CAS 741: SRS

Dynamical Systems: MPS

Karol Serkis
serkiskj@mcmaster.ca
GitHub: karolserkis

December 18, 2018

Contents

Revision History

Table 1: Revision History

Date	**Developer(s)**	**Change**
October 8, 2018	Karol Serkis	First full draft for submission
October 4, 2018	Karol Serkis	First full draft
October 3, 2018	Karol Serkis	First revision and all content sections added
September 28, 2018	Karol Serkis	First draft of document in landscape orientation for presentation
September 26, 2018	Karol Serkis	SRS presentation slides discussed with Dr. Spencer Smith
December 14, 2018	Karol Serkis	All GitHub issues and comments addressed

Reference Material

This section records information for easy reference: (Units, constants, symbols, abbreviations, and acronyms)

Notation

This section describes the notation conventions used in this document.

Mathematical Notation

The notation in this document follows the standard mathematical notation conventions. The standard mathematical spaces (specifically Euclidean space in this project) are used for the symbols in this document (see Table of Symbols). Example of mathematical notation usage and for double pendulum below:

$$x_1 = l_1 \sin\theta_1 \qquad y_1 = -l_1 \cos\theta_1$$

$$x_2 = l_1 \sin\theta_1 + l_2 \sin\theta_2 \qquad y_2 = -l_1 \cos\theta_1 - l_2 \cos\theta_2$$

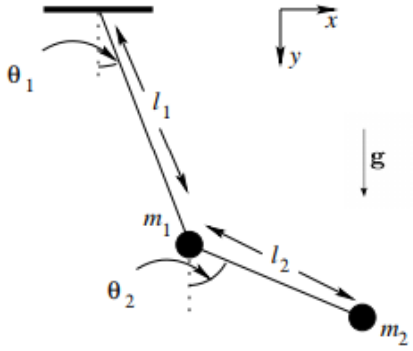

Figure 1: A simple gravity double pendulum (Szuminski, 2012)

Table of Units

Throughout this document SI (Système International d'Unités) is utilized as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	**unit**	**SI**
m	length	metre
kg	mass	kilogram
s	time	second
°	angle	degree

Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with calculus, ordinary differentials (ODE), the Lagrangian, kinematics etc. The standard mathematical spaces are used (e.g. $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, etc.) as well as some additional spaces defined in the following table.

symbol	**space**	**unit**	**description**
g	$\mathbb{R}$	–	gravitational constant
m_1	$\mathbb{R}$	kg	mass of the 1st pendulum weight
m_2	$\mathbb{R}$	kg	mass of the 2nd pendulum weight
m_n	$\mathbb{R}$	kg	mass of the nth pendulum weight
l_1	$\mathbb{R}$	m	length of the 1st pendulum rod
l_2	$\mathbb{R}$	m	length of the 2nd pendulum rod
l_n	$\mathbb{R}$	m	length of the nth pendulum rod
θ_1	$\mathbb{R}$	°	amplitude from the pivot point
θ_2	$\mathbb{R}$	°	amplitude from the 1st pendulum weight
θ_n	$\mathbb{R}$	°	amplitude from the nth pendulum weight
L	$\sum \mathbb{R}$	–	Pendulum system Lagrangian
T	$\sum \mathbb{R}$	–	Kinetic energy of system
V	$\sum \mathbb{R}$	–	Potential energy of system

Abbreviations and Acronyms

The symbols are listed in alphabetical order.

symbol	description
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
NF	Non-Functional Requirement
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
T	Theoretical Model

1 Introduction

This documents is an SRS for the MPS program. The directory for this project can be found at GitHub: /karolserkis/CAS-741-Pendula/
This SRS template is based on (Smith and Lai, 2005) & (Smith et al., 2007) (ex. based on the principle of information hiding (Parnas, 1972)).

1.1 Purpose of Document

The purpose of this document is to describe the requirements for finding a solution for a Multi-Pendulum Simulation (MPS) program and tracking the chaotic motion of the system.

