Remark:*  $S$  is a *functional*, as it takes as arguments the two functions  $q$  and  $\dot{q}$ .

## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous.

## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous. We take the (misused) notation  $\frac{\partial \mathcal{L}}{\partial q_i}$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$  to represent the two partial derivatives of  $\mathcal{L}$  evaluated at  $(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , i.e., such that  $\frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{L}_{x_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = \mathcal{L}_{u_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , respectively.



## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous. We take the (misused) notation  $\frac{\partial \mathcal{L}}{\partial q_i}$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$  to represent the two partial derivatives of  $\mathcal{L}$  evaluated at  $(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , i.e., such that  $\frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{L}_{x_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = \mathcal{L}_{u_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , respectively.

### Theorem

*A necessary condition for the action  $S$  to be minimal is that  $q$  satisfies for all  $t \in [a, b]$  the so-called Euler-Lagrange equations,*

$$\forall i = 0..n, \quad \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = 0.$$

## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous. We take the (misused) notation  $\frac{\partial \mathcal{L}}{\partial q_i}$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$  to represent the two partial derivatives of  $\mathcal{L}$  evaluated at  $(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , i.e., such that  $\frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{L}_{x_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = \mathcal{L}_{u_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , respectively.

### Theorem

*A necessary condition for the action  $S$  to be minimal is that  $q$  satisfies for all  $t \in [a, b]$  the so-called Euler-Lagrange equations,*

$$\forall i = 0..n, \quad \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = 0.$$

Exercise: proof!

## White board 1/6

## White board 2/6

## White board 3/6

## White board 4/6

## White board 5/6

## White board 6/6



## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous. We take the (misused) notation  $\frac{\partial \mathcal{L}}{\partial q_i}$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$  to represent the two partial derivatives of  $\mathcal{L}$  evaluated at  $(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , i.e., such that  $\frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{L}_{x_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = \mathcal{L}_{u_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , respectively.

### Theorem

A necessary condition for the action  $\mathcal{S}$  to be minimal is that  $\mathbf{q}$  satisfies for all  $t \in [a, b]$  the so-called Euler-Lagrange equations,

$$\forall i = 0..n, \quad \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = 0.$$

Exercise: proof!

## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous. We take the (misused) notation  $\frac{\partial \mathcal{L}}{\partial q_i}$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$  to represent the two partial derivatives of  $\mathcal{L}$  evaluated at  $(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , i.e., such that  $\frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{L}_{x_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = \mathcal{L}_{u_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , respectively.

### Theorem

A necessary condition for the action  $S$  to be minimal is that  $q$  satisfies for all  $t \in [a, b]$  the so-called Euler-Lagrange equations,

$$\forall i = 0..n, \quad \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = 0.$$

### Exercise: proof!

*Remark:* These equations only give a **necessary** condition for the action to be minimal. A trajectory  $q$  satisfying them actually corresponds to a **stationary** point of the action (minimum, maximum or saddle point).

## Euler-Lagrange equations

We assume  $\mathcal{L}$  is differentiable, and that its partial derivatives  $\mathcal{L}_{x_i} = \frac{\partial \mathcal{L}}{\partial x_i}$  and  $\mathcal{L}_{u_i} = \frac{\partial \mathcal{L}}{\partial u_i}$  are continuous. We take the (misused) notation  $\frac{\partial \mathcal{L}}{\partial q_i}$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i}$  to represent the two partial derivatives of  $\mathcal{L}$  evaluated at  $(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , i.e., such that  $\frac{\partial \mathcal{L}}{\partial q_i} = \mathcal{L}_{x_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$  and  $\frac{\partial \mathcal{L}}{\partial \dot{q}_i} = \mathcal{L}_{u_i}(\mathbf{q}(t), \dot{\mathbf{q}}(t), t)$ , respectively.

### Theorem

A necessary condition for the action  $S$  to be minimal is that  $q$  satisfies for all  $t \in [a, b]$  the so-called Euler-Lagrange equations,

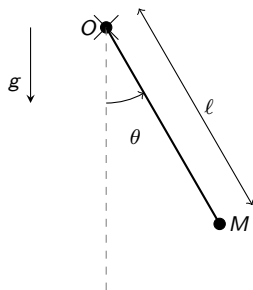
$$\forall i = 0..n, \quad \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = 0.$$

### Exercise: proof!

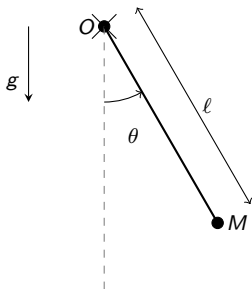
*Remark:* These equations only give a **necessary** condition for the action to be minimal. A trajectory  $q$  satisfying them actually corresponds to a **stationary** point of the action (minimum, maximum or saddle point).

