

Introduction to AnyLogic

Motion of Ordinary and Celestial Bodies

Administrative Information

Exam / Klausur

- 120 minutes, 90 points / 100 points
- 20/30 points for questions on the Semester Assignment
→ Without it, you'll need 2/3 of the remaining points just to pass!
- You already know all of the exam questions!
- If you want a Certificate of Attendance ("Schein"), pass the exam
- Registration online
- 5 CP for Bachelor students
6 CP for Master students

Course material account: "ItS", password: "*****"

AnyLogic by XJ Technologies

The simulation tool AnyLogic is available for

- Windows
- Linux
- Mac OS

Using AnyLogic

- Install it on your Computer (instructions are on our website)

Note: AnyLogic is way too slow on Netbooks

AnyLogic by XJ Technologies

Some Features

- Graphical modelling with only small Java code customizations
- Provides code completion (`<ctrl>+<space>`) and refactoring
- Graphical analysis of dynamic processes and simulation results
- Ability to export simulations as Java applets
- Supports multiple simulation paradigms, we'll use
 - Continuous simulation (system dynamics)
 - Discrete event-based simulation
- Extensive help system – **Use that before asking us!**

Today's Tasks

Get familiar with the basics of AnyLogic

Develop a mathematical model for the motion of the Earth around the Sun

Develop a simulation model from this in AnyLogic

The AnyLogic Window

The screenshot displays the AnyLogic University [EDUCATIONAL USE ONLY] interface. The main workspace shows a simulation model for a skydiver, featuring a flowchart with blocks for 'MyPosition', 'flow', 'TestVelocity', 'windResistance', 'Gravity', and 'b'. Two line graphs are visible: 'Velocity v' (blue line) and 'Position p' (orange line). The 'Properties' panel on the right shows the 'Main - Agent Type' configuration, including parameters and actions. The 'Problems' panel at the bottom lists 5 errors related to variable resolution and unused blocks.

Problems

5 error(s)

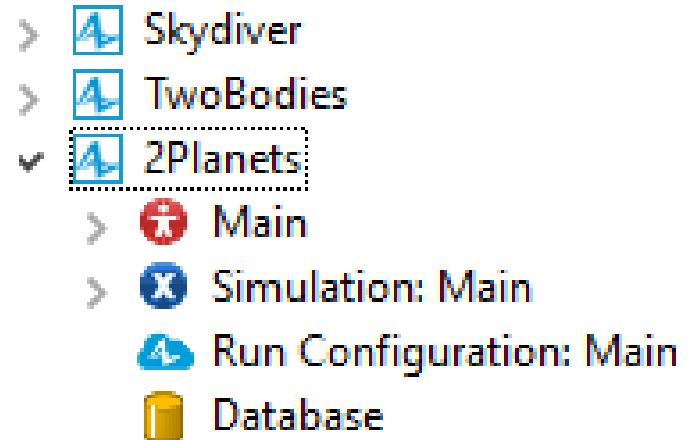
Description
Velocity cannot be resolved to a variable
Velocity cannot be resolved to a variable
Velocity cannot be resolved to a variable
TestVelocity is not used but expected
TestVelocity is not used but expected

Time uni... minutes

The AnyLogic Project

An AnyLogic project has
(at least) two parts:

- The simulation model
- One or more experiments



The model describes the system to be simulated

The experiments describe what is to be done with the model

The separation of model and experiment is very useful

Models in AnyLogic

The model consists of

- Parameters, variables, functions, events, ...

The model can contain visualizations

- Diagrams of model variable values
- Animation of model elements

These elements

- can be moved and placed freely on a canvas
- can be named using normal Java conventions

Basic Anylogic Element Types

Stocks

- Describe the system dynamics *using differential equations*
- Need only an initial value and a first derivative, no explicit dynamics (mathematical description of behavior)

Flows

- Describes a rate of change of a stock (inflow / outflow)

Parameters

- Represent ordinary Java variables (`int`, `float`, ...)
- Describe **input** parameters to the simulation

Functions

- Represent ordinary Java functions
- Return a value that is computed dynamically, potentially depending on the values of variables or parameters

Experiments in AnyLogic

There are different types of experiments (Educational Edition)

- Simulation (only one model run)
- Parameter Variation (outcomes for different parameter values)
- Optimization (automatically find suitable parameter values to minimize/maximize some expression)

An experiment can modify model parameters

- Automatically vary parameters within a given range
- Customize parameter handling through Java code

Skydiver – An Example from the Lecture

One simple physical model states:

