

REPEATABILITY EVALUATION INSTRUCTIONS

TRUST: STABILITY AND SAFETY CONTROLLER SYNTHESIS FOR UNKNOWN DYNAMICAL MODELS USING A SINGLE TRAJECTORY

JAMIE GARDNER, BEN WOODING, AMY NEJATI, AND ABOLFAZL LAVAEI

SCHOOL OF COMPUTING, NEWCASTLE UNIVERSITY, UNITED KINGDOM

{J.GARDNER3,BEN.WOODING,AMY.NEJATI,ABOLFAZL.LAVAEI}@NEWCASTLE.AC.UK

Welcome to the TRUST repeatability evaluation instructions. This document explains how to reproduce the results presented in the paper “TRUST: Stability and Safety Controller Synthesis for Unknown Dynamical Models Using a Single Trajectory”. The GitHub repository with the latest updates can be found at:

<https://github.com/thatgardnerone/TRUST>

Documentation and details about the tool can be found in this GitHub repository and are summarized in Sections 1-5 of this document. *These repeatability evaluation instructions will replicate the results of Table 1 and Table 2 of the paper.* Note that the figures are not part of the repeatability as their results are already included in the tables. We have personally got these results running for Apple Silicon, Linux (Ubuntu), and Windows.

We expect the repeatability evaluation committee to read Sections 1-4 of this guide covering the system requirements, installation, smoke test, and full evaluation of the results of the paper. Section 5 details the graphic user interface (GUI) of the tool and how to extend the tool for new case studies.

TRUST also offers a user-friendly and reactive web application, enabling seamless interaction with the tool. It is accessible directly at <https://trust.tgo.dev>, allowing users to work with the application via the web without requiring downloads or installations (see Section 5.2 for instructions on running case studies using the GUI). Note that while this hosted version is convenient, *the hardware is not as performant as a local setup due to containing only 2 CPU cores.* For the purposes of repeatability without requiring network access, we provide the following instruction for local setups of TRUST.

1. SYSTEM REQUIREMENTS FOR REPEATABILITY EVALUATION

A Workstation with Docker. We have tailored our repeatability evaluation instructions for Docker, with instructions on installation listed in Section 2.

All of our examples have been run on an M3 Max MacBook Pro with 36 GB memory. Machines with fewer CPU cores and less memory should still be able to run TRUST smoothly, but the results may take longer to generate. The largest example (**dtNPS Lorenz Attractor**) requires up to 0.5GB RAM. The slowest example (**hi_ord_8**) takes around 16 minutes to run on the M3 Max MacBook Pro.

MOSEK License. All results were obtained using the MOSEK solver, which requires a MOSEK license. Academic users can obtain a free license, and a free trial is also available if needed. You can get a MOSEK license at <https://www.mosek.com/license/request/?i=acp>. You will receive the license file via email, which you must download and save it on your machine to load into the GUI for running all cases.

2. INSTALLATION

2.1. Install Docker. Installing Docker on most operating systems is straightforward by following their documentation at <https://docs.docker.com/get-docker/>. Please follow the relevant link in their documentation to install Docker for your operating system.

2.2. Load and Run TRUST. *The Repeatability Evaluation Package (REP) consists of a Docker image `trust.tar.gz` and a ZIP archive `cases.zip` containing the data and supplementary inputs to the tool. Once these are downloaded in step (1), network access is not required for installation or running TRUST.*

- (1) The collected data and supplementary inputs for all of the case studies are provided as a ZIP archive, `cases.zip`. This archive can be downloaded as part of the latest release on GitHub at <https://github.com/thatgardnerone/TRUST/releases/latest/download/cases.zip>. You will need to unzip the archive to access each of the case study files within, which you can typically do by right-clicking the `cases.zip` file and choosing “Open” or “Extract” depending on your operating system.

TRUST is provided as a Docker image archive. This archive can be downloaded for the corresponding architecture of your machine as part of the latest release on GitHub.

For Apple Silicon, use `trust.arm64.tar.gz`.

For Intel/AMD 64, use `trust.amd64.tar.gz`.