*Remark:* This is a **variational** principle (minimum condition on a functional). It can be applied beyond dynamics, for instance to find object shapes with minimal weight, or to compute shapes at static equilibrium (see Class 5).

## Back to the simple pendulum



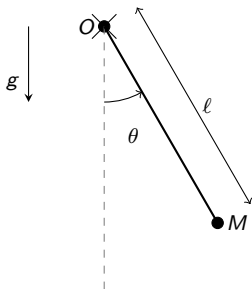
## Back to the simple pendulum



## Exercise

- Write the equations of motion of the simple pendulum...

## Back to the simple pendulum



## Exercise

- Write the equations of motion of the simple pendulum...
- ...using the Euler-Lagrange formalism

## White board 1/6

## White board 2/6



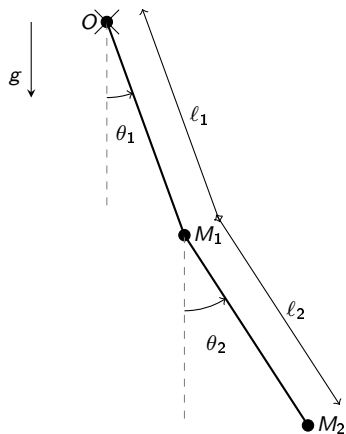
## White board 3/6

## White board 4/6

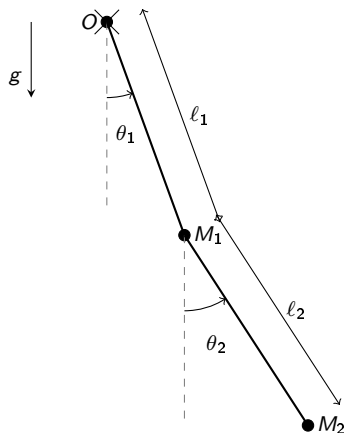
## White board 5/6

## White board 6/6

## The double pendulum



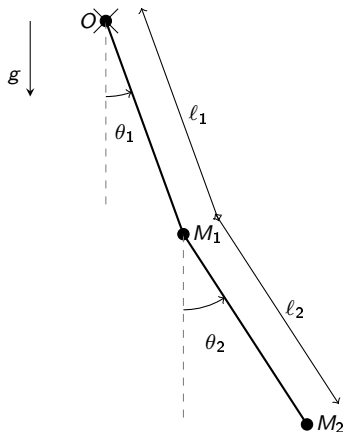
## The double pendulum



## Exercise

- Write the equations of motion of the double pendulum...

## The double pendulum



## Exercise

- Write the equations of motion of the double pendulum...
- ...using the Euler-Lagrange formalism

## White board 1/6



## White board 2/6

## White board 3/6

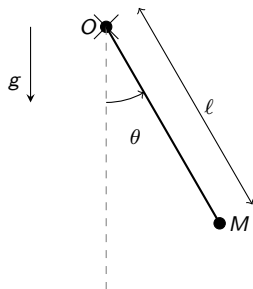
## White board 4/6

## White board 5/6

## White board 6/6

## Part II: Finite differences

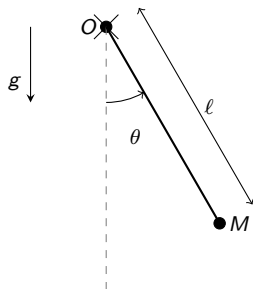
## Back to the simple pendulum



## Computing the dynamics

$$\ddot{\theta} + \frac{g}{\ell} \sin \theta = 0 \quad \text{with } \theta(0) = \theta_0 \text{ and } \dot{\theta}(0) = \lambda_0$$

## Back to the simple pendulum



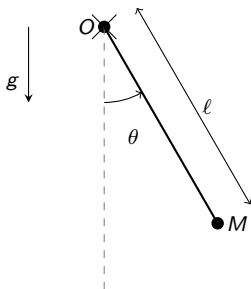
## Computing the dynamics

$$\ddot{\theta} + \frac{g}{\ell} \sin \theta = 0 \quad \text{with } \theta(0) = \theta_0 \text{ and } \dot{\theta}(0) = \lambda_0$$

- Nonlinear equation, no explicit solution



## Back to the simple pendulum



## Computing the dynamics

$$\ddot{\theta} + \frac{g}{\ell} \sin \theta = 0 \quad \text{with } \theta(0) = \theta_0 \text{ and } \dot{\theta}(0) = \lambda_0$$