The force due to wind resistance is proportional to the square of the velocity

We obtain

$$\frac{dv}{dt} = g - b \cdot v^2$$

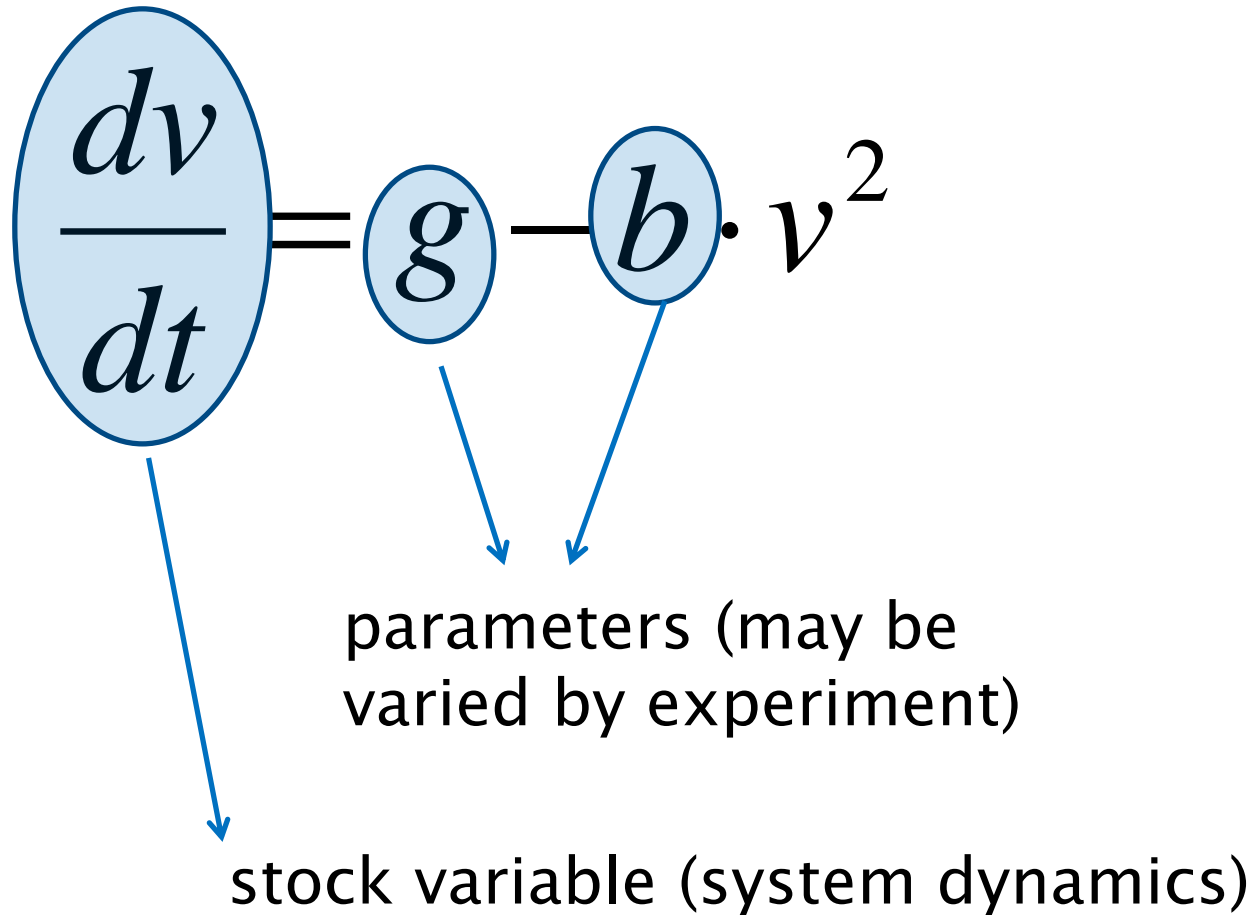
e.g. with

$g = 9.81 \text{ [m/s}^2\text{]}$ (gravity)

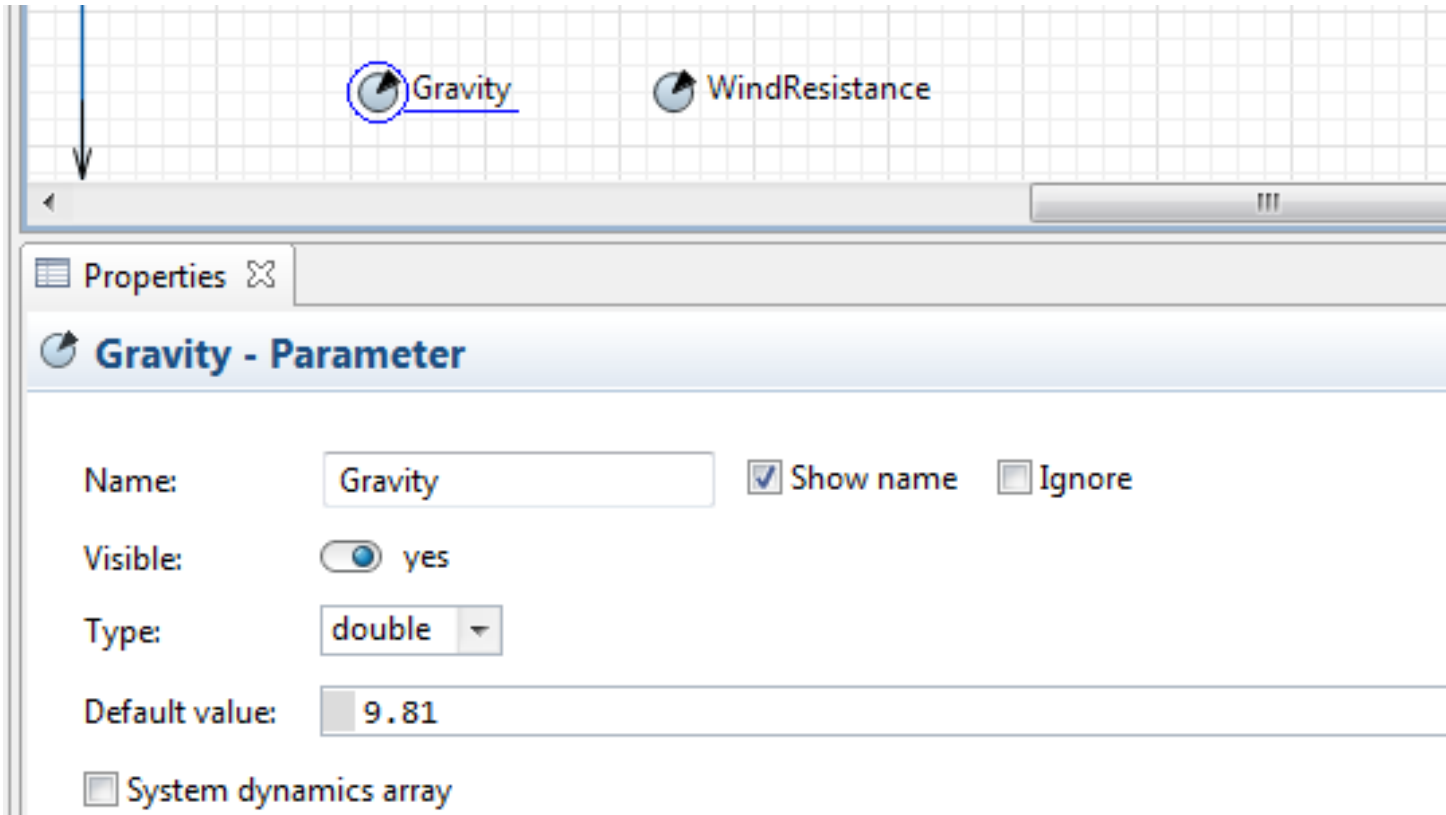
$b = 0.0033 \text{ [1/m]}$ (wind resistance of a sky diver)



