Table of Contents

Physiovalues documentation

This is user's guide and developer's guide of the open-source project www.physiovalues.org. The project consist of these applications:

- sim.physiovalues.org Online web simulation tool allows to show several screens, manipulate selected parameters and perform simulation. The idea and details was described by T.Kulhánek et al. [1].
- app.physiovalues.org Online web parameter estimation tool allows to perform parameter estimation using "curve-fitting" method and genetic algorithm selecting values of parameters which are simulated. The details and results were described by T.Kulhánek et al. [2].
- www.physiome.lf1.cuni.cz/ident3 BOINC server with a project for parameter sweep application. [3]

The topic of modeling the medical physiology is covered by the works maintained by the physiomodel project www.physiomodel.org and a prepared book which draft is published at Medical Physiology in Modelica.

References

> [1]. T. Kulhánek, F. Ježek, M. Mateják, J. Šilar, P. Privitzer, M. Tribula, et al., RESTful Web Service to Build Loosely Coupled Web Based Simulation of Human Physiology, Trans. Japanese Soc. Med. Biol. Eng. 51 (2013) R–32. doi:10.11239/jsmbe.51.R-32. ↩

> [2]. T. Kulhánek, M. Mateják, J. Šilar, J. Kofránek, Parameter estimation of complex mathematical models of human physiology using remote simulation distributed in scientific cloud, in: Biomed. Heal. Informatics (BHI), 2014 IEEE-EMBS Int. Conf., 2014: pp. 712–715. doi:10.1109/BHI.2014.6864463. ↩

> [3]. T. Kulhánek, Utilization of GRID technology in processing of medical information, Dissertation, thesis.pdf ↩

Developer's Guide

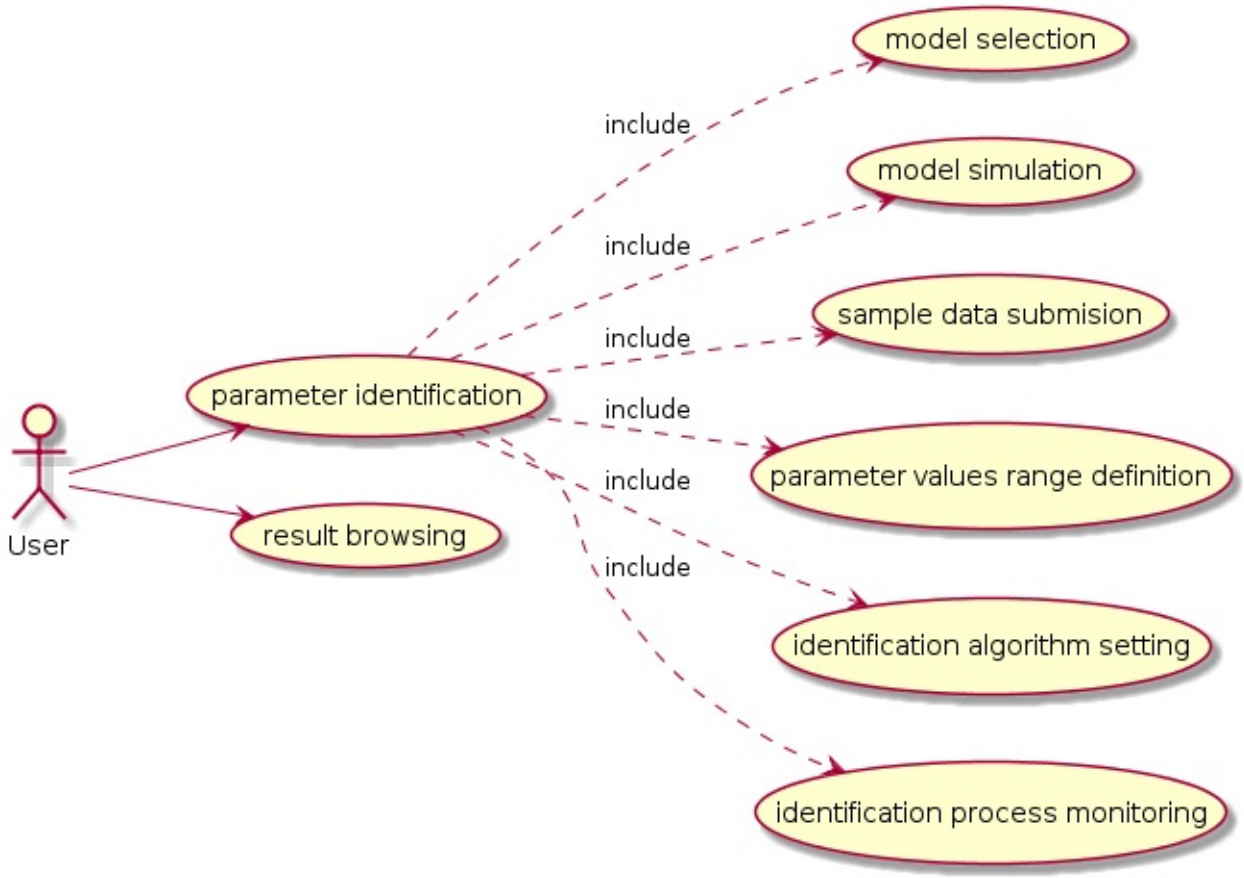
This part of documentation describes the source code and architecture of the subsystems. It presumes the knowledge and overview of the following technologies:

- REST - firstly noted by the work of Fielding and now used as a style for building scalable web services
- C#, .NET - used as backend for the REST service
- ServiceStack - framework utilizing some known design patterns to simplify development and maintenance of web services
- Javascript, HTML5, AJAX - used for frontend web application communicating with backend
- FMI - standard for exporting and integrating Modelica models with control application logic

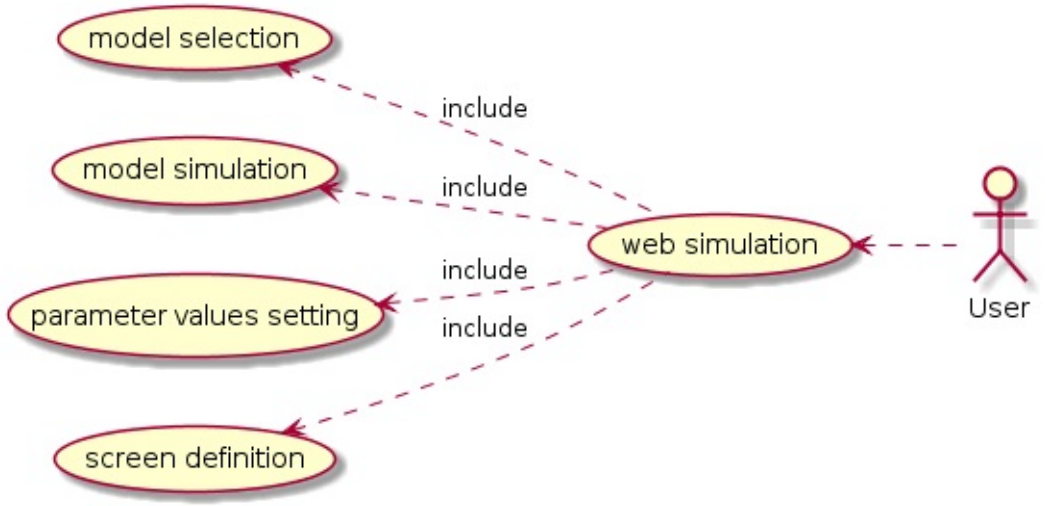
Web App Architecture

The system provides these use cases:

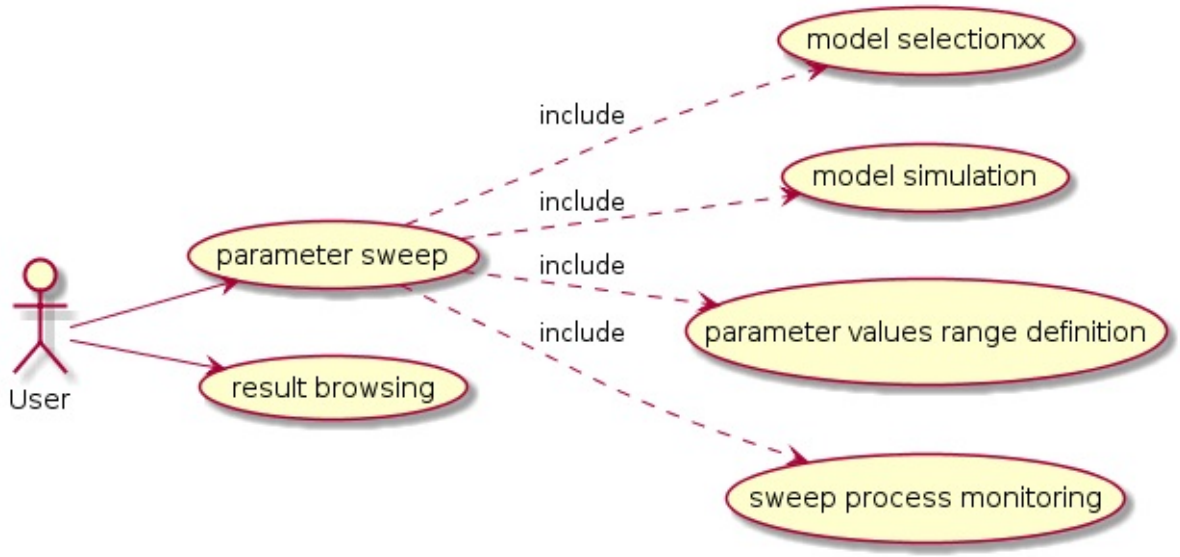
Parameter Identification - including model selection, sample data submission etc.



Web Simulation - including model selection, parameter settings, screen definition.



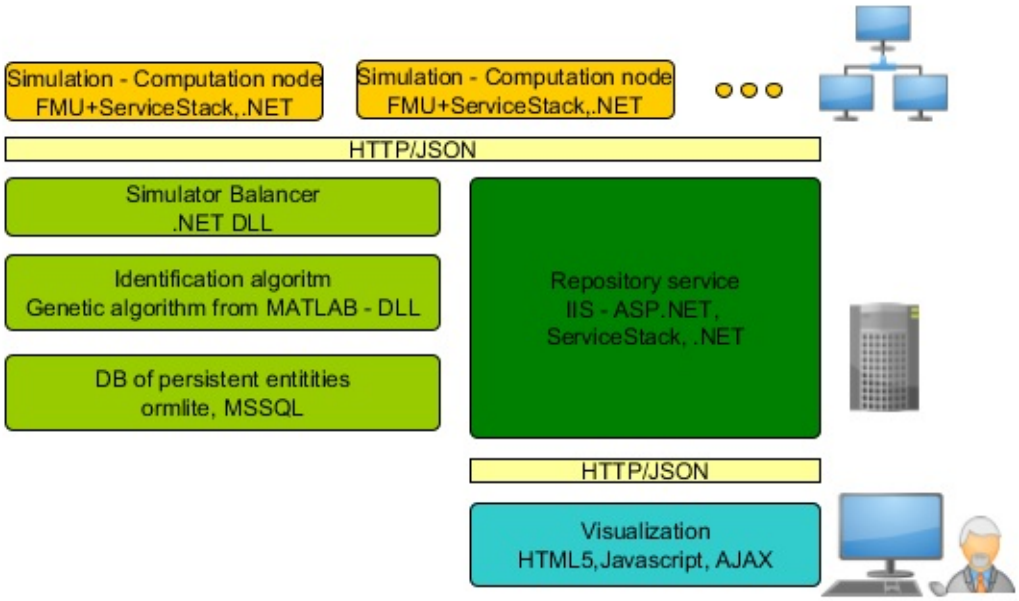
Parameter Sweep analysis - including model selection, parameter values range definition,



etc.

The use cases share common features like model selection, one instance of model simulation.

Architecture of the modules



Server modules for parameter identification

from https://github.com/TomasKulhanek/Physiovalues

`boinc` scripts and configuration for boinc modules
`doc` documentation
`src` source codes of server modules

`test` tests

- `src/IdentificationAlgorithm/*` MATLAB files utilizing Global Optimization Toolbox and genetic algorithm ga()
- `src/RestMasterService/*` web app with front-end (HTML, AJAX javascript) and backend (C# .NET) to control identification process
- `src/SimulatorBalancerLibrary/*` .NET DLL library to balance simulation requests going from MATLAB to remote computation nodes
- `boinc/*` BOINC related scripts and customization to configure BOINC server with dedicated

Further details in chapter server_modules.md.