- Nonlinear equation, no explicit solution
- → Recourse to **numerical integration**

## Cauchy problem

We consider the following first-order differential equation with initial value,

$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

## Cauchy problem

We consider the following first-order differential equation with initial value,

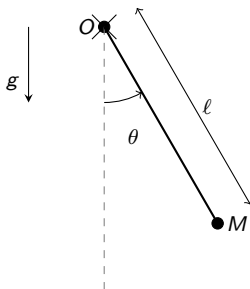
$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

where  $f$  is a continuous function from  $\mathbb{R}^p \times \mathbb{R}$  to  $\mathbb{R}^p$ ,  $p \geq 1$ . Such a problem belongs to the class of **Cauchy** problems. The **unknown** of the problem is the function  $x$  from  $\mathbb{R}$  to  $\mathbb{R}^p$ .

## Cauchy problem

## Exercise

Show that the simple pendulum equation of motion enters the formalism above.



## Equation of motion (reminder)

$$\ddot{\theta} + \frac{g}{\ell} \sin \theta = 0 \quad \text{with } \theta(0) = \theta_0 \text{ and } \dot{\theta}(0) = \lambda_0$$

# Existence and uniqueness of a solution

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall(x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall(x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

### Exercise

- Show that a Lipschitz continuous function is necessarily continuous.
- Is the opposite true?

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall(x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$



## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**.

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**.

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**. If  $K = 0$ , the function is necessarily **constant**.

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**.  
If  $K = 0$ , the function is necessarily **constant**.
- A Lipschitz continuous function is necessarily **continuous** (opposite is not true).

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**. If  $K = 0$ , the function is necessarily **constant**.
- A Lipschitz continuous function is necessarily **continuous** (opposite is not true).
- A differentiable function  $g$  is Lipschitz continuous on  $E$  if and only if its first derivative is **bounded** (and  $\sup_E \|g'(x)\|$  is the smallest Lipschitz constant).

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**. If  $K = 0$ , the function is necessarily **constant**.
- A Lipschitz continuous function is necessarily **continuous** (opposite is not true).
- A differentiable function  $g$  is Lipschitz continuous on  $E$  if and only if its first derivative is **bounded** (and  $\sup_E \|g'(x)\|$  is the smallest Lipschitz constant).
- A function  $g$  is called **locally Lipschitz continuous** if for every  $x$  in  $\mathbb{R}^p$  there exists a neighborhood  $U$  of  $x$  such that  $g$  restricted to  $U$  is Lipschitz continuous.

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**. If  $K = 0$ , the function is necessarily **constant**.
- A Lipschitz continuous function is necessarily **continuous** (opposite is not true).
- A differentiable function  $g$  is Lipschitz continuous on  $E$  if and only if its first derivative is **bounded** (and  $\sup_E \|g'(x)\|$  is the smallest Lipschitz constant).
- A function  $g$  is called **locally Lipschitz continuous** if for every  $x$  in  $\mathbb{R}^p$  there exists a neighborhood  $U$  of  $x$  such that  $g$  restricted to  $U$  is Lipschitz continuous.

### Cauchy-Lipschitz Theorem

Recall the Cauchy problem, with  $f$  continuous

$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

## Existence and uniqueness of a solution

### Definition

A function  $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$  is **Lipschitz continuous** if there exists a constant  $K \geq 0$  such that

$$\forall (x, y) \in \mathbb{R}^p, \|g(y) - g(x)\| \leq K \|y - x\|.$$

*Remark:*

- $K$  is called the **Lipschitz constant**. If  $K < 1$ , the function is said to be a **contraction**. If  $K = 0$ , the function is necessarily **constant**.
- A Lipschitz continuous function is necessarily **continuous** (opposite is not true).
- A differentiable function  $g$  is Lipschitz continuous on  $E$  if and only if its first derivative is **bounded** (and  $\sup_E \|g'(x)\|$  is the smallest Lipschitz constant).
- A function  $g$  is called **locally Lipschitz continuous** if for every  $x$  in  $\mathbb{R}^p$  there exists a neighborhood  $U$  of  $x$  such that  $g$  restricted to  $U$  is Lipschitz continuous.

### Cauchy-Lipschitz Theorem

Recall the Cauchy problem, with  $f$  continuous

$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

If  $f$  is locally Lipschitz continuous with respect to its first (state) variable  $x(t)$ , then there **exists** a **unique** solution  $\bar{x}$  to the Cauchy problem, and  $\bar{x}$  is  $C^1$  continuous.