Skydiver – An Example from the Lecture



Skydiver – Parameters (Model Constants)

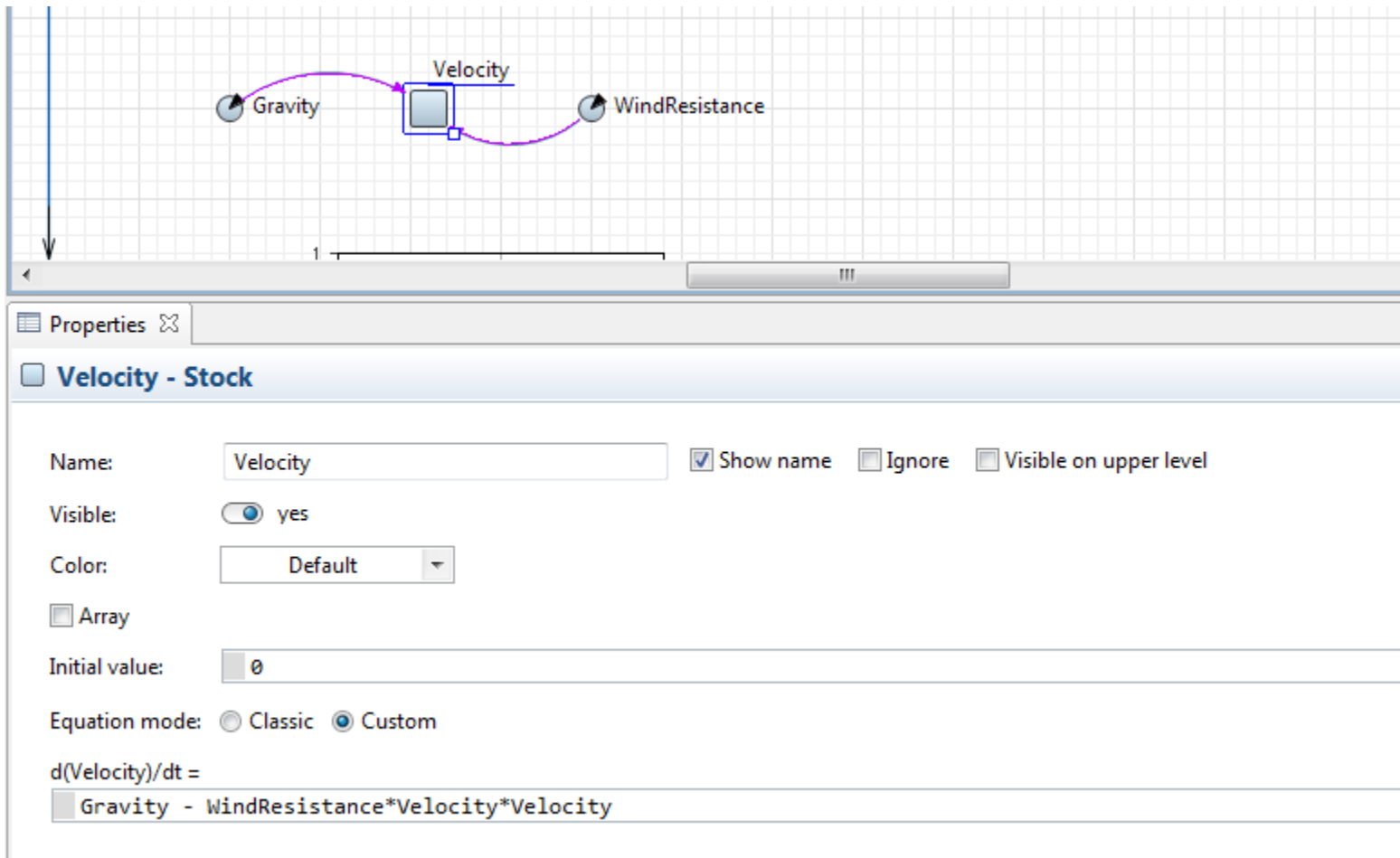


The screenshot shows the Skydiver simulation interface. At the top, there is a grid with a blue arrow pointing down, labeled "Gravity", and a circular icon labeled "WindResistance". Below the grid is a horizontal bar with a left arrow and a right arrow. Below the bar is a "Properties" tab. The "Gravity - Parameter" section is active, showing the following settings:

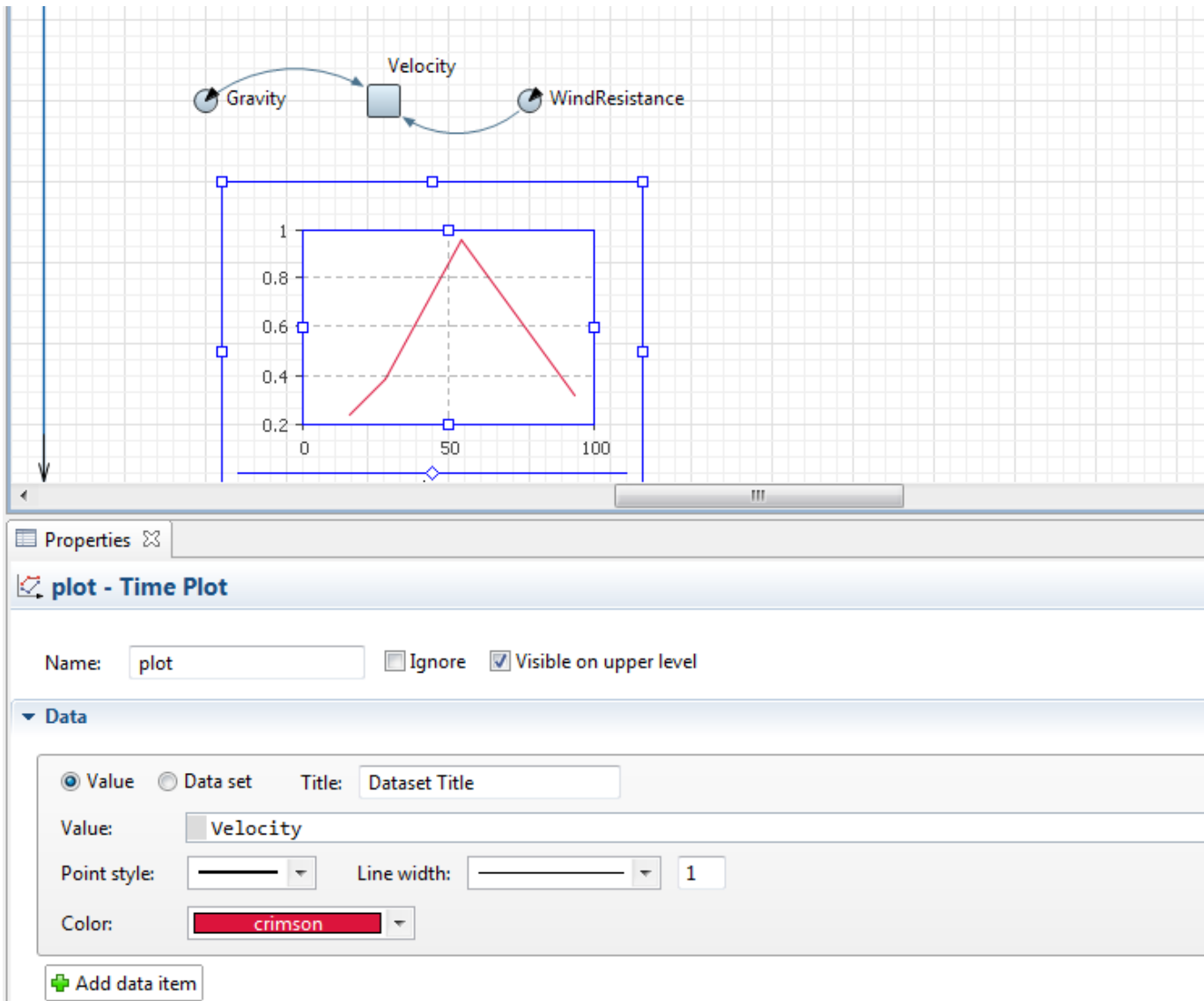
- Name: Gravity
- Visible: yes
- Type: double
- Default value: 9.81
- ☐ System dynamics array

There are also checkboxes for "Show name" (checked) and "Ignore" (unchecked).

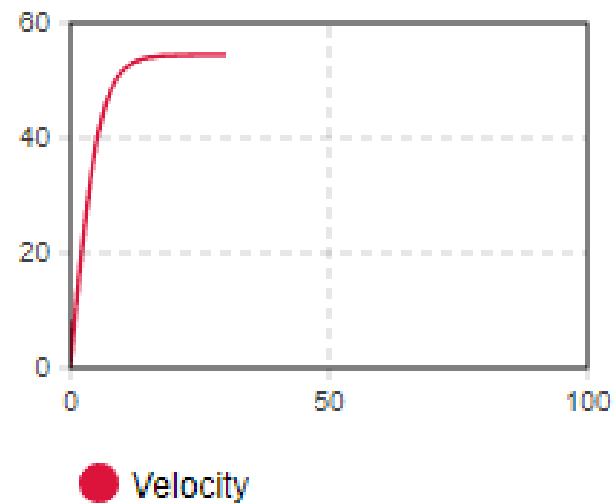
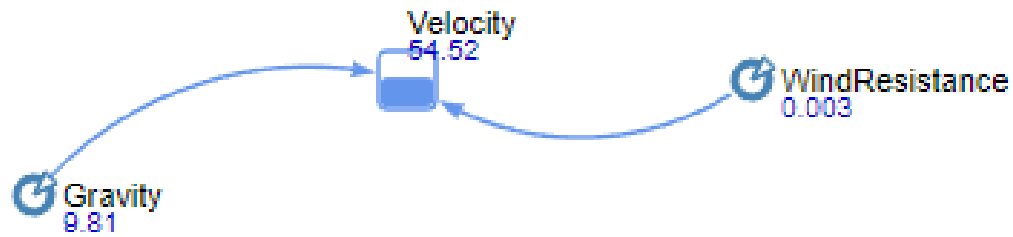
Skydiver – Stock Variable (System Dynamics)