The theoretical models used in the MPS code will be provided, insuring assumptions and unambiguous definitions are identified. This document is intended to be used as a reference to provide all information necessary to understand and verify the inputs to outputs. The SRS is abstract: the contents describe the problem being solved, but not how to solve it.

This document will be used as a starting point for subsequent development phases, including writing the design specification and the software verification and validation plan. The verification and validation plan will show the steps in the software documentation/implementation.

1.2 Scope of Requirements

The scope of the MPS program is limited to the generation of a plot trajectory simulation and other related plots. This document is to describe the requirements for a MPS program solution that only focuses on multi-pendulum simulations and tracking the chaotic motion of the system. It will allow users to generate plot trajectories over time using ODE/DAE initial value problem solvers. In the case of a double pendulum you have a new system that is dynamic and chaotic and requires a set of coupled ordinary differential equation solvers. Once one introduces multiple pendulums the system becomes chaotic and interesting to model and simulate.

Assumptions: The MPS will be a closed system. Air resistance and friction will not be considered for the simulation. The MPS will be limited to the user initialized inputs and the output of the MPS will either plot trajectories over time and limit the user to a specific duration of the simulation, in order to allow diagrams and trajectory history to be saved. The plot trajectory simulation should run on a local system (personal computer) that is capable of executing the simulation without serious performance problems. The user will be able to set a range of time and initialize the system.

1.3 Characteristics of Intended Reader

Simplification of some physical concepts are proposed to make the document technically accessible and also the software to be accessible. Nevertheless, the intended reader is expected

to have a basic knowledge in mathematics (calculus, differentials/ODEs) and physics (kinematics, kinetic energy, potential energy, Lagrangian) to get a deeper understanding of the document.

1.4 Organization of Document

- This document follows the template outlined in Smith and Lai (2005)
- The presentation follows the standard pattern of presenting goals, theories, definitions, and assumptions.
- The goal statements are refined to the theoretical models, and the theoretical models to the instance models. The data definitions are used to support the definitions of the different models.

2 General System Description

This section identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

2.1 System Context

- User Responsibilities:
  - Ensure that the input data is fits the system model (i.e. correct value, only positive integers, when required) (For example, weights and lengths and other appropriate units found in Table of Units and Symbols)
  - Ensure that the input data is of the correct type (i.e. Enter an integer not char or string when integer is asked for)
- MPS program Responsibilities:
  - Detect data type mismatch, such as a string of characters instead of a floating point number.
  - Determine if the inputs satisfy the required physical and software constraints.
  - Solve the system of equations arising from the input data to generate the output data.
  - Generate a plot of the output data and generate diagrams to display to the user.
  - Ensure that simulation is within the scope of the simulation window (For example the number of pendulums cannot exceed the boundry of the GUI window of the simulation)

2.2 User Characteristics

The end user of MPS program should have an understanding of first year undergraduate math and physics. Less understanding of physics and math are required to use the software than understand this document or the inner workings of the software program.

2.3 System Constraints

The system constraint will be the display requirements for the GUI and thus a contraint on the number of pendula possible to be displayed in the simulation.

3 Specific System Description

This section first presents the problem description, which gives a high-level view of the problems to be solved and the motivation behind MPS software. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

3.1 Problem Description

The MPS software will generate a plot trajectory in a 3D plot grid. A simple gravity pendulum is very easy to system to model and consists of a weight suspended from a pivot and the weight is given enough space to swing freely. To simplify the model we assume no air resistance with a friction-less pivot. The model and calculations for the simple gravity pendulum are well defined and only require simple derivations and differential solvers.

MPS program will produce a simulation given a set of constants and input (starting position away from equilibrium position). Terminologies and the physical system are described below.

3.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

Equilibrium position: The pendulum rod and weight position in its resting state.

