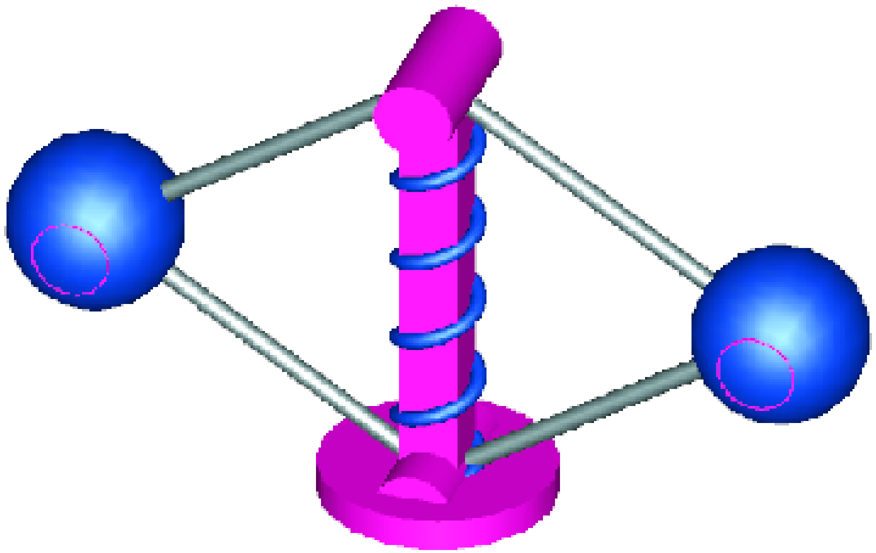
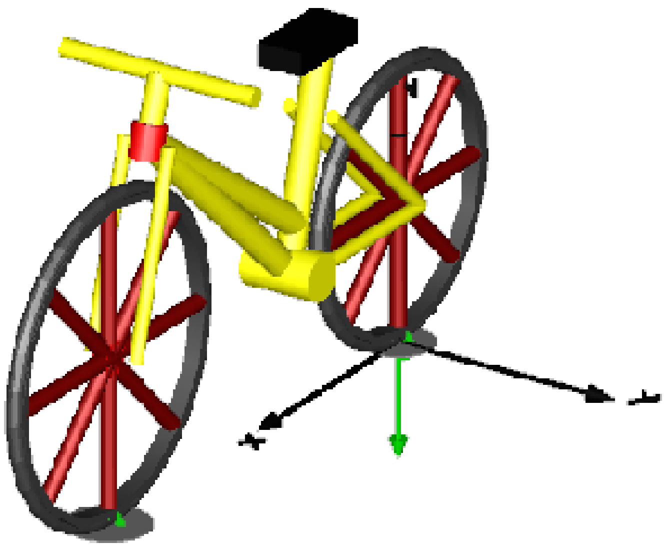
Virtual Physics
Equation-Based Modeling

TUM, November 25, 2014

3D Mechanics, Part I

Dr. Dirk Zimmer

German Aerospace Center (DLR), Robotics and Mechatronics Centre

3D Mechanics

Robotics and Mechatronics Centre

In this lecture, we look at the modeling of 3D mechanical systems.

- 3D mechanical models look superficially just like planar mechanical models. There are additional types of joints, but other than that, there seem to be few surprises.
- Yet, the seemingly similar appearance is deceiving. There are a substantial number of complications that the modeler has to cope with when dealing with 3D mechanics. These are the subject of this lecture.

3D Mechanics

Essentially, there are 3 major difficulties we have to cope with:

1. There are multiple ways to express the orientation of a body in three dimensional space.

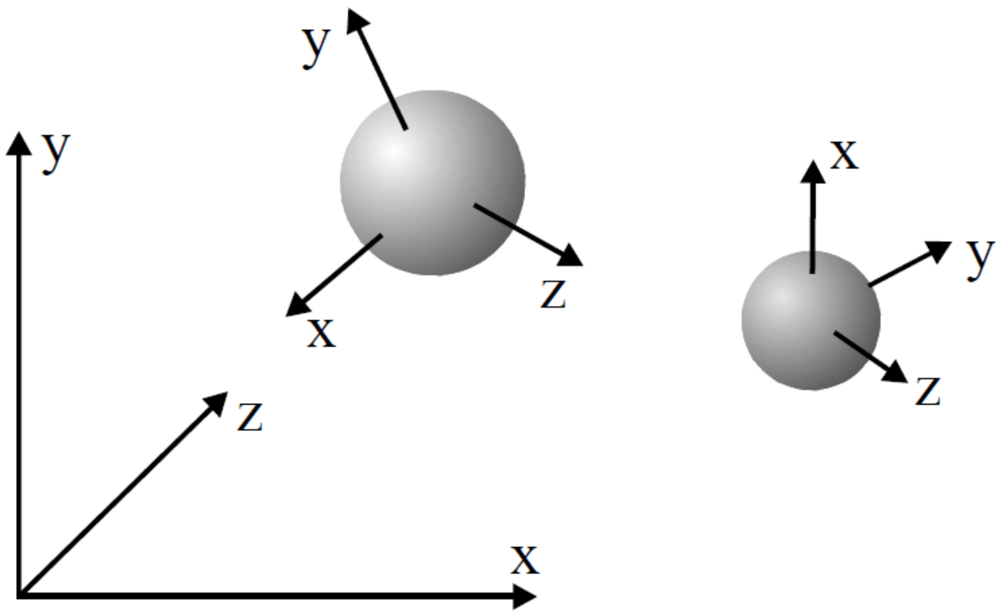
2. In planar mechanics, all potential variables could be expressed in one common coordinate system: The inertial system. In 3D-mechanics, such an approach is unfeasible.

3. The set of connector variables contains a redundant set of variables. This causes severe problems for the formulation of kinematic loops.

Orientation

Robotics and Mechatronics Centre

There are 4 major variants to express the orientation of an object in 3D

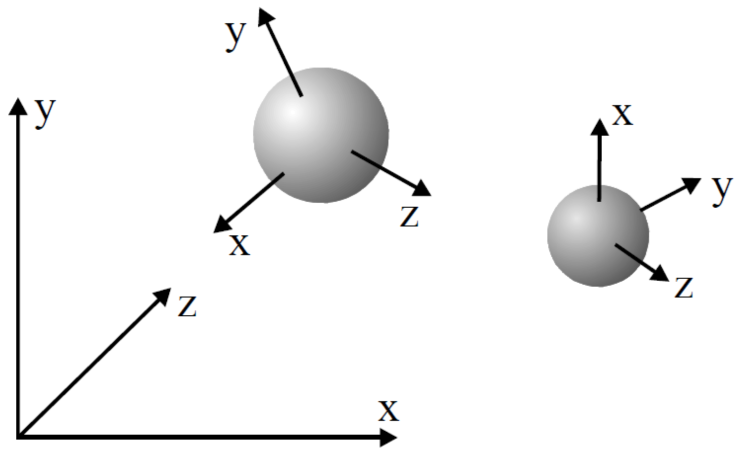


- The rotation matrix
- Planar rotation
- Cardan angles
- Quaternions

Orientation Matrix R

The rotation matrix **R**

- The orientation of an object is completely defined by the coordinate vectors of its body system.
- The relative orientation between two objects can then be described by a orthonormal matrix: the rotation matrix **R**.
- Given the rotational matrix, we can easily transform vectors between different coordinate systems, e. g., $\mathbf{R}\boldsymbol{\omega}_0 = \boldsymbol{\omega}_{body}$



$$\mathbf{R}^{-1} = \mathbf{R}^T$$

$$||\mathbf{R}||_2 = 1$$

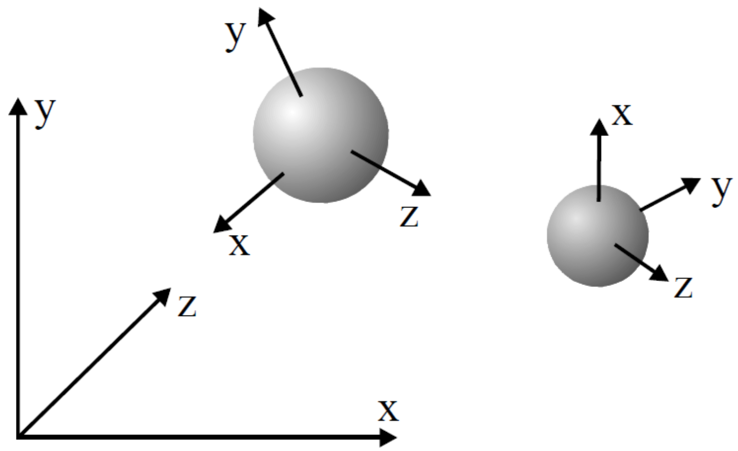
Orientation Matrix R

The rotation matrix **R**

- The rotational matrix **R** is highly redundant.