Skydiver – Presentation



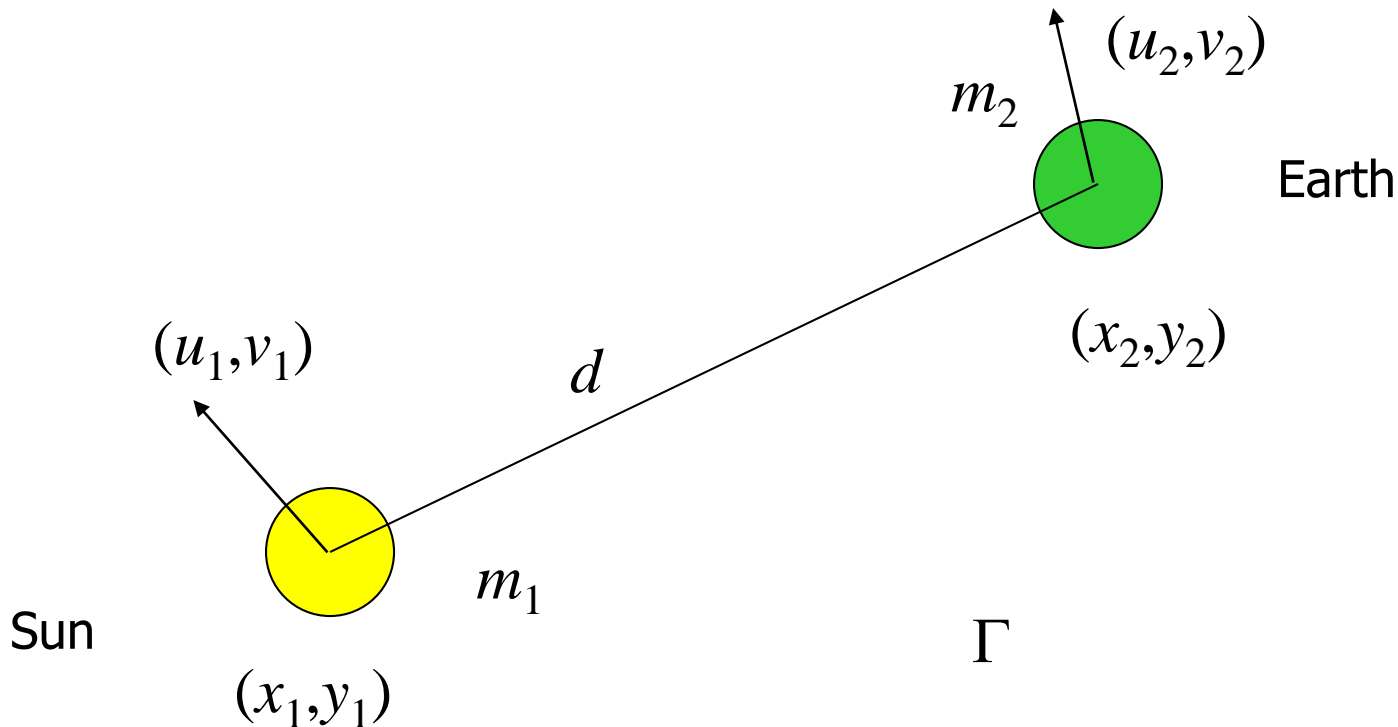
Skydiver – Result



And now it's your turn!

Simulation of Planet Orbit

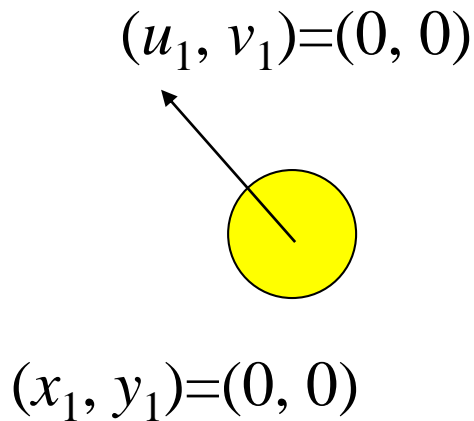
The basic quantities (2-D):



Newton's Law: Force = Mass \times Acceleration

Initial Conditions

State of the model at the beginning of the simulation:

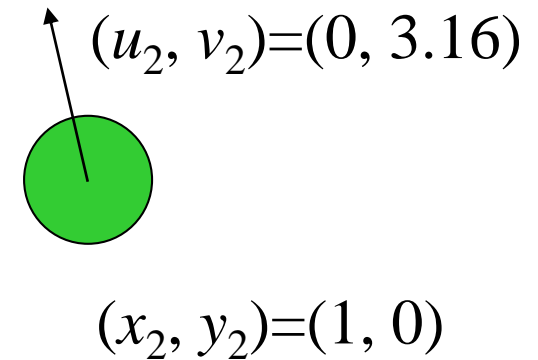


$(u_1, v_1) = (0, 0)$

$(x_1, y_1) = (0, 0)$

$$m_1 = 10$$

$$\Gamma = 1$$



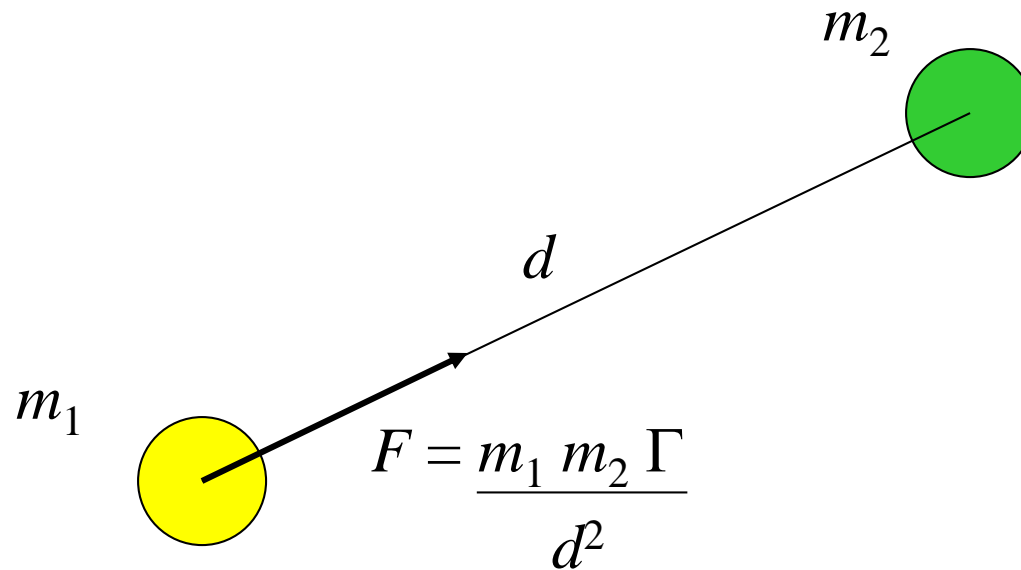
$(u_2, v_2) = (0, 3.16)$

$(x_2, y_2) = (1, 0)$

$$m_2 = 1$$

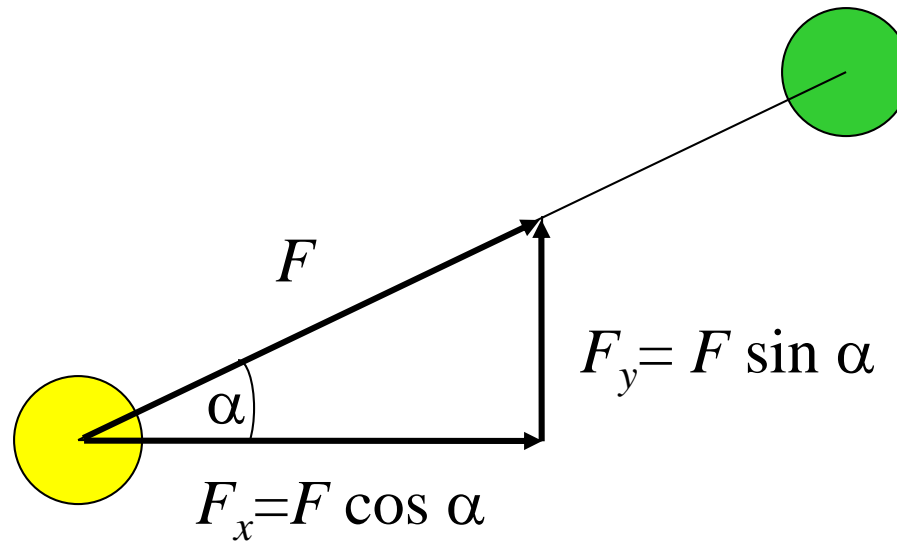
Gravitational Forces

Newton's law:



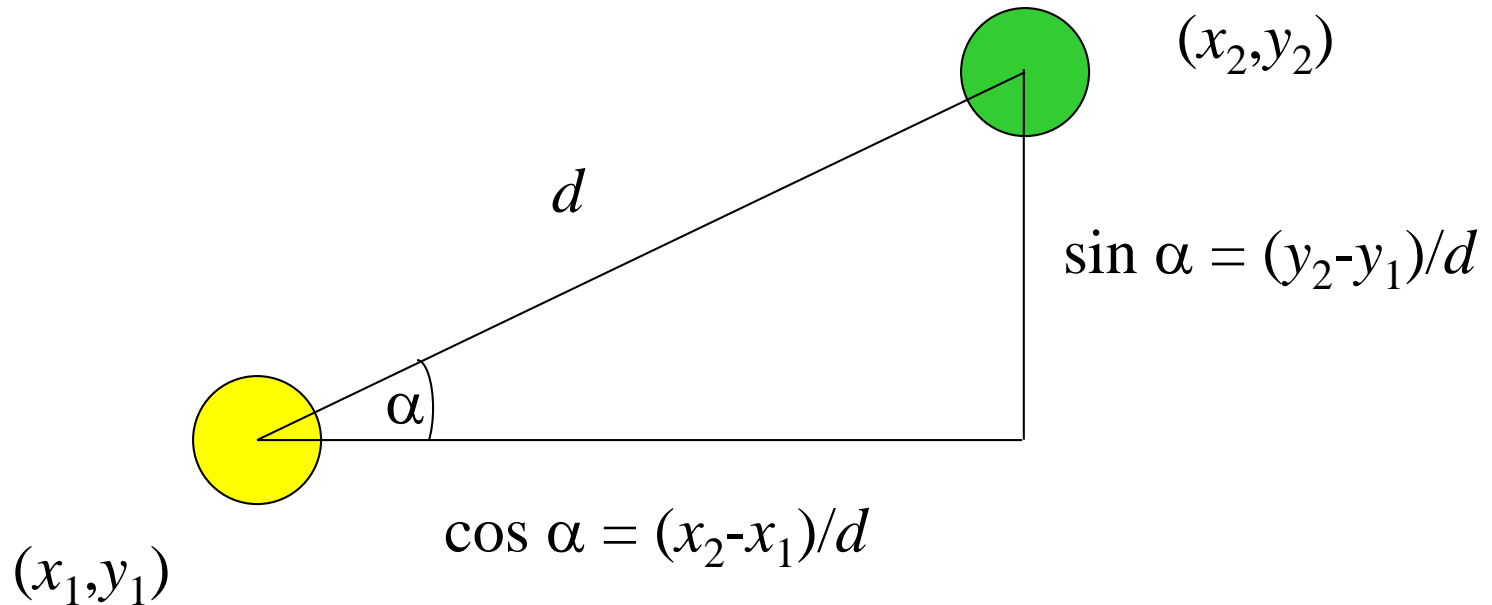
Gravitational Forces

Split F into its x and y components:



Computing `cos()` and `sin()`

Use right triangle to compute $\cos(\alpha)$ and $\sin(\alpha)$:



Governing Equations

Newton's Law:

- $\text{Force} = \text{Mass} \times \text{Acceleration}$

Definitions:

- $\text{Acceleration} = d(\text{Velocity}) / dt$
- $\text{Velocity} = d(\text{Position}) / dt$

Mathematical Model

Mathematical model:

$$\frac{du_1}{dt} = \frac{x_2 - x_1}{d} \frac{f}{m_1}$$

$$\frac{dv_1}{dt} = \frac{y_2 - y_1}{d} \frac{f}{m_1}$$

$$\frac{dx_1}{dt} = u_1$$

$$\frac{dy_1}{dt} = v_1$$

$$\frac{du_2}{dt} = \frac{x_1 - x_2}{d} \frac{f}{m_2}$$

$$\frac{dv_2}{dt} = \frac{y_1 - y_2}{d} \frac{f}{m_2}$$

$$\frac{dx_2}{dt} = u_2$$

$$\frac{dy_2}{dt} = v_2$$

Mathematical Model

Initial conditions and functions:

$$x_1 = 0 \quad y_1 = 0$$

$$u_1 = 0 \quad v_1 = 0$$

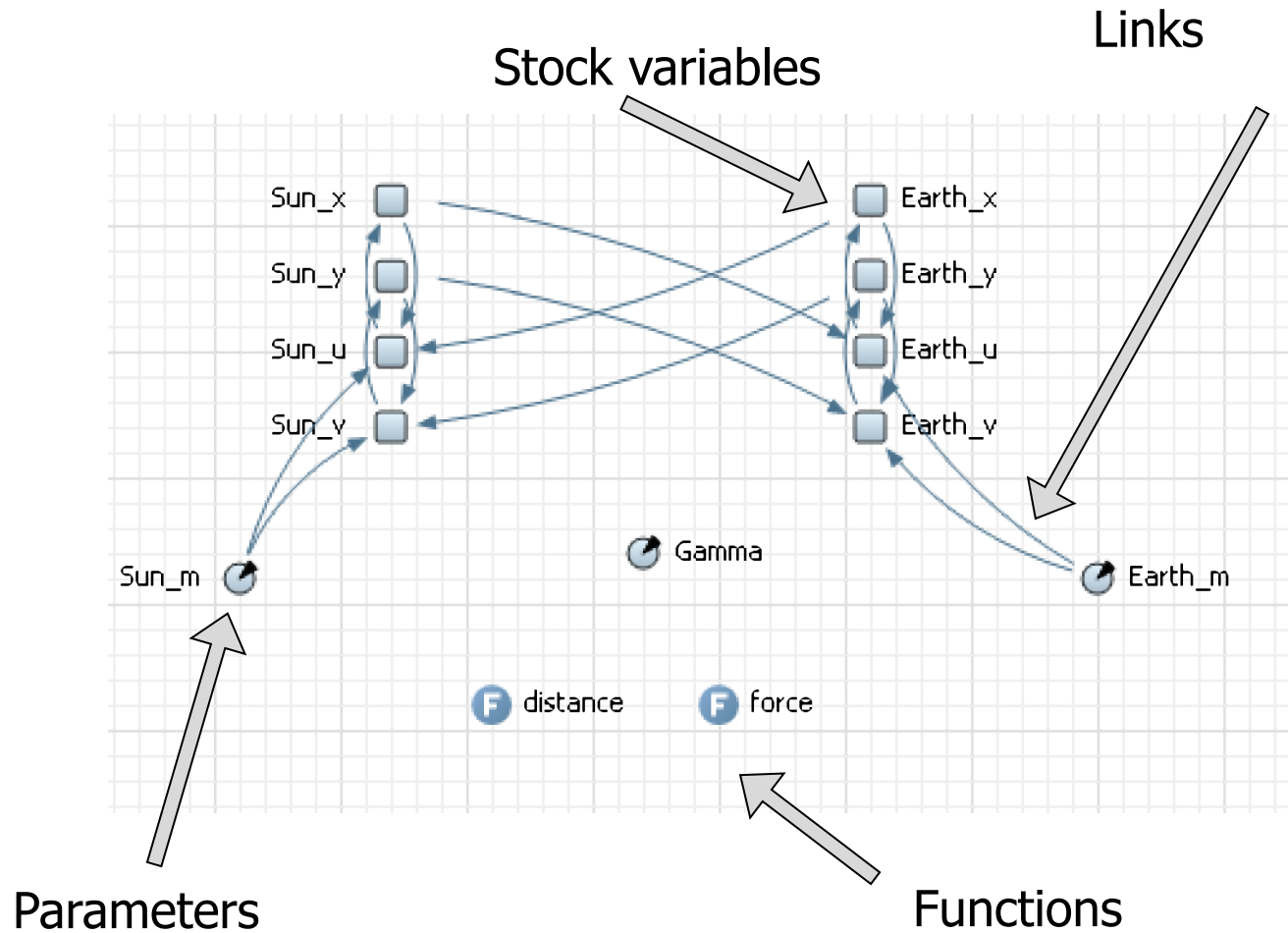
$$x_2 = 1 \quad y_2 = 0$$

$$u_2 = 0 \quad v_2 = 3.16$$

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

$$f = \frac{m_1 m_2 \Gamma}{d^2}$$

Element Types



Properties of Stock Variables

Properties

Sun_x - Stock

Name: Sun_x

☒ Show name ☐ Ignore ☐ Visible on upper level

Visible: ☒ yes

Color: Default

☐ Array

Initial value: 0

Equation mode: ☐ Classic ☒ Custom

$d(\text{Sun}_x)/dt =$
Sun_u

Properties

Sun_u - Stock

Name: Sun_u

☒ Show name ☐ Ignore ☐ Visible on upper level

Visible: ☒ yes

Color: Default

☐ Array

Initial value: 0

Equation mode: ☐ Classic ☒ Custom

$d(\text{Sun}_u)/dt =$
(Earth_x-Sun_x)/distance() * force() / Sun_m

Properties of Parameters

Properties ✕

🔄 Sun_m - Parameter

Name: ☒ Show name ☐ Ignore

Visible: ☒ yes

Type: ▼

Default value:

☐ System dynamics array

Properties of Functions

Properties

force - Function

Name: ☒ Show name ☐ Ignore

Visible: ☒ yes

☐ Just action (returns nothing)

☒ Returns value

Type:

Arguments

Name	Type	
		↑
		↓
		✕

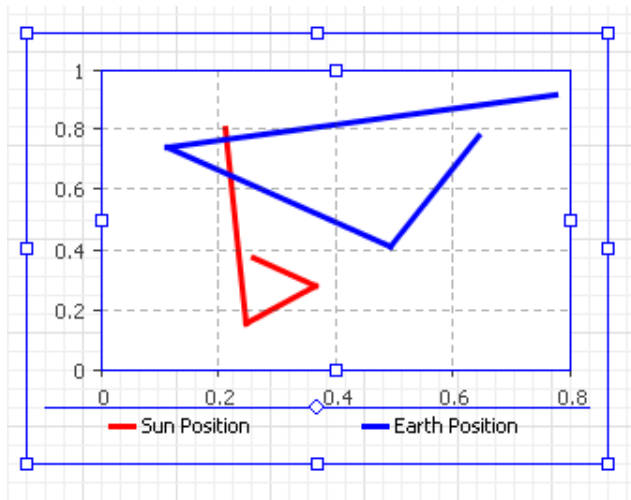
Function body

```
return Sun_m * Earth_m * Gamma / (distance() * distance());
```