## Numerical scheme

Recall that the Cauchy problem 
$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

admits the unique solution  $\bar{x}(t)$  for  $t \in [a, b]$  (under some regularity assumptions on  $f$ ). In general, this solution has no explicit form, so we search for a **numerical approximation** of  $\bar{x}(t)$ .

## Numerical scheme

Recall that the Cauchy problem 
$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

admits the unique solution  $\bar{x}(t)$  for  $t \in [a, b]$  (under some regularity assumptions on  $f$ ). In general, this solution has no explicit form, so we search for a **numerical approximation** of  $\bar{x}(t)$ .

→ We subdivide the interval  $[a, b]$  in  $N$  segments of length  $h$  ( $h =$  the **timestep**), and we aim at constructing a sequence of points  $x_0, x_1, \dots, x_k, \dots, x_N$  at discrete times  $t_0 + h, t_0 + 2h, \dots, t_0 + k h, \dots, t_0 + n h$ , which approximate well the exact solution.

## Numerical scheme

Recall that the Cauchy problem  $\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$

admits the unique solution  $\bar{x}(t)$  for  $t \in [a, b]$  (under some regularity assumptions on  $f$ ). In general, this solution has no explicit form, so we search for a **numerical approximation** of  $\bar{x}(t)$ .

→ We subdivide the interval  $[a, b]$  in  $N$  segments of length  $h$  ( $h =$  the **timestep**), and we aim at constructing a sequence of points  $x_0, x_1, \dots, x_k, \dots, x_N$  at discrete times  $t_0 + h, t_0 + 2h, \dots, t_0 + kh, \dots, t_0 + nh$ , which approximate well the exact solution.

### Definition

The numerical scheme is said to be **convergent** if

$$\sup_k \|e_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0,$$

where  $e_k = \bar{x}_k - x_k$  (with  $\bar{x}_k = \bar{x}(t_k)$ ) is the **convergence error** at  $t_k$ .

## Numerical scheme

Recall that the Cauchy problem 
$$\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$$

admits the unique solution  $\bar{x}(t)$  for  $t \in [a, b]$  (under some regularity assumptions on  $f$ ). In general, this solution has no explicit form, so we search for a **numerical approximation** of  $\bar{x}(t)$ .

→ We subdivide the interval  $[a, b]$  in  $N$  segments of length  $h$  ( $h =$  the **timestep**), and we aim at constructing a sequence of points  $x_0, x_1, \dots, x_k, \dots, x_N$  at discrete times  $t_0 + h, t_0 + 2h, \dots, t_0 + kh, \dots, t_0 + nh$ , which approximate well the exact solution.

### Definition

The numerical scheme is said to be **convergent** if

$$\sup_k \|e_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0,$$

where  $e_k = \bar{x}_k - x_k$  (with  $\bar{x}_k = \bar{x}(t_k)$ ) is the **convergence error** at  $t_k$ .

### Definition

The numerical scheme is **convergent of order  $m$**  if there exists a constant  $C \geq 0$  (independent of  $h$ ) such that  $\sup_k \|e_k\| \leq C h^m$ .

## Numerical scheme

Recall that the Cauchy problem  $\begin{cases} \dot{x} = f(x(t), t) & t \in [a, b] \\ x(a) = x_0 \end{cases}$

admits the unique solution  $\bar{x}(t)$  for  $t \in [a, b]$  (under some regularity assumptions on  $f$ ). In general, this solution has no explicit form, so we search for a **numerical approximation** of  $\bar{x}(t)$ .

→ We subdivide the interval  $[a, b]$  in  $N$  segments of length  $h$  ( $h =$  the **timestep**), and we aim at constructing a sequence of points  $x_0, x_1, \dots, x_k, \dots, x_N$  at discrete times  $t_0 + h, t_0 + 2h, \dots, t_0 + kh, \dots, t_0 + nh$ , which approximate well the exact solution.

### Definition

The numerical scheme is said to be **convergent** if

$$\sup_k \|e_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0,$$

where  $e_k = \bar{x}_k - x_k$  (with  $\bar{x}_k = \bar{x}(t_k)$ ) is the **convergence error** at  $t_k$ .

### Definition

The numerical scheme is **convergent of order  $m$**  if there exists a constant  $C \geq 0$  (independent of  $h$ ) such that  $\sup_k \|e_k\| \leq C h^m$ .

→ The greater  $m$ , the faster the method converges to the exact solution.

## Single-step numerical scheme

### Definition

A **single-step numerical scheme** is defined as the recurrence

$$x_0 \text{ given} \quad x_{k+1} = x_k + h\Phi(x_k, h, t_k)$$

where  $\Phi$  defines the type of numerical scheme used.