3D Cartesian coordinate system: The pendulum rod and weight swing from a pivot position origin (x, y, z)

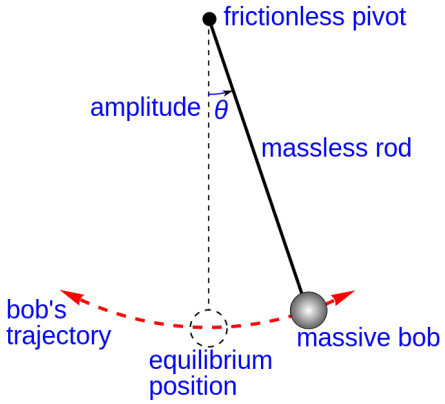

Figure 2: A simple gravity pendulum where the model assumes no friction or air resistance

Lagrangian: The Lagrangian equation $L = T - V$, where T and V are the kinetic and potential energies of the system respectively.

Poincar map: Starting with the Poincar section of the MPS in 3D space, the Poincar map P is a projection from point x onto point $P(x)$ transforming the 3D space into a 2D projection diagram plot.

3.1.2 Physical System Description

The physical system of MPS program includes the following elements:

PS1: Simulate an n-rod multi-pendulum system with no friction and no air resistance in a 3D space.

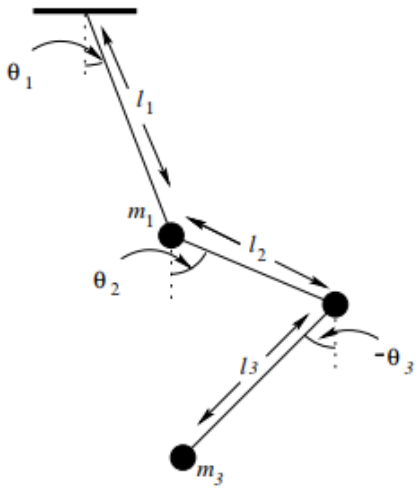

Figure 3: A triple pendulum example (Szuminski, 2012). See Table of Symbols for more info.

3.1.3 Goal Statements

Given the user input and the initial state of the MPS with reference to the table of symbols the goal statements are:

[Isn't the initial state of the multi-pendulum part of the input? —SS]

GS1: Generate a trajectory plot of the movement of the pendula from user input starting state to equilibrium state of rest and show logged statistics over time to the user. [I believe you are assuming no friction, so I don't think the pendula will come to a state of rest. —SS]

GS2: Generate a Poincar map plot of the movement of the pendula from user input starting state to equilibrium state of rest.

[Your goals are focused on plots. A more abstract output would be to generate the trajectory, without the word plot. You can still make generating a plot part of the requirements, but it would also be nice to see the option of outputting the results to a file —SS]

3.2 Solution Characteristics Specification

3.2.1 Assumptions

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [T], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1: In the model we assume no air resistance with a friction-less pivot, moving in space with respect to the Lagrangian calculations made. [T1][DD1][IM1] [You need to explicitly invoke these assumptions when they apply, not just list them. —SS]

A2: The kinetic energy will be represented by real-value and will fit the mathematical model and scope. [T2][DD1]

A3: The potential energy will be represented by real-value and will fit the mathematical model and scope. [T3][DD1]

A4: The user knows what the purpose of the simulation model and inputs weights and lengths according to possible simulation characteristics. [DD1][IM1][T1][T2][T3][T4]

[Don't you need the assumption that the angles are small enough that sin theta equals theta? —SS]

[I believe you should have an assumptions that the connecting rods are mass-less. —SS]

3.2.2 Theoretical Models

This section focuses on the general equations and laws that MPS program is based on.