To install the Docker image, you do not need to extract the image archive. Simply run the following commands in your terminal to load and run the tool. *The following steps do not require network access.*

- (2) From a terminal, run the following commands in order, selecting either `arm64` or `amd64` based on your machine:

```
cd ~/Downloads (or change ~/Downloads to the directory where you saved the files)
docker load -i trust.{arm64|amd64}.tar.gz
docker run --name trust -d -p 8000:8000 --rm trust:{arm64|amd64}
```

- (3) Once the installation is complete, you can access the tool by navigating to `http://localhost:8000` in your web browser; *we highlight again this does not require network access and is a locally hosted webpage.*
- (4) To stop the tool, you can run the following command:
`docker container stop trust`

Once all the previous installation instructions have been run, the tool should look like the labeled screenshot in Fig. 2. *We note that **TRUST** supports dark mode, so the GUI will reflect which mode your machine is in. Also clicking calculate without entering any values should prompt the user to input valid data.*

3. SMOKE TEST

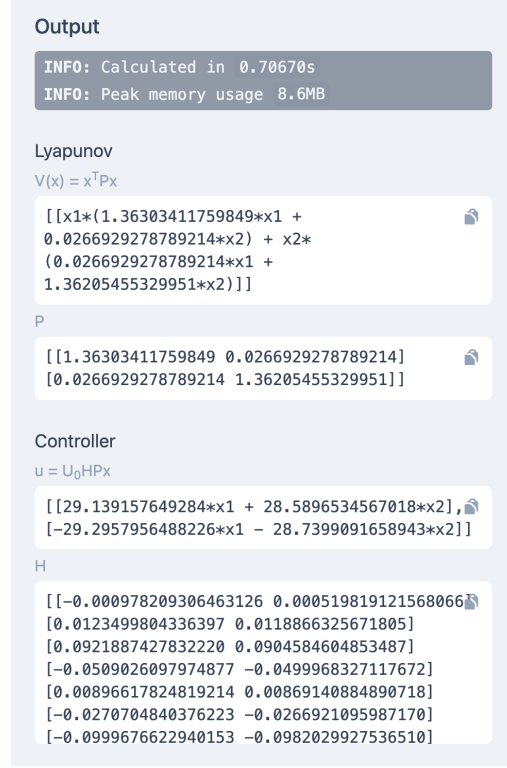


FIGURE 1. TRUST dt-LS DC Motor results

For the smoke test, we suggest to enter data for a discrete-time linear system (dt-LS) stability case, the **dt-LS DC Motor**.

This can be easily input in the GUI by leaving the default selections for the left-hand menu specifications, and by opening the folder **storage/cases/dtLS.dc_motor** and copying and pasting the values for **X0**, **U0** and **X1** into the respective locations of the GUI. *Note that each folder contains different datasets for the stability and safety problems, as indicated within the folders.* For this smoke test, the dataset corresponding to stability should be used. In addition, the MOSEK license should be uploaded in the respective section. Once the values have been added, click “Calculate” and the results should be displayed within a moment.

The results for the **dt-LS DC Motor** will appear in the right-hand results column. These values should roughly match the results that can be seen in the screenshot in Figure 1. The smoke test has failed if an error is thrown instead.

In general, there may be some slight variations between results computed on different machines due to hardware differences and how the optimizer solves the problem.

TABLE 1. Data-driven design of CBCs and safety controllers. The symbol n denotes the dimension, while T represents the number of collected samples. All cases were run on a MacBook Pro (Apple M3 Max with 36 GB RAM).

Experiment Name	System	n	T	γ	λ	Time (s)	Memory (MB)
Lotka-Volterra Predator-Prey Model	ct-NPS	2	12	0.11	0.14	50.92	45.0
Van der Pol Oscillator	ct-NPS	2	15	5.73	14.26	345.79	105.5
DC Motor	ct-LS	2	15	3.11	3.43	0.53	1.2
Room Temperature System 1	ct-LS	2	15	631.02	1838.62	0.34	0.8
Two Tank System	ct-LS	2	12	5.48	6.17	0.35	10.3
High Order 4	ct-LS	4	16	13,910.89	14,374.75	5.98	6.2
High Order 6	ct-LS	6	16	20,072.83	20,600.83	91.06	17.0
High Order 8	ct-LS	8	20	47,437.80	66,261.32	1008.80	44.0
Lotka-Volterra Predator Prey	dt-NPS	2	12	0.46	0.57	59.86	58.3
Lorenz Attractor	dt-NPS	3	12	1,052.68	3,931.40	948.52	323.7
DC Motor	dt-LS	2	15	0.64	0.70	0.61	1.4
Room Temperature System 1	dt-LS	2	15	8.87	9.07	0.77	1.8
Room Temperature System 2	dt-LS	3	15	8.02	12.00	2.58	9.5
Two Tank System	dt-LS	2	8	1.80	3.17	1.05	2.3
High Order 4	dt-LS	4	16	261.63	264.94	5.59	5.6
High Order 6	dt-LS	6	16	22.50	23.14	85.01	16.0
High Order 8	dt-LS	8	20	21.93	30.46	1003.39	48.5

4. FULL EVALUATION

We now provide the details for the evaluation of Tables 1-2.

