OpenIPDM

OpenIPDM V1.0 reference manual

Zachary Hamida and James-A. Goulet
Polytechnique Montreal

July 16, 2021

Contents

1 Getting started

1.1 About OpenIPDM

OpenIPDM is a MATLAB open-source platform that stands for **i**nfrastructures **p**robablstic **d**eterioration **m**odel. This software is developed to perform analyses on a network-scale visual inspection data, while accounting for the uncertainty associated with each inspector. The main application window in OpenIPDM enables assessing the structural deterioration behaviour and the effect of interventions at different levels starting from the structural element level up to the network level. OpenIPDM also include several toolboxes that facilitate performing verification and validation analyses on visual inspection data, in addition to learning model parameters. Furthermore, OpenIPDM has the capacity to handle missing data such as, missing interventions or missing structural attributes. OpenIPDM is available for download from GitHub at https://github.com/CivML-PolyMtl/OpenIPDM.

Keywords: visual inspection, state-space models, kernel regression, deterioration model, bridge network, Bayesian analysis.

1.2 Installing OpenIPDM

The following instructions show how to download and setup OpenIPDM on your local machine for direct use, testing, and development purposes.

1.2.1 Prerequisites

MATLAB (version 2020b or higher) installed on Mac OS X or Windows.

The MATLAB *Statistics and Machine Learning Toolbox* is required.

1.2.2 Installation

1. If an older OpenIPDM version is already installed, it is recommended to remove it from your MATLAB and perform a fresh installation for the newer version.
2. Download and extract the ZIP file from https://github.com/CivML-PolyMtl/OpenIPDM (or clone the git repository) in your working directory.
3. The working directory includes the following folders:
    - Scripts
    - Tools
    - Parameters
    - Network Data
    - Figures
    - ExtractedData
    - Help

The folder *Scripts* includes all the scripts that run in the background and are directly associated with the main window of OpenIPDM interface. The scripts include the functions responsible for performing and displaying the analyses as well as filtering the data. The folder *Tools* includes the scripts and applications for all the toolboxes that are accessible by the main menu bar. The folder *Parameters* contain the files associated with the pre-trained model parameters. The folders *Network Data* and *ExtractedData* contain the pre-processed network-scale datasets that are extracted to improve access to information by OpenIPDM. Finally, the folder *Figures* includes the ".tex" files extracted by the user from OpenIPDM and the folder *Help* contains the documentations of the software functionality.

1.3 Starting OpenIPDM

There are two possible options for starting the OpenIPDM software,

- Double click *OpenIPDM.mlapp* file to start MATLAB *App Designer* and from the top ribbon in *App Designer*, click *Run*
- Install OpenIPDM as MATLAB application by double clicking on *OpenIPDM.mlappinstall*, thereafter, OpenIPDM can be found in the *APPS* tab in MATLAB.

After starting OpenIPDM, the main user interface will open along with a message box to load the database. Note that the message box will not show up, if pre-processed data already exist in the folder *Network Data*.

1.4 Demo

In this demo, the two possible starting modes for OpenIPDM are shown. The first mode (or mode 1) is when OpenIPDM is being started without any pre-processed inspections database. In this mode, OpenIPDM will start as shown in the Figure 1. In the loading database