## Single-step numerical scheme

### Definition

A **single-step numerical scheme** is defined as the recurrence

$$x_0 \text{ given} \quad x_{k+1} = x_k + h\Phi(x_k, h, t_k)$$

where  $\Phi$  defines the type of numerical scheme used.

### Examples

- **Explicit Euler:**  $x_{k+1} = x_k + hf(x_k, t_k)$

## Single-step numerical scheme

### Definition

A **single-step numerical scheme** is defined as the recurrence

$$x_0 \text{ given} \quad x_{k+1} = x_k + h\Phi(x_k, h, t_k)$$

where  $\Phi$  defines the type of numerical scheme used.

### Examples

- **Explicit Euler:**  $x_{k+1} = x_k + hf(x_k, t_k)$   
 $\rightarrow \Phi(x_k, h, t_k) = f(x_k, t_k)$



## Single-step numerical scheme

### Definition

A **single-step numerical scheme** is defined as the recurrence

$$x_0 \text{ given} \quad x_{k+1} = x_k + h\Phi(x_k, h, t_k)$$

where  $\Phi$  defines the type of numerical scheme used.

### Examples

- **Explicit Euler:**  $x_{k+1} = x_k + hf(x_k, t_k)$   
 $\rightarrow \Phi(x_k, h, t_k) = f(x_k, t_k)$
- **Implicit Euler:**  $x_{k+1} = x_k + hf(x_{k+1}, t_{k+1})$

## Single-step numerical scheme

### Definition

A **single-step numerical scheme** is defined as the recurrence

$$x_0 \text{ given} \quad x_{k+1} = x_k + h\Phi(x_k, h, t_k)$$

where  $\Phi$  defines the type of numerical scheme used.

### Examples

- **Explicit Euler:**  $x_{k+1} = x_k + hf(x_k, t_k)$   
 $\rightarrow \Phi(x_k, h, t_k) = f(x_k, t_k)$
- **Implicit Euler:**  $x_{k+1} = x_k + hf(x_{k+1}, t_{k+1})$   
 $\rightarrow \Phi$  defined **implicitly** (is well-defined if  $f_x(x, t)y \cdot y \leq 0 \quad \forall x, y \in \mathbb{R}^p, \forall t \in [a, b]$ )

## Single-step numerical scheme

### Definition

A **single-step numerical scheme** is defined as the recurrence

$$x_0 \text{ given} \quad x_{k+1} = x_k + h\Phi(x_k, h, t_k)$$

where  $\Phi$  defines the type of numerical scheme used.

### Examples

- **Explicit Euler:**  $x_{k+1} = x_k + hf(x_k, t_k)$   
 $\rightarrow \Phi(x_k, h, t_k) = f(x_k, t_k)$
- **Implicit Euler:**  $x_{k+1} = x_k + hf(x_{k+1}, t_{k+1})$   
 $\rightarrow \Phi$  defined **implicitly** (is well-defined if  $f_x(x, t)y \cdot y \leq 0 \quad \forall x, y \in \mathbb{R}^p, \forall t \in [a, b]$ )

*Remark:* Convergence is generally hard to prove directly. Instead, one usually prefers to prove **consistency + stability w.r.t. errors** instead.

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .  
Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .  
Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

A single-step numerical scheme is said to be **consistent** if

$$\sup_k \|R_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0.$$

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .  
Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

A single-step numerical scheme is said to be **consistent** if

$$\sup_k \|R_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0.$$

The scheme is **consistent of order  $m$**  if there exists a constant  $C \leq 0$  (independent of  $h$ ) such that

$$\|R_k\| \leq C h^m \quad \forall k = 0, \dots, N$$



## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .

Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

A single-step numerical scheme is said to be **consistent** if

$$\sup_k \|R_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0.$$

The scheme is **consistent of order  $m$**  if there exists a constant  $C \geq 0$  (independent of  $h$ ) such that

$$\|R_k\| \leq C h^m \quad \forall k = 0, \dots, N$$

### Remark:

- “Consistent” means that the scheme “converges” to the original Cauchy problem when  $h \rightarrow 0$ . If the scheme is not consistent, it means that we are trying to find an approximate solution to another problem!

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .  
Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

A single-step numerical scheme is said to be **consistent** if

$$\sup_k \|R_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0.$$

The scheme is **consistent of order  $m$**  if there exists a constant  $C \geq 0$  (independent of  $h$ ) such that

$$\|R_k\| \leq C h^m \quad \forall k = 0, \dots, N$$

### Remark:

- “Consistent” means that the scheme “converges” to the original Cauchy problem when  $h \rightarrow 0$ . If the scheme is not consistent, it means that we are trying to find an approximate solution to another problem!  
→ Consistency is **necessary** to have convergence (but not sufficient...).