FMI driver for C/C++, .NET and ControlWeb

from https://github.com/TomasKulhanek/Physio.FmiDriver4CW

FMU Driver for Control Web[1] and managed .NET (C#,C++)

`doc` Documentation for ControlWeb integration `include` Header files `lib` Third party libraries - FMI library from Modelon `project` project files to compile under MS VS2010 `src` C source files `test` Test project of ControlWeb, Modelica model and FMI driver `thirdparty` Sources of thirdparty library

The release contains

- CW2FMIGenerator.exe - generator of PAR file from FMU package
- CW2FMIDriver.dll - driver for ControlWeb 3+, .NET
- fmilib_shared.dll - fmilibrary redistribution package from fmi-library.org

Computational node

from https://github.com/TomasKulhanek/Physio.FmiRestService

- `BatchWrapperTest/*` unit test form FmiBatchWrapper
- `FmiBatchWrapper/*` console application for performing FMI simulation, parameter sweep arguments, BOINC client application
- `FmiRestServiceWrapper/*` local server giving RESTful web service for performing simulation via HTTP
- `TestFmiRestService/*` unit test for FmiRestServiceWrapper

FMI driver

from https://github.com/TomasKulhanek/Physio.FmiDriver4CW

- `doc/*` instruction for control web driver instalation and helper files generation
- `include/*`
- `lib/*` binaries of fmi library from jmodelica[2]
- `project/*` - ms visual studio project files
- `src/*` - C,C++ sources
- `test/*`
- `thirdparty/*` - sources of fmi library from jmodelica

Web Simulator

from https://github.com/TomasKulhanek/Physio.WebSimulator.git `HumModWebSimulator/*` web app with front-end (HTML,AJAX javascript) and backend (C# .NET) to define screen of specific simulator and show web simulators.

> [1]. http://www.mii.cz/ ↩

> [2]. http://www.jmodelica.org/FMILibrary ↩

Parameter Identification Module

from https://github.com/TomasKulhanek/Physiovalues

`doc` documentation
`src` source codes od server modules
`test` tests

Identification Algorithm

`IdentificationAlgorithm/` Contains MATLAB files utilizing Global Optimization Toolbox and genetic algorithm ga().

`identify_main.m` Main function, loads DLL to perform simulation e.g. SimulatorBalancer.DLL (which delegates simulation request to available computation nodes via HTTP and REST api), reshapes the array of parameter and variables, initializes the log files (absolute paths are wired) and performs minimalization

`identify_minimize.m` prepares data for genetic algorithm and performs ga() call with callback to identify_objective

`identify_objective.m` is called by the genetic algorithm with set of values of parameters to be evaluated. This function calls identify_evalModel and identify_ssq to perform simulation and quantify distance from sample data.

`identify_simulate.m` is called by identify_evalModel. Executes the call from .NET DLL of mySimulator.Simulate().

compilation

This module must be compiled using MATLAB Compiler as .NET Assembly - IdentificationAlgorithm.prj contains preconfigured functions which will be in the .NET Assembly class available for calling from .NET. After compilation the resulting DLL can be used in any application. On other computer the Matlab Compiler Runtime is needed to be installed.
The compilation was done and tested with MCR 8.1.

Web Application

C# ASP.NET web application. Uses these external package for .NET: ServiceStack, SignalR; and Javascript: JQuery, Dygraph, Handsontable.

`RestMasterService/*` contains web application with front-end (HTML, AJAX javascript) and backend (C# .NET) to control identification process.
The back-end is based on REST service utilizing ServiceStack framework and controlin/notification baased on SignalR framework.

`WebApp/GenericUI.html` `WebApp/GenericUIen.html` -HTML with embeded Javascript code utilizing jquery,signalr,dygraph,handsontable etc. to collect inputs from web interface, start identification process, show notification from server and browse results.

`WebApp/IdentifyStateTicker.cs` Performs external call of matlab library to estimate parameters, pass all arguments from the user, notify about the changes the clients. SignalR is used to notify all connected clients about the changes of computation, errors, etc.

`Webapp/IdentifyStateHub.cs` Interface for controlling the computation using SignalR.

`ComputationNodes/Workers.cs` RESTful web service utilizing ServiceStack processing the requests on HTTP URLS:

- `/workersByName/{ModelName}`
- `/workers`
- `/workers/{Id}`

Computation node register it's models by PUT on `/workers/{Id}` or POST on `/workers`. All registered workers can be listed via GET on the above URLs.