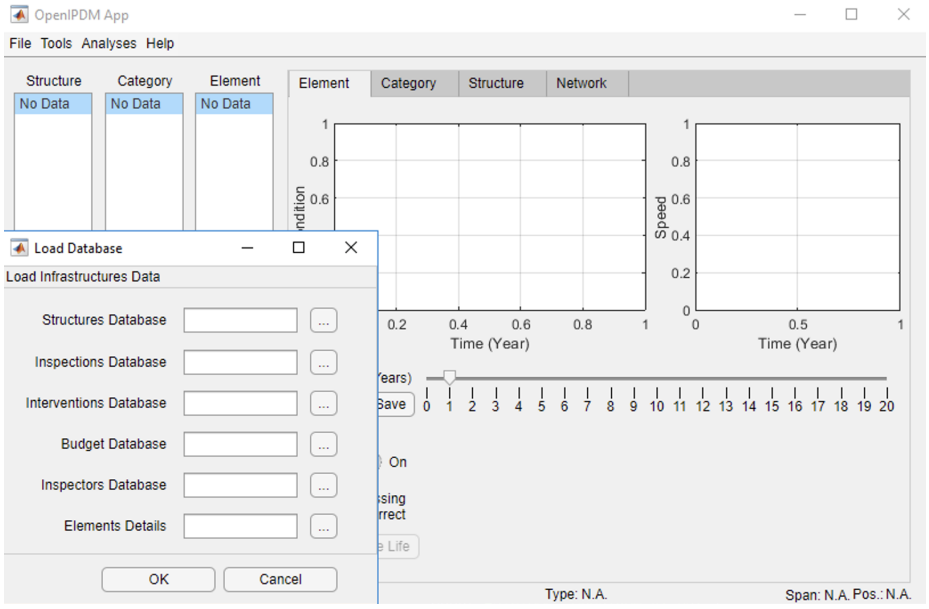


Figure 1: Starting OpenIPDM without pre-processed data (Mode 1).

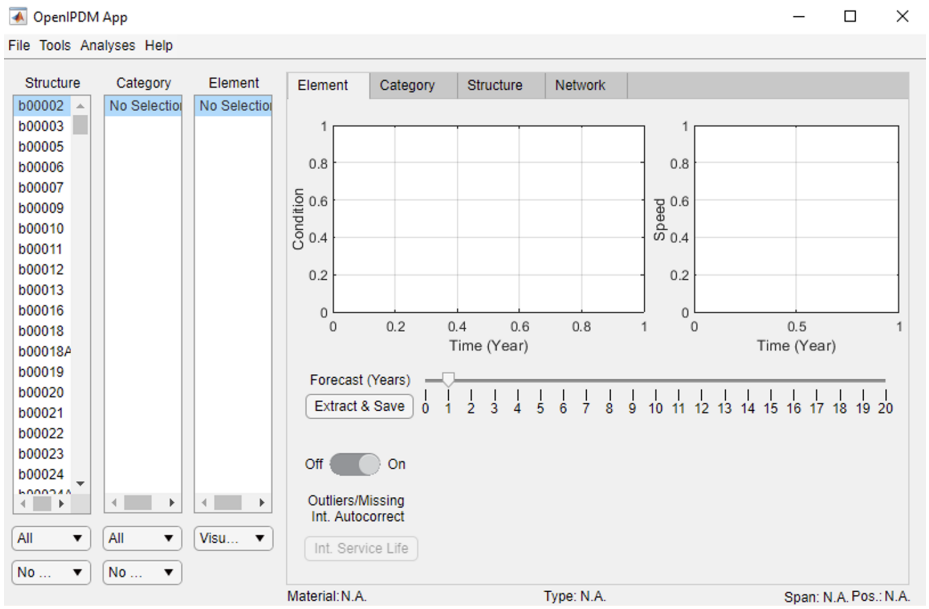


Figure 2: Starting OpenIPDM after pre-processing the database (Mode 2).

interface, there are 6 input boxes for each of the database files. The required file format is ".csv", and each file represents a table of information about the infrastructures. The required files are,

- *Structures Database*: includes information about bridges such as the bridge's identifying number, location and other bridge level information.
- *Inspections Database*: includes the inspection information such as the inspection date and results, in addition to the structural element type and material.
- *Interventions Database* involves information about the interventions such as the intervention date and type as well as the structural element ID.
- *Budget Database*: includes information about records of investments made on each bridge over time.
- *Inspectors Database*: contains information about the inspector responsible for the intervention report.
- *Elements Details*: includes information associated with the characteristics of structural elements, such as the element quantity.

After loading each of the ".csv" files, a selection menu will show a list of columns to be imported. The mandatory columns are already pre-selected for each case by the application. Nonetheless, further details about the mandatory columns will be discussed in the Section 2. It should be noted that working with OpenIPDM in mode 1 is slow and not recommended for large-scale analyses.

The second mode to start OpenIPDM can be accessed by pre-processing the database, which allows a faster navigation throughout the analyses. The pre-processing step can be done after having loaded the database files for the first time. In order to pre-process the database,

go to *File* then *Pre-process Data*. The pre-processing can take a considerable amount of time (e.g., days), however, it is only required once. After the pre-processing is done, the structures will load automatically every time the platform is started. Note that when the platform is in the second mode, all the structures' IDs starts with the letter "b".