Each case study is stored with its corresponding name in the `storage/cases` folder, and both the safety and stability datasets within each folder are saved as `*.csv` files. Additionally, a `supplementary_inputs.txt` file is included within each case as needed, which contains descriptions of the state space, initial set and all unsafe sets (*if the problem is safety*), the values for the monomials (*if the problem is nonlinear polynomial*), and the matrix $\theta(x)$ (*if the problem is nonlinear polynomial but in the discrete-time setting*), as required.

If the problem is safety, there may be variations in the computation of level sets λ and γ , depending on the machine that solves the problem. However, as long as the results show that $\gamma < \lambda$ (the theoretical condition stated in the paper), a valid barrier certificate is considered to have been found, even if the values of the level sets and computation times reported in the table vary slightly.

To run the case studies, the following process can be carried out:

TABLE 2. Data-driven design of CLFs and stability controllers. The symbol n denotes the dimension, while T represents the number of collected samples. All cases were run on a MacBook Pro (Apple M3 Max with 36 GB RAM).

Experiment Name	System	n	T	Time (s)	Memory (MB)
Lotka-Volterra Predator-Prey Model	ct-NPS	2	12	49.76	33.8
Van der Pol Oscillator	ct-NPS	2	15	347.67	139.7
DC Motor	ct-LS	2	15	0.02	0.1
Room Temperature System 1	ct-LS	2	15	0.05	0.1
Two Tank System	ct-LS	2	12	0.06	0.1
High Order 4	ct-LS	4	16	0.10	0.2
High Order 6	ct-LS	6	16	0.17	0.4
High Order 8	ct-LS	8	16	0.35	0.8
Academic System	dt-NPS	2	12	44.72	43.9
Lorenz Attractor	dt-NPS	3	12	720.81	251
DC Motor	dt-LS	2	15	0.06	0.2
Room Temperature System 1	dt-LS	2	15	0.05	0.1
Room Temperature System 2	dt-LS	3	15	0.08	0.2
Two Tank System	dt-LS	2	8	0.06	0.1
High Order 4	dt-LS	4	16	0.15	0.3
High Order 6	dt-LS	6	16	0.29	0.6
High Order 8	dt-LS	8	16	0.49	1.2

4.1. Problem specification.

Upload your MOSEK license in the upper left section.

In the left column of the tool, select the correct options for the respective case study that you wish to run. Choose “Stability” when finding a Control Lyapunov Function (CLF) or “Safety” when finding a Control Barrier Certificate (CBC). For systems with prefix “dt” select “Discrete-Time” and for prefix “ct” select “Continuous-Time”. For systems with suffix “LS” choose “Linear” and suffix “NPS” choose “Non-Linear Polynomial”.

For example, for the safety experiment with the name “Lotka-Volterra Predator-Prey Model” for a ct-NPS system, select “Continuous-Time”, “Nonlinear Polynomial”, and “Safety”; open up the corresponding `storage/cases/ctNPS_lotka_volterra_predator_prey_model` folder, then open the three relevant datasets, i.e. `safety_X0T`, `safety_U0T` and `safety_X1T`, and copy and paste the respective values into the corresponding input in the tool. Alternatively, you may click “Upload” and find the file manually to add it to the tool, or, conveniently, once you click the “Upload” tab, simply drag and drop the relevant file from your PC to the upload box for it to be automatically uploaded.

4.2. Set Descriptions, Monomials and $\Theta(x)$. In the middle column, the GUI will dynamically display inputs for monomials, the matrix $\theta(x)$, the state space, initial set and all unsafe sets, as required.

Continuing with the safety example “Lotka-Volterra Predator-Prey Model” for a ct-NPS system, in the same folder `storage/cases/ctNPS_lotka_volterra_predator_prey_model`, open the `supplementary_inputs.txt`, and individually copy each of the input values for the monomials, state space, initial set and unsafe sets.

You will notice that the input for the matrix $\theta(x)$ will not be presented for this example (this is only for the discrete-time nonlinear setting), as the GUI will automatically display required inputs based on your selection in the previous column.

4.3. Results. The right column will show the output of the tool. Initially, a status indicator will show that the GUI is connected to the back-end system. Whilst a solution is being calculated, the status indicator will change to let you know that a calculation is in progress. Once the results have been calculated, they will be displayed in this right column.