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .

Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

A single-step numerical scheme is said to be **consistent** if

$$\sup_k \|R_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0.$$

The scheme is **consistent of order  $m$**  if there exists a constant  $C \geq 0$  (independent of  $h$ ) such that

$$\|R_k\| \leq C h^m \quad \forall k = 0, \dots, N$$

### Remark:

- “Consistent” means that the scheme “converges” to the original Cauchy problem when  $h \rightarrow 0$ . If the scheme is not consistent, it means that we are trying to find an approximate solution to another problem!  
→ Consistency is **necessary** to have convergence (but not sufficient...).
- The order of convergence is directly related to the order of consistency of a numerical scheme.

## Consistency of a numerical scheme

Recall that  $x_{k+1} = x_k + h\Phi(x_k, h, t_k)$ , i.e.  $\frac{x_{k+1} - x_k}{h} - \Phi(x_k, t_k, h) = 0$ .

### Definition

We define  $R_k = \frac{\bar{x}_{k+1} - \bar{x}_k}{h} - \Phi(\bar{x}_k, t_k, h)$  as the **consistency error** of the scheme at  $t_k$ .

Reminder:  $\bar{x}_k = \bar{x}(t_k)$  where  $\bar{x}(t)$  is the exact solution.

A single-step numerical scheme is said to be **consistent** if

$$\sup_k \|R_k\| \rightarrow 0 \quad \text{when } h \rightarrow 0.$$

The scheme is **consistent of order  $m$**  if there exists a constant  $C \geq 0$  (independent of  $h$ ) such that

$$\|R_k\| \leq C h^m \quad \forall k = 0, \dots, N$$

### Remark:

- “Consistent” means that the scheme “converges” to the original Cauchy problem when  $h \rightarrow 0$ . If the scheme is not consistent, it means that we are trying to find an approximate solution to another problem!  
→ Consistency is **necessary** to have convergence (but not sufficient...).
- The order of convergence is directly related to the order of consistency of a numerical scheme.

## Consistency of a numerical scheme

### Exercise

Show that Explicit Euler is a **consistent** scheme of **first** order.

Reminder: Explicit Euler:  $x_{k+1} = x_k + hf(x_k, t_k)$

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment,

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

### Definition

The numerical scheme is **stable with respect to errors** if there exists a constant  $M > 0$  and a constant  $\bar{\varepsilon} > 0$  (both independent of  $h$ ) such that

$$\forall h,$$



## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

### Definition

The numerical scheme is **stable with respect to errors** if there exists a constant  $M > 0$  and a constant  $\bar{\varepsilon} > 0$  (both independent of  $h$ ) such that

$$\forall h, \forall \varepsilon_k < \bar{\varepsilon},$$

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

### Definition

The numerical scheme is **stable with respect to errors** if there exists a constant  $M > 0$  and a constant  $\bar{\varepsilon} > 0$  (both independent of  $h$ ) such that

$$\forall h, \forall \varepsilon_k < \bar{\varepsilon}, \quad \sup_k \|x_k - \tilde{x}_k\| < M \sup_k \|\varepsilon_k\|$$

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

### Definition

The numerical scheme is **stable with respect to errors** if there exists a constant  $M > 0$  and a constant  $\bar{\varepsilon} > 0$  (both independent of  $h$ ) such that

$$\forall h, \forall \varepsilon_k < \bar{\varepsilon}, \quad \sup_k \|x_k - \tilde{x}_k\| < M \sup_k \|\varepsilon_k\| \leq M \bar{\varepsilon}$$

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

### Definition

The numerical scheme is **stable with respect to errors** if there exists a constant  $M > 0$  and a constant  $\bar{\varepsilon} > 0$  (both independent of  $h$ ) such that

$$\forall h, \forall \varepsilon_k < \bar{\varepsilon}, \quad \sup_k \|x_k - \tilde{x}_k\| < M \sup_k \|\varepsilon_k\| \leq M \bar{\varepsilon}$$

*Remark:*

- Means that a perturbation on the initial condition and on the increment  $\Phi$  only yields a bounded perturbation on the numerical scheme, and so the scheme **does not amplify** errors.