1.5 Main Window - OpenIPDM

In this section, the user interface (UI) components of the main window are described in details. Figure 3 shows these components with numbered labels 1-9. From Figure 3, the user

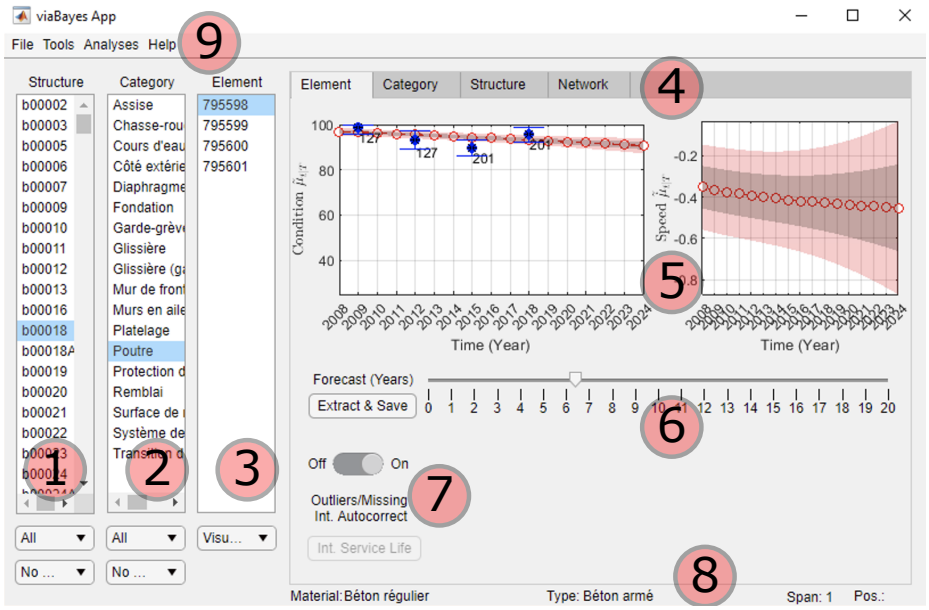


Figure 3: Main window of OpenIPDM with numbered labels.

interface component under each label (1-9), are listed below,

1. Label 1 includes the structures list where the user can select a bridge from this list or filter this list according to the options provided in the dropdown menu. This includes filtering bridges according to the type of structure, municipality and bridge status.

2. Label 2 includes a list of structural categories that are inspected within the bridge (e.g., beams). The user is also capable of filtering this list according to the components being related directly to the serviceability of the bridge or the safety of the bridge.

3. Label 3 includes a list of structural elements under the selected structural category. The user can filter this list according to the evaluation criteria (i.e., visual inspections (VI) v.s. behaviour metric (CEC)).

4. The tabs under Label 4 include the element tab, the category tab, the structure tab and the network tab, with each tab showing the results of analyses at each of its respective level.

5. The figures show the analyses results for the deterioration condition (on the left) and the deterioration speed (on the right).

6. The timeline determines the duration of the forecast after the last recorded inspection, with the button *Extract & Save* to extract the figures into a MATLAB figure window and export a ".tex" file of the same figure.

7. The section under Label 7 includes autocorrect functions to handle missing data (e.g., missing intervention report). In addition, the button *Int. Service Life* shows the results of the expected time until the structural element return to the state prior to the intervention. Note that this button is only active for structural elements with an intervention event.

8. The information bar under Label 8 includes information about the selected structural element such as the material, the position and type.

9. The context menu bar include different sections for accessing toolboxes associated with training the model, verifying and validating the deterioration framework.

2 Inputs and outputs

This section describes the file format as well as the essential information from the inputs and outputs of OpenIPDM.

2.1 Inputs - Visual Inspections Database

The main input file format for OpenIPDM is comma-separated values file (or ".csv"). OpenIPDM is developed in accordance with the visual inspection system and the database of the Ministry of Transportation of Quebec (MTQ) [1]. The network-scale visual inspections database is composed of multiple tables, which are originally embedded in a single Microsoft Access database. These tables are discussed in the following subsections.