Number	T1
Label	**Double Pendulum Lagrangian** ($L = T - V$)
Equation	$$L = \frac{1}{2}(m_1 + m_2)l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2 l_2^2\dot{\theta}_2^2 + m_2 l_1 l_2 \dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2)$$ $$+(m_1 + m_2)gl_1\cos\theta_1 + m_2 g l_2 \cos\theta_2$$
Description	Lagrangian model system solution
Source	(Assencio)
Ref. By	[A1][A4] [Use cross-references to generate these, rather than hard-coding. —SS]

[How is this equation derived? —SS] [I would think you would have theoretical models on kinetic energy and potential energy, defined in an abstract way, so that they would be useful for other problems. —SS] [These models are for a double pendulum, but the problem is supposed to have more than two pendulums. —SS]

Number	T2
Label	**Double Pendulum Kinetic Energy**
Equation	$$T = \frac{1}{2}m_1 v_1^2 + \frac{1}{2}m_2 v_2^2$$ $$= \frac{1}{2}m_1(\dot{x}_1^2 + \dot{y}_1^2) + \frac{1}{2}m_2(\dot{x}_2^2 + \dot{y}_2^2)$$ $$= \frac{1}{2}m_1 l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2\left[l_1^2\dot{\theta}_1^2 + l_2^2\dot{\theta}_2^2 + 2l_1 l_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2)\right]$$
Description	Potential Energy model system solution
Source	(Assencio)
Ref. By	[A2][A4]

Number	T3
Label	**Double Pendulum Potential Energy**
Equation	$$V = m_1 g y_1 + m_2 g y_2$$ $$= -m_1 g l_1 \cos\theta_1 - m_2 g(l_1 \cos\theta_1 + l_2 \cos\theta_2)$$ $$= -(m_1 + m_2) g l_1 \cos\theta_1 - m_2 g l_2 \cos\theta_2$$
Description	Potential Energy model system solution
Source	(Assencio)
Ref. By	[A3][A4]

3.2.3 General Definitions

We will use the Lagrangian and ODEs. No need for general definitions in current documentation.

3.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models. The models here are to satisfy the theoretical models constrained and closed 3D space.

Number	DD1
Label	**Closed, Real intervals**
Equation	$$\mathbb{R}(\texttt{for Langrangian equation})$$ $$P(x) : \mathbb{Z} \times \mathbb{R} \implies \mathbb{R}$$
Description	Simple cartesian coordinate model system solution
Source	(Assencio)
Ref. By	T1,T2,T3,T4,T5

[I don't know what this means/ —SS]

3.2.5 Instance Models

This section transforms the problem defined in problem description into one which is expressed in mathematical terms.

Number	IM1
Label	**Addition of closed, real intervals**
Equation	$\sum \mathbb{R}(\texttt{for Langrangian equation})$ $P(x) : \mathbb{Z} \times \mathbb{R} \implies \mathbb{R}$
Description	Simple cartesian coordinate model system solution
Source	(Assencio)
Ref. By	T1,T2,T3,T4,T5

[The instance model is the most refined version of your problem that is closest to the eventual code. I don't understand how the above IM fits in to this. If anything, isn't this a data definition? Why is it even necessary? Are you using interval arithmetic? I would think the ODEs that you are going to solve are your IMs. Your theoretical models look more like IMs. Your theoretical models could then be replaced with more abstract versions of the general concepts of Lagrangian etc. The theoretical models are supposed to be something that could be reused in a different project. —SS]

3.2.6 Data Constraints

The data constraints on the input and output variables, respectively. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

- Constraint on gravity: g = $9.8 m/s^2$

[What about the constraint that your lengths have to be positive? Are there any bounds on the initial angles? Aren't they supposed to be small? What about your masses? —SS]

3.2.7 Properties of a Correct Solution

A correct solution must satisfy the system of non-linear equations described. The user will also be able to judge the results based on the knowledge about the model and input.

[The properties here should be something in addition to meeting the requirements. Meeting the requirements is a given. We are interested here in identifying things that should be

true, but that are independent of the requirements. In your case, a correct solution should conserve energy. The energy at the start should be maintained as the simulation proceeds. —SS]

4 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

4.1 Functional Requirements

R1: MPS program should be able to have axis labels & 3D Cartesian coordinates.

R2: MPS program will take the following inputs: [It would be nice to use symbols to represent the inputs. You also want a way to identify how many masses you have. —SS]

1. The initial mass of the weights. [This is a confusing sentence, since mass and weight are different concepts. If you use a symbol, like m_i then you could probably simplify the presentation. —SS]
2. The initial length of the rods.