`ComputationNodes/Results.cs` RESTful web service utilizing ServiceStack processing the requests on HTTP URLS:

- `/results` POST - registers new result, GET gets all results
- `/results/{Id}` PUT - register new,update existing with an ID, GET - gets existing result with Id

compilation

The Visual Studio 2013 Solution - `RestMasterService.sln` can be used for compilation of the C# .NET modules.

Simulator Balancer

C# .NET Library to provide balancing feature to registered computation nodes.

`SimulatorBalancerLibrary/*` .NET DLL library to balance simulation requests going from MATLAB to remote computation nodes, based on their previous registration on ComputationNodes/W#3orkers.

compilation

The SimulatorBalancer is compiled as DLL, which is then loaded by the Identification Algorithm.

Web Simulator Module

C# ASP.NET with ServiceStack RESTfull web services holding the simulation screen definition and simulation screen viewer.

Computational Node Module

from https://github.com/TomasKulhanek/Physio.FmiRestService

- `FmiBatchWrapper/*` console application for performing FMI simulation, parameter sweep, etc. It is used as the BOINC client application to perform parameter sweep in obtained parameters
  `Program.cs` main program with following arguments:

```
FMIBatchWrapper [resultfilename] [modelname] [paramnames]
                [paramvalues] [paramsweppnames]
                [parametersweepstartvalues]
                [parametersweepstopvalues] [parametersweepsteps]
                [variabletoreturn] [starttime] [stoptime]
                [steps] [stepstoreturn(<=steps)]
```

- `FmiRestServiceWrapper/*` local server giving RESTful web service for performing simulation via HTTP. The application uses ServiceStack for building REST api via HTTP. The `Program.cs` is main program and detects all FMU files in working directory. These FMU's are presented and available to simulation via HTTP using URL: http://localhost:port/simulation/[fmufilename]/
- `BatchWrapperTest/*` unit test form FmiBatchWrapper
- `TestFmiRestService/*` unit test for FmiRestServiceWrapper

Known Issues

An attempt was made to load a program with an incorrect format.

```
2016-02-18 11:15:49.8855 ERROR exception during identification The type initializer fo
r 'IdentificationAlgorithm.Class1' threw an exception. stacktrace:   at Identification
Algorithm.Class1..ctor()
   at RestMasterService.WebApp.IdentifyStateTicker.IdentifyComputation() in c:\Users\t
omaton\Documents\KOFRLAB-simenv\VersionedProjects\Physiovalues\src\RestMasterService\W
ebApp\IdentifyStateTicker.cs:line 158
2016-02-18 11:15:49.8855 ERROR innerexception:The type initializer for 'MathWorks.MATL
AB.NET.Utility.MWMCR' threw an exception. stacktrace:   at IdentificationAlgorithm.Cla
ss1..cctor()
2016-02-18 11:15:49.8855 ERROR innerexception:The type initializer for 'MathWorks.MATL
AB.NET.Arrays.MWArray' threw an exception. stacktrace:   at MathWorks.MATLAB.NET.Utili
ty.MWMCR..cctor()
2016-02-18 11:15:49.8855 ERROR innerexception:An attempt was made to load a program wi
th an incorrect format. (Exception from HRESULT: 0x8007000B) stacktrace:   at MathWork
s.MATLAB.NET.Arrays.MWArray.mclmcrInitialize2(Int32 primaryMode)
at MathWorks.MATLAB.NET.Arrays.MWArray..cctor()
```

This is caused by 32-bit vs. 64-bit DLL mismatch. Install the MCR 8.1 32-bit version, compile the IdentificationAlgorithm, RestMasterServer, SimulatorBalancerLibrary in 32-bit version (AnyCPU, x86). Or install 64-bit and compile with 64-bit version.

User's Guide

This part of documentation describes the functionality from the user's view perspective. Doesn't presume any other technology knowledge. The functionality covers some parts of system analysis. For the overview of the topic of system analysis you may find usefull the book of M.Khoo[1]. The sensitivity analysis is well covered by the work of Saltelli et. al[2].

[1] M. C. K. Khoo. Physiological Control Systems. IEEE press, 2000. [2] A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto. Sensitivity Analysis in Practice.

1. doi:10.1002/0470870958.