2.1.1 Structures Database

This database include general information about each of the infrastructures involved in the network-scale visual inspections system. The mandatory information to be provided from this table are,

- Structure's ID which represents a unique identifier for the structure.
- Type of the structure which correspond to predefined structural categories in the MTQ manual of inspections [1].
- The municipality which represents the agency responsible of the structure.
- The year of construction of the bridge.
- The location represented by the longitude and latitude.
- The status of the bridge which mainly determine if the bridge has been demolished or is not under the supervision of MTQ.
- The structural attributes which include, the length of the bridge, the length of the slab, number of lanes and the annual average of daily traffic.

2.1.2 Element Inspections Database

The inspections database include information about structural elements that have been inspected over time. The mandatory information to be extracted from this database are,

- Structure's ID which represents a unique identifier for the structure.
- Inspection date.
- Span number and the element-number/position within the span, which help create an alternative for the unique identifier of each structural element.
- Element category (e.g., beam).
- Element material and type which provide specific information about the structural element.

- The visual inspection data represented by $A\%, B\%, C\%, D\%$ and the behavioural assessment (CEC) [1].
- The unique element identifier.
- The unique inspection identifier, which help in identifying the inspector responsible for the inspection results.

2.1.3 Realized Interventions

This database includes information about the interventions that have happened on each structural element. The mandatory information to be imported from this database are,

- Structure's ID which represents a unique identifier for the structure.
- Element category (e.g., beam).
- The unique element identifier.
- The starting date of the intervention.
- The type of the intervention represented by an intervention code [1].

2.1.4 Planned Interventions

The planned interventions database includes information about interventions that would take place on the bridge. The content of this database are not essential for the deterioration analyses, however, it can be useful for the planning part. This is because the costs of interventions actions are reported in this database based on historical cost data. The main content in this database are,

- Structure's ID which represents a unique identifier for the structure.
- Element category (e.g., beam).
- Action unit cost.
- The type of the intervention represented by an intervention code [1].

2.1.5 Responsible Engineer Database

This database include information about the inspectors responsible for the inspection report about the structural element. The mandatory information to be imported from this database are,

- The unique inspection identifier.
- The unique inspector identifier.

2.1.6 Element Details Database

The element details database include qualitative information about the structural elements. The mandatory information to be extracted from this database are,

- Structure's ID which represents a unique identifier for the structure.
- Span number and the element-number/position within the span, which help create an alternative for the unique identifier of each structural element.
- Element category (e.g., beam).
- Element material and type which are interchangeable depending on the structural elements category. This is because some elements has the material information mentioned in the element type section.
- The unique element identifier.
- Element quantity, which can have different unites depending on the structural element category.

Each of the aforementioned tables can be exported from the Access database as a comma-separated values file (or ".csv" file), so it can be utilized by OpenIPDM. It should be noted that since the MTQ database include French language characters, it is recommended to use the "UTF-8" formatting for the ".csv" file. Furthermore, a separate database file is provided by MTQ which contain the annual budget associated with each bridge. The mandatory information to be imported from the budget file are the investments starting and ending dates as well as the structure's ID.

2.2 Inputs - Model Parameters

The model parameters file is formatted as a MATLAB cell-array ".mat" file, with multiple cell-array variables. These variables include the following information.

2.2.1 Deterioration Model Parameters

The model parameters associated with each structural element category are organized in the MATLAB cell-array file "AllElementsParameters". This file has 10 columns which include the following information from left-to-right,

1. Name of structural element category.
2. General model parameters.
3. Inspectors' ID and parameters.
4. Regression model parameters.
5. Index of the structural attribute involved in the regression analyses.
6. Expected change in the deterioration state following an intervention.
7. The covariance associated with expected change in the deterioration state following an intervention.

8. Intervention model parameters.
9. Meta data for the selected structural attributes and element's material.
10. Transformation function parameter.

2.2.2 Missing Interventions - State

The file "StateNA_ExVar" is designated for structural elements with no information about the expected change in the deterioration state following an intervention. The first and second row in this file represent the expected change following an intervention for the primary and secondary elements, respectively. On the other hand, the third and forth rows represent the covariance associated with the aforementioned change for the primary and secondary elements respectively.