R3: MPS program will calculate the kinetic and potential energy after the user has set the initialization parameters of input from the user have been entered.

R4: MPS program will calculate the Lagrangian (and Hamiltonian (differential on Lagrangian) if needed) after the user has set the initialization parameters of input have been entered and the kinetic and potential energy of the system as whole has been calculated. [This is the only time you mention the Hamiltonian. You should define it in your documentation, or not mention it. —SS]

R5: MPS program will ensure that the inputs do not violate the constraints specified in the Data Constraints section:

1. MPS program will generate diagrams with and plot lines and timeline of logged movement.
2. The timeline of swings of the pendulum will be logged and eventually return to a resting state in equilibrium

4.2 Nonfunctional Requirements

MPS program will be try to be small and simple, so performance is not a priority. Any reasonable implementation will be very quick and use minimal storage. Rather than performance, the non-functional requirement priorities are correctness, understand-ability, re-usability, maintainability, and portability.

NF1: MPS program should be reliable and portable and easy to use for beginners or experts.

Correctness

- The MPS tool must be correct in its generation of plot trajectories.

Reliability

The MPS should run successfully and have error checking for user input.

Robustness

- The MPS must be able to recognize violated data constraints and report them to the user.
- The MPS tool must inform the user when it encounters any unspecified state.

Performance

Performance is a priority in the MPS specification. It needs to be able to generate a plot reasonable amount of time.

Verify-ability

- The MPS must be verifiable with respect to the correctness of its calculations. The calculation procedures used by the MPS tool must be implemented such that they can be verified using mathematical proofs.

Usability

- The user must be able to enter values using standard mathematical notation.
- The plot should generate and be large enough for the user's display.

Maintainability

- The evolve-ability of the MPS must allow the addition of real intervals.

Re-usability

Reusability is not a priority because there are currently no future products that will rely on MPS

Portability

To ensure the portability, the MPS software will be multi-platform.

[Your NFRs are ambiguous. You might have been better off with the blanket statement in the first paragraph, since the list of qualities brings attention to the fact that what you have stated is not measurable. —SS]

5 Likely Changes

LC1: Generation of simulation only in 2D

LC2: Generation of plot and diagrams only in 2D

LC3: Generation of diagrams using distributed/parallel computing

6 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well.

	A1	A2	A3	A4
T1	X			X
T2		X		X
T3			X	X
T??				X
DD1	X	X	X	X
IM1	X	X	X	X
LC1	X	X	X	X
LC2	X	X	X	X
LC3	X	X	X	X

Table 2: Traceability Matrix Showing the Connections Between Assumptions and Other Items

	T1	T2	T3	T??	DD1	IM1
T1	X	X	X	X	X	X
T2	X	X	X	X	X	X
T3	X	X	X	X	X	X
T??					X	X
DD1	X	X	X		X	X
IM1	X	X	X		X	X

Table 3: Traceability Matrix Showing the Connections Between Items of Different Sections

	T1	T2	T3	T??	DD1	IM1
R2	X	X	X		X	X
R3	X	X	X	X	X	X
R4	X	X	X		X	X
R5	X	X	X	X	X	X

Table 4: Traceability Matrix Showing the Connections Between Requirements and Instance Models

References

Diego Assencio. *The double pendulum: Lagrangian formulation*. Available at https://diego.assencio.com/?index=1500c66ae7ab27bb0106467c68feebc6.

David L. Parnas. On the criteria to be used in decomposing systems into modules. *Comm. ACM*, 15(2):1053–1058, December 1972.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Ågerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages 107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

W. Spencer Smith, Lei Lai, and Ridha Khedri. Requirements analysis for engineering computation: A systematic approach for improving software reliability. *Reliable Computing, Special Issue on Reliable Engineering Computation*, 13(1):83–107, February 2007.

Olsztyn Szuminski. *Dynamics of multiple pendula*, 2012. Available at http://wmii.uwm.edu.pl/~doliwa/IS-2012/Szuminski-2012-Olsztyn.pdf.