- Each row vector and each column vector of **R** is of length 1, hence there are 6 constraint equations connecting the 9 matrix elements.

- As expected, there are only 3 degrees of freedom, describing the relative rotation of one coordinate system to another.



$$\mathbf{R}^{-1} = \mathbf{R}^T$$

$$||\mathbf{R}||_2 = 1$$

The Cardan Angles (φ_x, φ_y, φ_z)

The cardan angles (φ_x, φ_y, φ_z)

- A non-redundant form to describe the orientation are cardan angles.
- This technique decomposes the rotation into three subsequent rotations around predetermined axes.
- In this case:
  first x,
  then y,
  finally z.

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\varphi_x) & sin(\varphi_x) \\ 0 & -sin(\varphi_x) & cos(\varphi_x) \end{pmatrix}$$

$$\mathbf{R}_y = \begin{pmatrix} cos(\varphi_y) & 0 & -sin(\varphi_y) \\ 0 & 1 & 0 \\ \sin(\varphi_y) & 0 & cos(\varphi_y) \end{pmatrix}$$

$$\mathbf{R}_z = \begin{pmatrix} cos(\varphi_z) & sin(\varphi_z) & 0 \\ -sin(\varphi_z) & cos(\varphi_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{R} = \mathbf{R_z} \cdot \mathbf{R_y} \cdot \mathbf{R_x}$$

The Cardan Angles (φ_x, φ_y, φ_z)

Robotics and Mechatronics Centre

The cardan angles (φ_x, φ_y, φ_z)

- Unfortunately, the decomposition into separate yields a singularity at $\varphi_y = 90°$. The other two rotation axes are then aligned and there are infinitely many solutions.
- So cardan angles are only useful, if one can make sure this case won't appear during simulation time.
- The sequence of axis rotation can be chosen arbitrarily. Other sequences are of course possible as well and each valid sequence has a specific point where the systems becomes singular.

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\varphi_x) & sin(\varphi_x) \\ 0 & -sin(\varphi_x) & cos(\varphi_x) \end{pmatrix}$$

$$\mathbf{R}_y = \begin{pmatrix} cos(\varphi_y) & 0 & -sin(\varphi_y) \\ 0 & 1 & 0 \\ \sin(\varphi_y) & 0 & cos(\varphi_y) \end{pmatrix}$$

$$\mathbf{R}_z = \begin{pmatrix} cos(\varphi_z) & sin(\varphi_z) & 0 \\ -sin(\varphi_z) & cos(\varphi_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

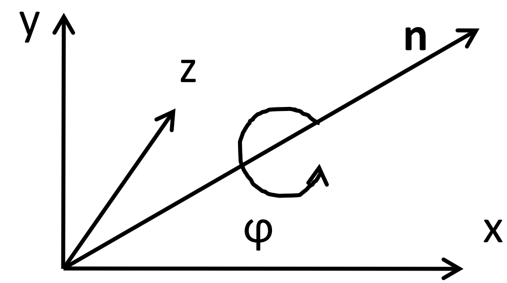
$$\mathbf{R} = \mathbf{R_z} \cdot \mathbf{R_y} \cdot \mathbf{R_x}$$

The Planar Rotation (n, φ)

The planar rotation (**n**, φ):

- Every rotation can be regarded as a planar rotation with the angle φ around a certain axis given by a unit vector **n.**

- We therefore have 4 variables and one constraint equation for the unit vector.



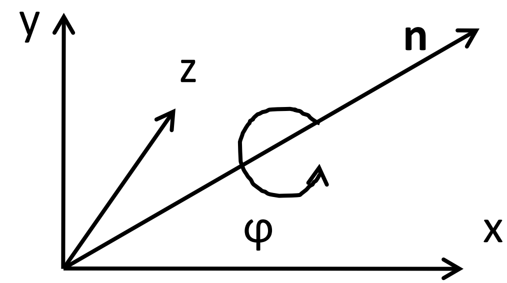
$$\mathbf{R} = \mathbf{n}\mathbf{n}^T + \left(I - \mathbf{n}\mathbf{n}^T\right) cos(\varphi) - \tilde{\mathbf{n}} sin(\varphi)$$

The Planar Rotation (n, φ)

The planar rotation (**n**, φ):

- Unfortunately, also the planar rotation method is not always invertible in a unique fashion. A null rotation does not have a well defined axis of rotation.