To finish the ct-NPS system example for “Lotka-Volterra Predator-Prey Model”, you should see the *time taken and memory used* by the tool to compute the barrier and synthesize a controller, all of which will be displayed as both the resulting barrier and controller function descriptions, their underlying matrices, and the γ and λ level sets. Compare the time, memory and level set values (rounded to two decimal places) with the matching record in Table 1.

4.4. Safety and Stability Differences. Table 1 refers exclusively to CBC results and Table 2 with CLF results. In general for many stability cases, the middle column will need no entries, with the exception of selecting monomials and $\Theta(x)$ for the nonlinear polynomial systems.

5. GRAPHIC USER INTERFACE (GUI)

The screenshot shows the TRUST GUI interface with the following sections and annotations:

- 1**: Upload MOSEK License section.
- 2**: Class selection (Discrete-Time, Continuous-Time).
- 3**: Model selection (Linear, Non-Linear Polynomial).
- 4**: Specification selection (Stability, Safety).
- 5**: Add X0 section (Manual, Upload).
- 6**: TRUST logo and description.
- 7**: Dimensions section (Dimensions: 2).
- 8**: Monomials section (Monomials: x_1, x_2, x_1^2, x_1^3).
- 9**: Matrix $\theta(x)$ section (Matrix: $\begin{bmatrix} 1, 0 \\ 0, 1 \\ x_1, 0 \\ x_1^2, 0 \end{bmatrix}$).
- 10**: State Space section (Lower and upper bounds: $x_1: -1, 1$; $x_2: -1, 1$).
- 11**: Initial Set section (Lower and upper bounds: $x_1: 0.1, 0.5$; $x_2: 0.1, 0.5$).
- 12**: Unsafe Sets section (Unsafe Set 1: $x_1: 0.1, 0.5$).
- 13**: Output section (INFO: Calculated in 98.82941s; INFO: Peak memory usage 70.0MB).
- 14**: Barrier section (Barrier: $B(x) = x^T P x$).
- 15**: Controller section (Controller: $u = U_{gh}(x) P x$).
- 16**: Level Sets section (Level Sets: $y: 2.495204960232225$; $\lambda: 2.632047429391349$).
- 17**: Calculate (Ctrl+Enter) button.

FIGURE 2. TRUST GUI with numbered annotations indicating respective sections.

5.1. Running New Case Studies with TRUST. We have aimed to make using TRUST for your own examples as straightforward as possible. This can be done by following same process as you followed to evaluate our case studies.

TRUST accepts input datasets that include a *persistently excited* data trajectory, which should be copy-and-pasted into the “Manual” input fields or uploaded in one of the following common file formats: **.csv*, **.txt*, or **.json*. *It will be the responsibility of the user to generate meaningful data and inputs for any new case studies for testing. TRUST automatically verifies that the data is persistently excited by checking a rank condition on the data collection horizon. If this condition is not met, an error message will be returned to the user (see Error Handling Section 5.2.2, item (iv)).*

5.2. Graphical User Interface (GUI) in TRUST. To maximize accessibility and ensure a highly user-friendly experience, TRUST utilizes a Model-View-Controller (MVC) architecture with an integrated GUI. The GUI enhances usability by abstracting the underlying technical complexities, allowing users to construct either a CLF or CBC using data through a *reactive push-button* interface. While TRUST provides GUIs for all four system classes, each focusing on either stability or safety properties, we only depict it for dt-NPS, as

it offers the most comprehensive inputs to the GUI (see Figure 2). Labels in this figure are referenced with $\langle \cdot \rangle$.

As noted earlier, TRUST relies on MOSEK, which requires a license. Users must upload their license to run the tool $\langle 1 \rangle$. Note that the license is *not* stored; it is only used on a per-session basis, meaning users must re-upload their license for each new session. TRUST supports both discrete-time and continuous-time system classes $\langle 2 \rangle$ as well as linear and nonlinear polynomial models $\langle 3 \rangle$. It can design controllers to enforce either stability or safety properties $\langle 4 \rangle$. Users may upload datasets for $\mathcal{X}_0, \mathcal{U}_0$, and \mathcal{X}_1 or enter them manually $\langle 5 \rangle$. A link to the tool’s source code is provided in the footer $\langle 6 \rangle$. The GUI is dynamically rendered, hiding unnecessary elements to avoid user confusion.