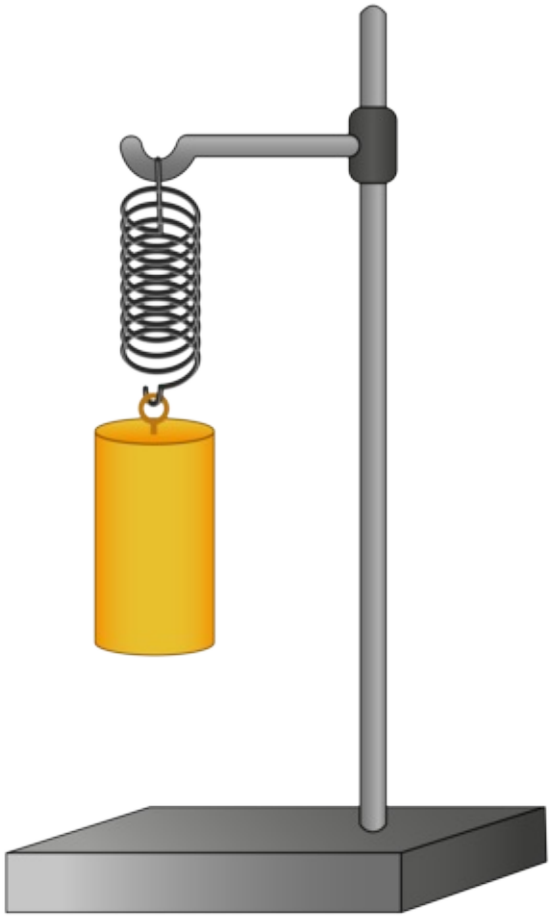
Parameter Estimation Tutorial

Introduction

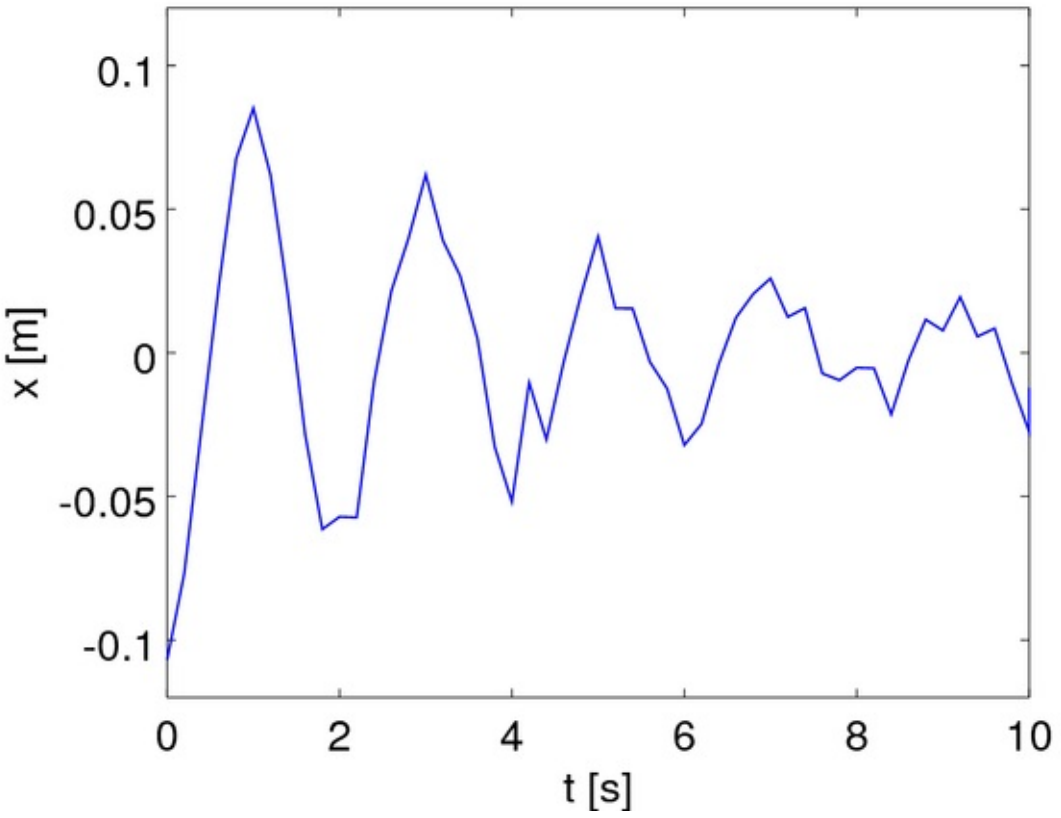
Parameter estimation (or model identification) is process of searching for parameter values of a model so that the model gives some expected results. Let us demonstrate it by an example.