Advanced

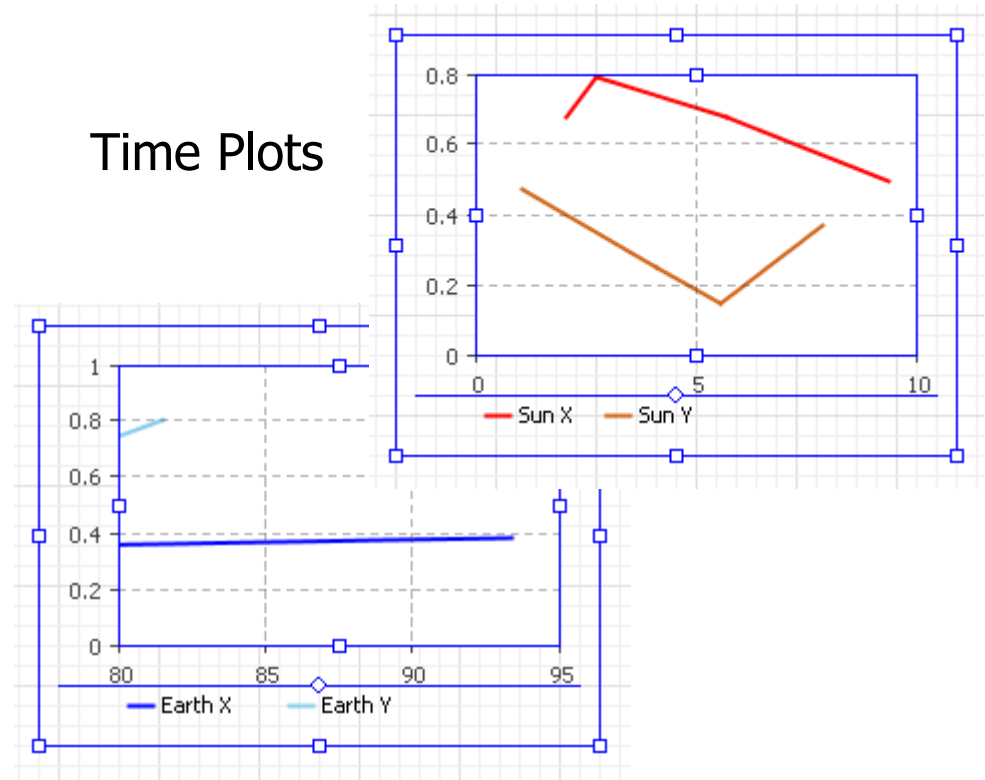
Description

Plot as visualization





Plot

Time Plots



Plot as visualization

Properties 

 **plot - Plot**



Name: ☐ Ignore ☒ Visible on upper level


▼ Data

☒ Value ☐ Data set Title:

X axis value:

Y axis value:



Point style:  Line width:  3

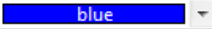
Color: 


☒ Value ☐ Data set Title:

X axis value:

Y axis value:

Point style:  Line width:  3

Color: 

 Add data item

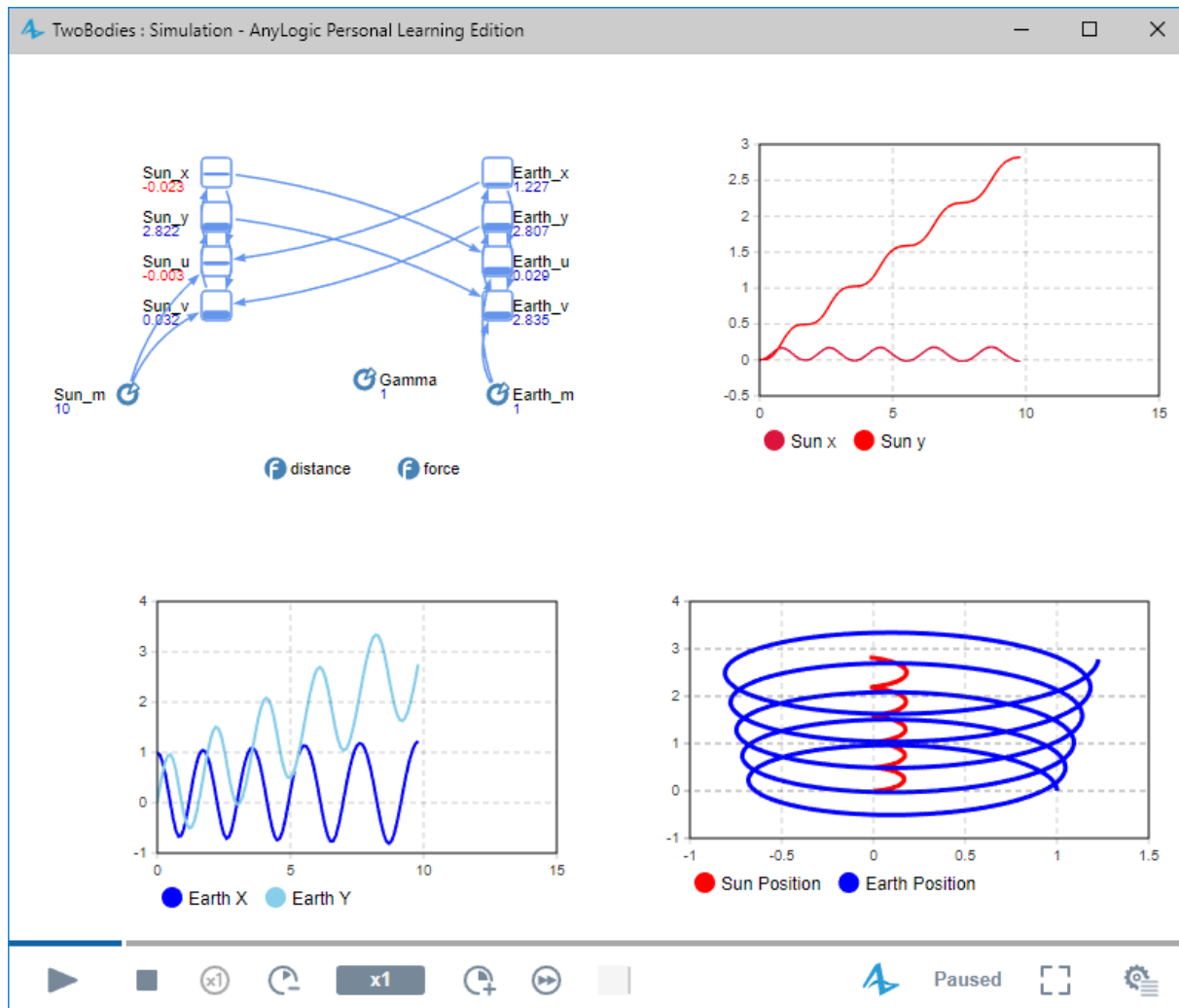
▼ Data update

☒ Update data automatically
☐ Do not update data automatically

Recurrence time:

Display up to latest samples (applies to "Value" data items only)

Simulation Experiment



Learning Goals

After this class:

- You are able to solve the first question of the last ItS exam
- You know how to model the differential equations of the Semester Assignments in AnyLogic