Artifact Evaluation: Quantifying System Conformance using the Skorokhod Metric

Jyotirmoy V. Deshmukh[1], Rupak Majumdar[2], and Vinayak S. Prabhu[2]

[1] Toyota Technical Center; [2] MPI-SWS
jyotirmoy.deshmukh@tema.toyota.com {rupak,vinayak}@mpi-sws.org

Abstract. This document describes how to use the tool for checking system conformance between two Simulink® models using the Skorokhod metric. The tool is a Matlab® toolbox that utilizes a Nelder-Mead optimizer-in-the-loop to find an input trace for the two Simulink models that maximizes the Skorokhod distance between the corresponding output traces. The actual program to monitor whether two given output traces are within bounded Skorokhod distance is written in C++and is integrated with the Matlab® toolbox using the 'mex' functionality within Matlab. The tool user is not expected to know these details, but is assumed to have some familiarity with Matlab® and Simulink®.

1 Introduction

This tool consists of two parts:

1. A C++ implementation used for monitoring if two signal traces are within bounded Skorokhod distance.
2. A Matlab® toolbox that uses Nelder-Mead optimization to find an input to two given Simulink® models that maximizes the Skorokhod distance between the outputs of the two models.

2 Requirements

The experiments reported in the paper were performed on a Matlab®/Simulink® setup on a computer with the Windows 7 operating system[1].

1. Matlab®, Simulink®. Preferred Version: R2013b, for 32-bit Windows [2] 7. Please refer to `http://www.mathworks.com` for additional details.

[1] The following instructions can be suitably modified to run on a Linux based computer as long as the correct compiler options are provided to the `mex -setup` command. We have not used the Ubuntu virtual machine provided by the CAV AEC as our experiments were performed in a Windows environment.

[2] Please note that our implementation has been tested on the R2013b version. As Matlab® and Simulink® features keep changing with every release, our tool is not guaranteed to work in the same fashion with other releases. We have tested it against a couple of different releases (R2013a, R2012b), and it did seem to work as expected for these versions.

2. Microsoft Visual C++ compiler. Our version: compiler included with Microsoft SDK 7.1, installed under Visual Studio 10.0.

3 Installation

The following steps assume that the user has setup mex within Matlab. This can be done by the running the command: `mex -setup`.

1. Unzip the file `Package.zip`. This should create a directory named `Package`.
2. Open Matlab and within Matlab change folder to `path-to/Package/`.
3. Run `Install.m`. If there are no errors, the script should print `setup completed.` to the screen. If there are errors, please debug your mex setup.

4 Running examples from the paper

Change directory to the example that you wish to run. Examples are inside the directory named `examples` within the `Package` directory.

1. Water tank Example: Change directory to `examples/Example0`. Run the script `RunExample0.m`. The script should print a superset of the data presented in Table 1 in the paper.
2. LQR-based pitch controller: Change directory to `examples/Example2`. Run the script `RunExample2.m`. The script should produce the results indicated in Table 2 in the paper[3].
3. Air-Fuel Ratio Controller: Change directory to `examples/Example1`. Run the script `RunExample1.m`. The script should produce the results indicated in Table 3 in the paper.
4. Engine Timing Model: Change directory to `examples/Example3`. Run the script `RunExample3.m`. The script should produce the figure shown in Fig. 2 in the paper.

5 Information about the tool and runnning it on your own examples

1. Please use the template script `RunExampleTemplate.m` provided in the top-level directory. Please populate the script with information specific to your model!
2. More information about running the tool, the options that can be tweaked, and details on input parameterization, output scaling (in time and value domains), choice of window size can be found in the comments within the script `RunExampleTemplate.m`.

[3] Please note that the experiments reported in the paper were performed on a 64-bit Windows 7 machine with 32 GB RAM. The authors have noticed up to 20% difference in run-times when the experiments were performed on a less powerful machine, e.g. a 32-bit Windows 7 laptop with 4 GB RAM.