Experiment

Imagine we have a real mechanical oscilator (a mass suspended on a spring).



Weight of the mass is known (let say 1kg), but stifness and damping factor of the spring are unknow and we want to determine them using model paremeter estimation. We perform an experiment: we pull the mass, release it and let it oscilate. The position of the mass is repetitively measured. Results of the measurement are



- Download the measured data: experimentalData.txt

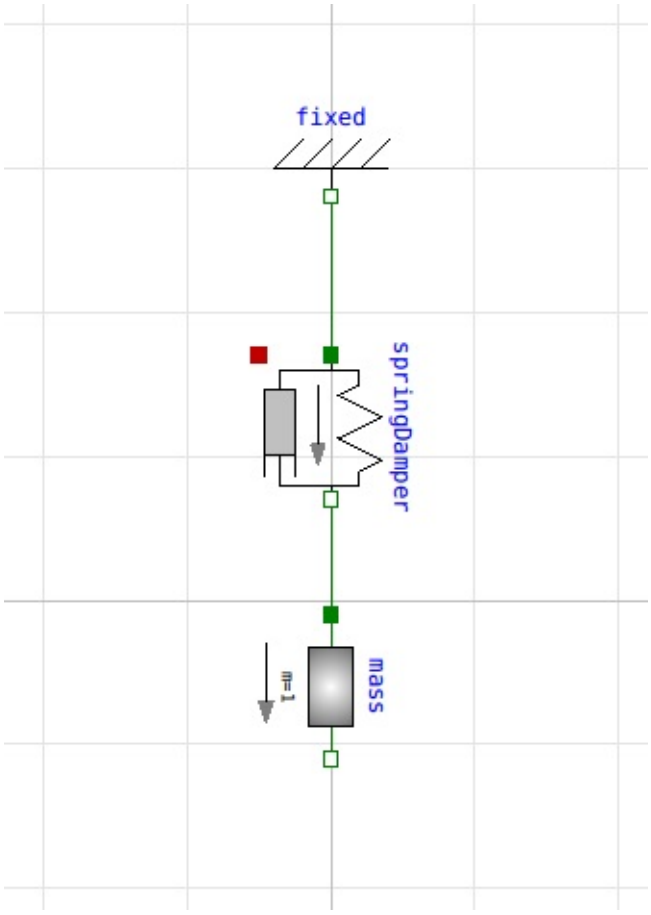
Values are noisy, as the measurement method was not precise.

Model

Mathematical model of the mechanical oscilator is

$$m\frac{d^2x}{dt^2}+kx+c\frac{dx}{dt}=0.$$

m is weight of the mass, k is the spring stiffnes constant and c the spring damping factor. We implement the model using Dymola:



- Download the model: damped_oscillator.mo

You may learn about modeling using Modelica at e.g. book.xogeny.com.

The model must be exported using FMI (Functional Mockup Interface) – a standardised interface for cooperation of diferent simulation tools. It must be FMU version 1.0, type “Co-simulation” generated under Windows platform.

- Open dymola and load the model: File→ Open
- Choose Type: Co-simulation using Dymola solvers, Version: 1.0
- Confirm to generate the FMU. The fmu file should be generated next to the model file.
- Open the FMU export dialog: Simulation→ Translate→ FMU
- Click the “Simulation tab” at bottom right.

Ready fmu may be also downloaded: damped_oscillator.fmu

Using the identification application

- Open the application: http://physiome.lf1.cuni.cz/identifikace
- Login using your google account
- Upload the fmu file using “Browse” and “Upload” buttons.
- Choose our fmu file in the list.