- Hence, this method should only be used if the axis of rotation is always known, as in a revolute joint.



$$\mathbf{R} = \mathbf{n}\mathbf{n}^T + \left(I - \mathbf{n}\mathbf{n}^T\right) cos(\varphi) - \tilde{\mathbf{n}} sin(\varphi)$$

Matrix notation of the cross product

$$\mathbf{a} \times \mathbf{b} = \tilde{\mathbf{a}}\mathbf{b}$$

$$\tilde{\mathbf{a}} = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}$$

Quaternions Q

- Quaternions are an extension of complex numbers and offer a robust way to describe any rotation. A quaternion number consists of one real and three imaginary components, denoted by i, j and k.
- The imaginary components can be summarized by a vector **u.**

$$Q = c + ui + vj + wk. = c + \mathbf{u}$$

- The multiplication rules for the imaginary components are as follows:

$$\begin{aligned} ij &= k; \quad & ji &= -k; \quad & i^2 &= -1 \\ jk &= i; \quad & kj &= -i; \quad & j^2 &= -1 \\ ki &= j; \quad & ik &= -j; \quad & k^2 &= -1 \end{aligned}$$

Quaternions Q

- So the product of two quaternions can be written as:

$$QQ' = (c+\mathbf{u})(c'+\mathbf{u}') = (cc' - \mathbf{u}\cdot\mathbf{u}') + (\mathbf{u}\times\mathbf{u}') + c\mathbf{u}' + c'\mathbf{u}$$

- The complement of a quaternion number is defined to be:

$$\bar{Q} = c + \bar{\mathbf{u}} = c - \mathbf{u}$$

- The product of a quaternion number with its complement results in its norm:

$$|Q| = c^2 + |\mathbf{u}|^2$$

- A unit quaternion is a quaternion of norm 1.

$$|Q| = c^2 + |\mathbf{u}|^2 = 1$$

Quaternions Q

- According to the trigonometric Pythagoras…

$$cos(\varphi/2)^2 + sin(\varphi/2)^2 = 1$$

- there is an angle φ for every unit quaternion such that:

$$c = cos(\varphi/2) \text{ and } |\mathbf{u}| = sin(\varphi/2)$$

- It is now evident how a unit quaternion can be used to describe an orientation. The idea is related to the planar rotation. The imaginary component **u** describes the axis, and the length of the axis describes the rotation angle.
- The rotation matrix is then defined by:

$$\mathbf{R} = 2\mathbf{u}\mathbf{u}^T + 2(\tilde{\mathbf{u}} \cdot c) + 2c^2\mathbf{I} - \mathbf{I}$$

Selection of Method

- So which of the four methods shall we apply?
- The answer is: **all of them**

- The rotational matrix is highly redundant but purely linear.

  ➔ It is used in the connector

- Cardan angles can be used for a spherical joint if the motion is limited to non-singular (or ill-conditioned) areas.

  ➔ Free rotational motion, spherical joint

- Planar rotation is used when the rotational axis is known.

  ➔ Revolute Joint

- Quaternions are the methods that avoids any singularity with the slightest degree of redundancy. (But leads to non-linear equations)

  ➔ Free rotational motion, spherical joint

Motion in 3D

Robotics and Mechatronics Centre

- In planar mechanics, ω was the derivative of φ.
- In 3D mechanics, this is not so easy anymore. **ω** represents a vector.
- |**ω**| represents the actual angular velocity
- **ω** /|**ω**| is the unit-vector of the rotation axis.
- **ω** can either be resolved w.r.t. the inertial frame ($\boldsymbol{\omega}_0$) or w.r.t to the body frame ($\boldsymbol{\omega}_{\text{body}}$) .
- The body frame is the coordinate system attached to the body.

Motion in 3D: Rotation Matrix

- The rotational matrix is the one to integrate:

$$\tilde{\omega}_0 \mathbf{R} = \mathbf{R}\tilde{\omega}_{body} = \dot{\mathbf{R}}$$

- This generates 9 differential equations and is thus never used.