## Stability with respect to errors

We consider that at each time step, the computation of  $x_k$  may be altered by a **perturbation**  $\varepsilon_k$  on the increment, i.e., instead of computing the exact sequence  $x_k$  as previously, we compute  $\tilde{x}_k$  such that

$$\tilde{x}_0 = x_0 + \varepsilon_0 \quad \tilde{x}_{k+1} = \tilde{x}_k + h(\Phi(\tilde{x}_k, t_k, h) + \varepsilon_{k+1}).$$

### Definition

The numerical scheme is **stable with respect to errors** if there exists a constant  $M > 0$  and a constant  $\bar{\varepsilon} > 0$  (both independent of  $h$ ) such that

$$\forall h, \forall \varepsilon_k < \bar{\varepsilon}, \quad \sup_k \|x_k - \tilde{x}_k\| < M \sup_k \|\varepsilon_k\| \leq M \bar{\varepsilon}$$

### Remark:

- Means that a perturbation on the initial condition and on the increment  $\Phi$  only yields a bounded perturbation on the numerical scheme, and so the scheme **does not amplify** errors.
- **Easy criterion:** stability w.r.t. errors is guaranteed when  $\Phi$  satisfies some regularity properties: typically, when  $\Phi$  is **Lipschitz continuous** w.r.t.  $x$ .

# Convergence

## Theorem

*Convergence = consistency + stability*

# Convergence

## Theorem

*Convergence = consistency + stability*

## Exercise

Show that Explicit Euler is a **convergent** scheme.

Other type of stability: Stability of the numerical solution (A-stability)



## Other type of stability: Stability of the numerical solution (A-stability)

## Exercise

Derive the Explicit Euler scheme for the simple scalar Cauchy problem

$$x(a) = x_0 \quad \dot{x}(t) = -\lambda x(t) \quad \forall t \in [a, b] \quad \text{with } \lambda > 0.$$

## Other type of stability: Stability of the numerical solution (A-stability)

## Exercise

Derive the Explicit Euler scheme for the simple scalar Cauchy problem

$$x(a) = x_0 \quad \dot{x}(t) = -\lambda x(t) \quad \forall t \in [a, b] \quad \text{with } \lambda > 0.$$

How does the approximate solution behave for a fixed timestep  $h$  when the time interval grows ( $b \rightarrow +\infty$ )?

## Other type of stability: Stability of the numerical solution (A-stability)

## Exercise

Derive the **Explicit Euler** scheme for the simple scalar Cauchy problem

$$x(a) = x_0 \quad \dot{x}(t) = -\lambda x(t) \quad \forall t \in [a, b] \quad \text{with } \lambda > 0.$$

How does the approximate solution behave for a fixed timestep  $h$  when the time interval grows ( $b \rightarrow +\infty$ )?

## Definition

A numerical scheme is said to be **stable** if there exists  $h^* > 0$  and  $R \geq 0$  such that

$$\|x_k\| \leq R \quad \forall k = 0, \dots, N \quad \text{and } \forall h \in [0, h^*[.$$

## Other type of stability: Stability of the numerical solution (A-stability)

## Exercise

Derive the **Explicit Euler** scheme for the simple scalar Cauchy problem

$$x(a) = x_0 \quad \dot{x}(t) = -\lambda x(t) \quad \forall t \in [a, b] \quad \text{with } \lambda > 0.$$

How does the approximate solution behave for a fixed timestep  $h$  when the time interval grows ( $b \rightarrow +\infty$ )?

## Definition

A numerical scheme is said to be **stable** if there exists  $h^* > 0$  and  $R \geq 0$  such that

$$\|x_k\| \leq R \quad \forall k = 0, \dots, N \quad \text{and } \forall h \in [0, h^*[.$$

Moreover, the scheme is said to be **unconditionally stable** if  $h^* = +\infty$ .

## Other type of stability: Stability of the numerical solution (A-stability)

### Exercise

Derive the **Explicit Euler** scheme for the simple scalar Cauchy problem

$$x(a) = x_0 \quad \dot{x}(t) = -\lambda x(t) \quad \forall t \in [a, b] \quad \text{with } \lambda > 0.$$

How does the approximate solution behave for a fixed timestep  $h$  when the time interval grows ( $b \rightarrow +\infty$ )?

### Definition

A numerical scheme is said to be **stable** if there exists  $h^* > 0$  and  $R \geq 0$  such that

$$\|x_k\| \leq R \quad \forall k = 0, \dots, N \quad \text{and } \forall h \in [0, h^*].$$

Moreover, the scheme is said to be **unconditionally stable** if  $h^* = +\infty$ .

*Remark:* **Stability w.r.t. errors** and **stability** (as defined above) are different notions of stability. A scheme can be stable w.r.t. errors but unstable (in the sense above) for certain timesteps.