- Change to “Sample data” tab
- Delete the previous data
- Paste our experimental data from the downloaded file. First line is heading containing names of variables. First column is “time”, other columns contain variables to be compared with the model solution, here “mass.s” only. When you enter the data, you should see them plotted on the right.
- Change to “Parameters” tab
- Fill in parameters you want to estimate or set their value. First column is name.
    - If the checkbox “identify?”in the second column is checked, corresponding parameter is estimated, “initial value” is a hint value for the identification algorithm, “minimal” and “maximal” are extramal values, so that values outside this interval are never used. Some parameter values may cause crash of the model evaluation e.g. due to division by zero. It is helpful to keep this values outside the interval.
    - If the “identify?” check box is not checked,the “initial value” actualy determines fixed value for this parameter,so that it is not estimated.
    - Parameters of the model that are not listed here take their default values from the model. For now you may input two lines:

name	identify?	initial value	minimum	maximum
springDamper.c	x	25	0.1	50
springDamper.d	x	5	0.01	10

- Chenge to “Identification” tab.
- Fill in parameters for the genetical optimization algorithm. “generations” is maximal number of iterations, “polulations” is number of parallel evaluation per one iteration. “tolfun” influence the accuracy of the evaluation. Enter eg. generations=1000, populations=20, tolfun=1e-5 for now.
- Start the identification (button “start”). Actual best estimate of unknown paremeters is in the table below, estimated model data compared to the experimental data are plotted on the right during the computation.
- Results of previous identifications may be browsed in the “Results” tab.

Parameter Sweep

Web Simulator

Is accessible at http://sim.physiovalues.org First choose model which have some screen definition. Second choose the screen. Third see simulation In the simulation edit tab the editor supports this syntax:

screen [screenname] - should be at the first row of the definition, screenname is the name with namespace delimited by dot '.' which is visualized as menu and submenus. Example: screen Clinic.Chart
model [modelname] - modelname defines the model which will be simulated. Example: model HummodHab label [title] - logical divider of some variables or section. Example label Pressure value [title] [modelvariablename] - will try get the value of modelvariablename and shows as a number. Example: value Pressure CVS.heart.leftVentricle.Pressure graph [title] [listofvariables] - listofvariables are separated by comma ','. smallgraph [title] [listofvariables] - half sized graph, listofvariables are separated by comma ','. slider [title] [modelvariablename] [min] [default] [max] - slider from min to max with the default value bar [title] [modelvariablename] [defaultvalue] - show a bar to compar with default value radiobutton [title] [variable] [label1] [value1] [label2] [value2] ... - radio buttons with labels and discrete values. text [anytext] - anytext will be shown. svg [svgid] [svgelementid] [svgattribute=expression with model variables] - will render SVG object of svgid and connects the value of svgattribute with value from algebraic expression. Special editor for svg objects at GraphicElements.htm createjs [functiontocall] [expression with model variables] - will render createJS script (exported from Adobe Flash) and at each animation point it will call "functiontocall" with the value in "expression.." documentation [language] - documentation section begins, other rows may contain HTML or special tags:

```
    #W [wikipedia_page] - the link to wikipedia and some preview content from wikipedi
a will be rendered
    #I [wikiskripta_page] - the link to wikiskripta (wikiskripta.eu) will be rendered
```

Custom Installation

The application is published at: http://physiome.lf1.cuni.cz/identifikace. You may publish the application within any other server.

Download Physiovalues.zip release from https://github.com/TomasKulhanek/Physiovalues/releases unpack and follow instruction.

Identification release

This folder contains release of identification system

- identifikace/ - ASP.NET 4.0 to be deployed to IIS
- worker14.05 - worker node for FMU v 1.0 preduced before 2014
- worker17.01 - worker node for FMU v 1.0 produced after 2014, for newer repaired behavior

Server

The content of the folder "identifikace" should be deployed into the directory c:\inetpub\wwwroot\identifikace
Mandatory settings for MSSQL:

- database url: localhost
- database name: restmasterservice

Optional settings:

- IDENTIFIKACE_BALANCER_DEBUG - environment variable, if set, then the debug log is appended to file ** c:\inetpub\wwwroot\identifikace\logs\simulatorbalancer.log
- IDENTIFIKACE_MATLAB_DEBUG - environment variable, if set, then matlab debug log is appended to file ** c:\inetpub\wwwroot\identifikace\logs\matlab-{date}.log

Workers

Worker can be deployed in any other machine, e.g. on the same server for low network latency, or on another server, whose IP is accessible from the server, as worker registers it's URL to the server for bilateral communication.

runhosts1.cmd script is example.

For debuging purpose run 1 worker. For performance purpose launch as many workers as there are CPUs, each with dedicated port number.