Motion in 3D: Cardan Angles

- The rotation matrix **R** results out of a planar rotation:

$$\mathbf{R}\,\omega_0 = \omega_{body} = \mathbf{n} \cdot \dot{\varphi}$$

- 1 differential equations

Motion in 3D: Planar Rotation

- The rotation matrix **R** results out of the cardan angles:

$$\boldsymbol{\omega}_{body} = \dot{\varphi}_z + \mathbf{R}_z \dot{\varphi}_y + \mathbf{R}_z \mathbf{R}_y \dot{\varphi}_x$$

$$\boldsymbol{\omega}_0 = \dot{\varphi}_x + \mathbf{R}_x^T \dot{\varphi}_y + \mathbf{R}_x^T \mathbf{R}_y^T \dot{\varphi}_z$$

- 3 differential equations (non-redundant)

Motion in 3D: Quaternions

Robotics and Mechatronics Centre

- The rotation matrix **R** results out of the quaternion rotation:

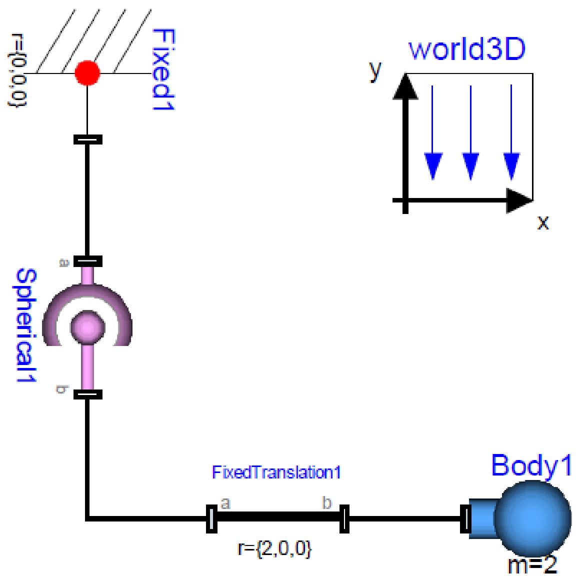
$$\omega_{body} = 2\begin{pmatrix} c & -w & v & u \\ w & c & -u & v \\ -v & u & c & w \end{pmatrix} \cdot \begin{pmatrix} \dot{c} \\ \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix}$$

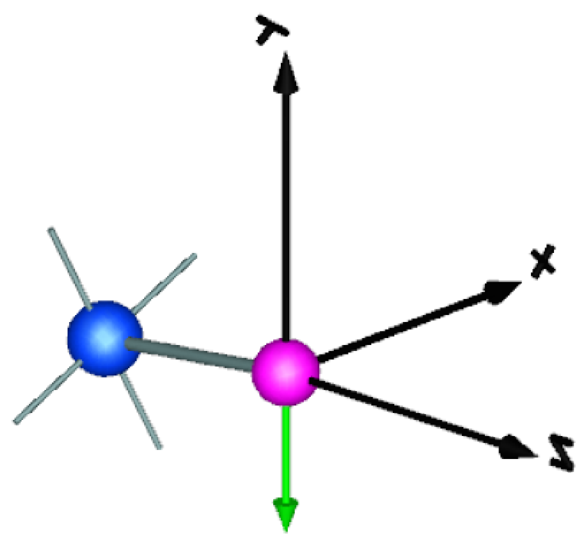
$$\omega_{0} = 2\begin{pmatrix} c & w & -v & u \\ -w & c & u & v \\ v & -u & c & w \end{pmatrix} \cdot \begin{pmatrix} \dot{c} \\ \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix}$$

- 4 differential equations (1 redundant causes dynamic state selection)

Selection of Method

- The choice of a method can severely impact the simulation performance:

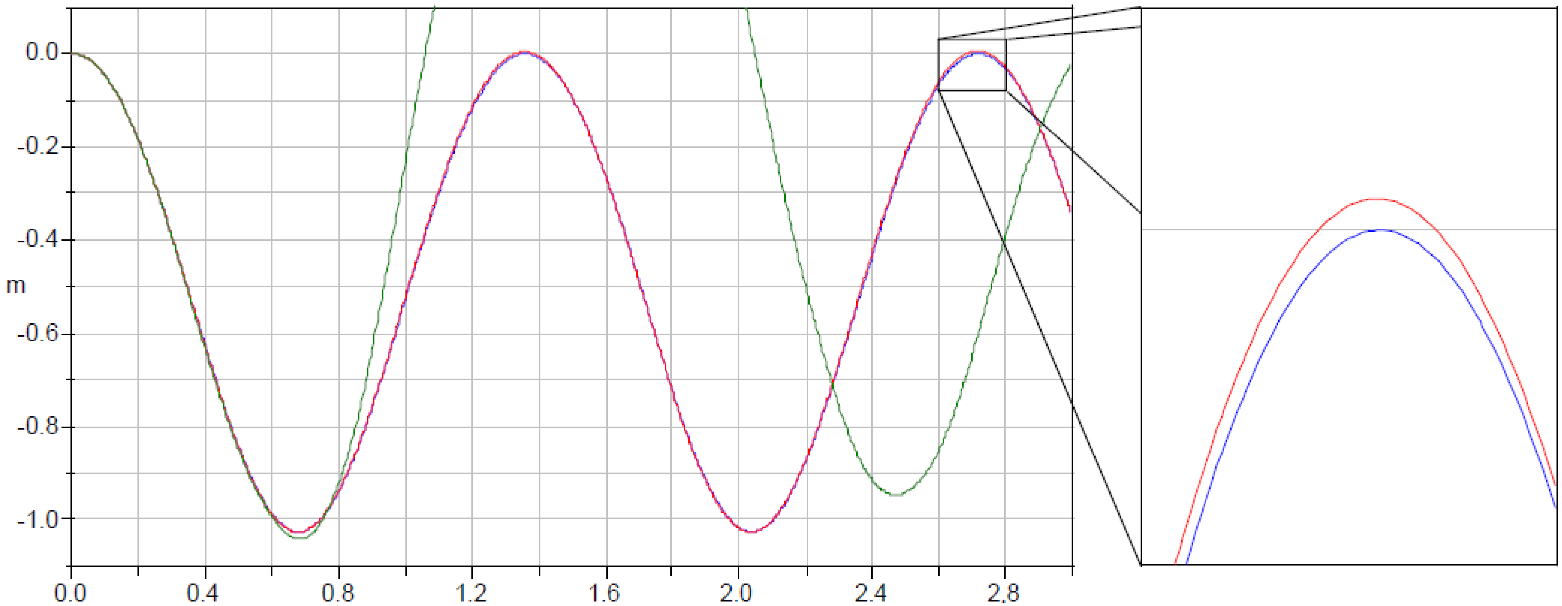




- This experiment was simulated 3 times with a different method for the orientation: 1) well chosen cardan angles, 2) badly chosen cardan angles 3) quaternions

Selection of Method

- The choice of a method can severely impact the simulation performance:



- This experiment was simulated 3 times with a different method for the orientation: 1) well-chosen cardan angles, 2) badly chosen cardan angles 3) quaternions

Selection of Method

- The choice of a method can severely impact the simulation performance:

	good cardan angle seq.		quaternions		bad cardan angle seq.	
tolerance	error	steps	error	steps	error	steps
$1.0 \cdot 10^{-4}$	$4.9 \cdot 10^{-4}$	$2.9 \cdot 10^{3}$	$5.0 \cdot 10^{-3}$	$2.6 \cdot 10^{4}$	$1.8 \cdot 10^{-0}$	$5.4 \cdot 10^{4}$
$1.0 \cdot 10^{-6}$	$9.7 \cdot 10^{-6}$	$6.2 \cdot 10^{3}$	$3.1 \cdot 10^{-4}$	$4.8 \cdot 10^{4}$	$2.9 \cdot 10^{-4}$	$9.5 \cdot 10^{4}$
$1.0 \cdot 10^{-8}$	$1.2 \cdot 10^{-7}$	$1.4 \cdot 10^{4}$	$1.1 \cdot 10^{-5}$	$8.4 \cdot 10^{4}$	$3.5 \cdot 10^{-5}$	$2.0 \cdot 10^{5}$
$1.0 \cdot 10^{-10}$	$1.2 \cdot 10^{-7}$	$2.3 \cdot 10^{4}$	$1.1 \cdot 10^{-6}$	$1.4 \cdot 10^{5}$	$3.0 \cdot 10^{-6}$	$4.4 \cdot 10^{5}$

- The choice drastically impacts the computational efficiency and the precision.

Questions ?