The system dimension is automatically detected based on the shape of \mathcal{X}_0 $\langle 7 \rangle$. For nonlinear polynomial systems, users must specify the monomials $\mathcal{M}(x)$ $\langle 8 \rangle$, separated by semicolons. Note that SymPy notation requires “*” for *multiplication* and “**” for *exponents*. For dt-NPS, users also need to define their desired $\Theta(x)$ (cf. (5.1)) in $\langle 9 \rangle$. $\Theta(x)$ is an $(N \times n)$ matrix polynomial such that

$$\mathcal{M}(x) = \Theta(x)x, \quad (5.1)$$

If desired, users can enable an “autofill” option for $\Theta(x)$ in $\langle 9 \rangle$, allowing TRUST to automatically compute its value. If the property is safety, TRUST uses hypercubes to define the state space $\langle 10 \rangle$, initial sets $\langle 11 \rangle$, and one or more unsafe sets $\langle 12 \rangle$, with both lower and upper bounds required for each dimension. For stability properties, items $\langle 10 \rangle - \langle 12 \rangle$ are hidden.

Finally, the output is displayed in the far-right section, showing a brief summary with the total computation time and peak memory usage $\langle 13 \rangle$. If TRUST fails to find a valid solution, this is prominently displayed in red in the “INFO” section, and no other output is shown. For successful runs, the CLF or CBC is designed, along with the underlying matrix P $\langle 14 \rangle$. This is followed by the controller, including the matrix H $\langle 15 \rangle$. For the CBC, the designed level sets are also displayed $\langle 16 \rangle$. All results are formatted in Python syntax.

To run the tool, the user clicks the “Calculate” button to initiate computation or “Reset” to clear their input $\langle 17 \rangle$. The keyboard shortcut **Cmd+Enter** or **Ctrl+Enter**, depending on the operating system, can also start the computation.

Note that the choice of $\Theta(x)$ satisfying the equality condition in (5.1) is not unique, and different choices may influence the proposed conditions in the underlying theory. To accommodate this flexibility, we allow the user to input this $\Theta(x)$ as an additional parameter (see $\langle 9 \rangle$ in Figure 2). Alternatively, the user can enable the autofill option, prompting the tool to solve (5.1) and design $\Theta(x)$ automatically. We have provided the choice of $\Theta(x)$ used for solving the dt-NPS problems in the corresponding folders (`supplementary_inputs.txt`).

5.2.1. *User inputs.* The user is expected to input the required parameters as detailed previously. The GUI will automatically prompt the user with an error message if any required field is left empty. The GUI will also show relevant error messages where necessary. Error handling is discussed in detail in the following section.

5.2.2. *Error Handling.* TRUST is developed as a responsive and reactive Python Flask web application, offering an *intuitive, user-friendly* interface that allows seamless interaction. If a user error occurs, TRUST provides responsive error messages to guide the user in correcting their input. Listed below are some common errors that may be returned to the user:

- (i) For an invalid “state space”, “initial set” or “unsafe set(s)”: *“Provided spaces are not valid. Please provide valid lower and upper bounds”*.
- (ii) For an invalid shape of $\Theta(x)$: *“Theta.x should be of shape (N, n)”*.
- (iii) If monomials are provided with commas: *“Monomial terms should be split by semicolon”*; if they are not suitable for the set dimensions: *“Monomials must be in terms of x_1 (to x_n)”*; if some unspecified error has occurred with the monomials: *“Invalid monomial terms”*.
- (iv) If the rank condition is not met for nonlinear polynomial systems: *“The number of samples, T , must be greater than the number of monomial terms, N ”, or “The N_0 data is not full row-rank”*, depending on which part of the rank condition failed. Similarly for linear systems: *“The number of samples, T , must be greater than the number of states, n ”, or “The X_0 data is not full row-rank”*, again depending on which part of the rank condition failed.
- (v) If data files are uploaded with an invalid format: *“Unable to parse uploaded file(s)”*.
- (vi) If the MOSEK solver cannot find a solution for the given values: *“Solution Failure”*, with a dynamic error description provided by the solver. If the MOSEK solver did find a solution but the solution does not contain an SOS decomposition: *“No SOS decomposition found”* with a dynamic error description. Similarly, if the solution does not contain valid SOS constraints: *“Constraints are not sum-of-squares”*.
- (vii) Any other errors in the tool will be caught with the generic error message: *“An unknown error occurred”* and a brief description that can be reported.

6. UNINSTALL TRUST

If you no longer wish to have TRUST installed on your machine, you may run the following commands in the terminal to uninstall the tool.

```
cd ~/TRUST
```

```
docker compose down
```