## Other type of stability: Stability of the numerical solution (A-stability)

### Exercise

Derive the **Explicit Euler** scheme for the simple scalar Cauchy problem

$$x(a) = x_0 \quad \dot{x}(t) = -\lambda x(t) \quad \forall t \in [a, b] \quad \text{with } \lambda > 0.$$

How does the approximate solution behave for a fixed timestep  $h$  when the time interval grows ( $b \rightarrow +\infty$ )?

### Definition

A numerical scheme is said to be **stable** if there exists  $h^* > 0$  and  $R \geq 0$  such that

$$\|x_k\| \leq R \quad \forall k = 0, \dots, N \quad \text{and } \forall h \in [0, h^*].$$

Moreover, the scheme is said to be **unconditionally stable** if  $h^* = +\infty$ .

*Remark:* **Stability w.r.t. errors** and **stability** (as defined above) are different notions of stability. A scheme can be stable w.r.t. errors but unstable (in the sense above) for certain timesteps.

→ To keep in mind: stability w.r.t. errors is useful for proving **convergence**. Stability as defined above is useful when considering integration of systems on **moderate or large time intervals**, or when using a **large timestep** (which is often useful in practice!).

## Explicit vs. Implicit Euler

## Exercise 1

Consider the linearized pendulum problem (valid for small angle  $\theta$ ),

$$\ddot{\theta} + \frac{g}{l}\theta = 0 \quad \text{with } \theta(0) = \theta_0 \text{ and } \dot{\theta}(0) = \lambda_0,$$

and express the condition on the time step  $h$  for Explicit Euler to be stable.

## Explicit vs. Implicit Euler

### Exercise 2

Same question for Implicit Euler.



## Explicit vs. Implicit Euler

In the general case:

### Stability of Explicit Euler

Explicit Euler is **conditionally** stable.

## Explicit vs. Implicit Euler

In the general case:

### Stability of Explicit Euler

Explicit Euler is **conditionally** stable.

### Stability of Implicit Euler

Implicit Euler is **unconditionally** stable.

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence = consistence + stability w.r.t. errors**

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.
- Take home message 3: **stability** of the numerical solution is very important in practice (see python practical this afternoon).

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.
- Take home message 3: **stability** of the numerical solution is very important in practice (see python practical this afternoon).

### Going further

- Many other important single-step schemes: **Runge-Kutta** schemes (better order of convergence).

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.
- Take home message 3: **stability** of the numerical solution is very important in practice (see python practical this afternoon).

### Going further

- Many other important single-step schemes: **Runge-Kutta** schemes (better order of convergence).
- Multi-step schemes to yet improve accuracy while gaining efficiency...



## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.
- Take home message 3: **stability** of the numerical solution is very important in practice (see python practical this afternoon).

### Going further

- Many other important single-step schemes: **Runge-Kutta** schemes (better order of convergence).
- Multi-step schemes to yet improve accuracy while gaining efficiency...
- Construction of “symplectic” schemes (mix of explicit/implicit): guarantee of **energy conservation** during large time intervals (see python practical).

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.
- Take home message 3: **stability** of the numerical solution is very important in practice (see python practical this afternoon).

### Going further

- Many other important single-step schemes: **Runge-Kutta** schemes (better order of convergence).
- Multi-step schemes to yet improve accuracy while gaining efficiency...
- Construction of “symplectic” schemes (mix of explicit/implicit): guarantee of **energy conservation** during large time intervals (see python practical).
- Solving of **linear** or **nonlinear** systems stemming from **implicit** or **semi-implicit** finite differences (see python practical)

Linear system solves: LU, Cholesky, iterative methods... Nonlinear system solves: Newton, Broyden,...

## Conclusion

### In a nutshell

- A short tour on the important notions and properties of finite difference schemes
- Take home message 1: **convergence** = **consistence** + **stability w.r.t. errors**
- Take home message 2: the **order** of consistency gives the **accuracy** of the scheme.
- Take home message 3: **stability** of the numerical solution is very important in practice (see python practical this afternoon).

### Going further

- Many other important single-step schemes: **Runge-Kutta** schemes (better order of convergence).
- Multi-step schemes to yet improve accuracy while gaining efficiency...
- Construction of “symplectic” schemes (mix of explicit/implicit): guarantee of **energy conservation** during large time intervals (see python practical).
- Solving of **linear** or **nonlinear** systems stemming from **implicit** or **semi-implicit** finite differences (see python practical)

Linear system solves: LU, Cholesky, iterative methods... Nonlinear system solves: Newton, Broyden,...

- References: G. Allaire, Analyse numérique et optimisation (“Numerical analysis and optimization”), E. Hairer et al., Geometric numerical integration.