2.2.3 Missing Interventions Model - Parameters

The file "NaNParam" represents the model parameters associated with intervention framework for structural elements with no information about the expected change in the deterioration state following an intervention. The first row in this is for intervention model parameters for the primary elements while the second row is for the secondary elements.

2.3 Outputs

There are two different output information that can be displayed on OpenIPDM main interface. The first output is the state estimation associated with deterioration condition and speed for each structural element, structural category, structure and sub-network/full network. The second output is related to the service-life of an intervention or the time for a structural element to return to the deterioration state prior to an intervention.

2.3.1 Deterioration State Estimation

The deterioration state estimation can be performed on different levels as mentioned earlier. In order to perform the deterioration state estimation for structural elements, the user must select a structure from the structure list, then select a structural category and then select a structural element from the elements list. The estimation results will be displayed on the structural element tab. The forecast time can be changed using the timeline under the figures. Furthermore, the figures can be exported from the interface using the button *Extract & Save*. This button will export the figures from the interface into an independent MATLAB figure window and save a ".tex" version of the figures in the *Figures* folder. In order to perform analyses on the structural category level, the user is required to click on the *Category* tab, which will show the interface components associated with structural category analyses. The user can increase or decrease the forecast time as required using the timeline and the results can be exported using the *Extract & Save*. The *Structure* and the *Network* tabs follow the same processes, except that the user here has additional options to filter structural elements according to their role (i.e., primary elements and secondary elements). It should be noted that in all tabs, there is an option to automatically detect outliers and correct missing interventions according to the methods proposed in [2].

2.3.2 Service-Life of an Intervention

If a structural element has an intervention, the button *Int. Service Life* becomes active. Clicking the aforementioned button provides the user with an estimate for time to return to the deterioration state prior to the intervention. This estimate is described by a PDF plot as shown in Figure 4. The intervention service life analyses can be also performed at the

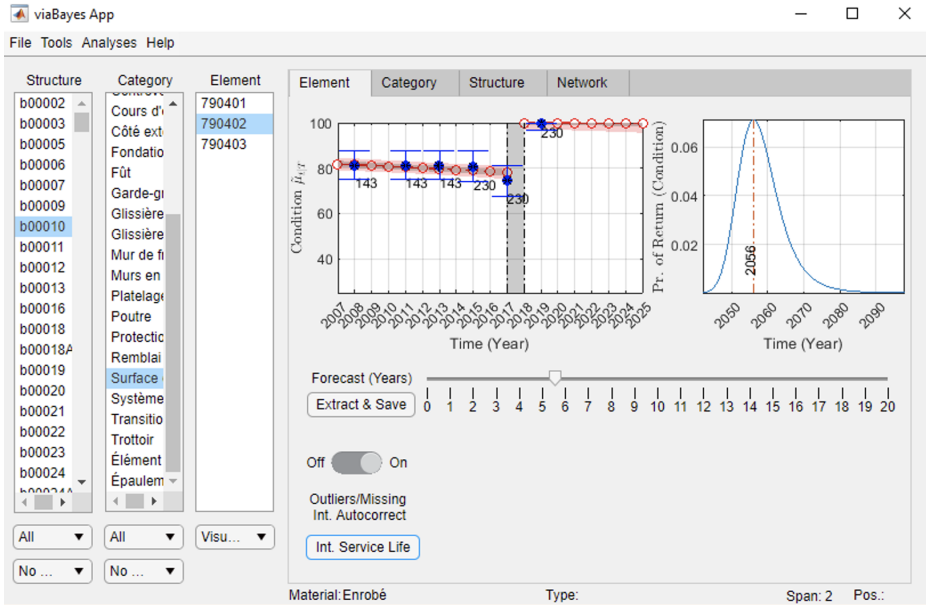


Figure 4: Example for a structural element with an intervention after clicking the button *Int Service Life*.

structural category level. This can be done by navigating to the *Category* tab, then clicking *Intervention CDF* button. This results in the software to perform the service life analyses for the selected structural category and selected intervention type using the data from either the currently selected structure which can be done by selecting the option *Local*, or the entire network with the option *Network* from the drop down menu.

3 OpenIPDM Toolboxes

This section provides general information about the main toolboxes included in OpenIPDM package. The toolboxes are developed using App Designer in MATLAB, and have a variety of purposes. The list of toolboxes described in this manual is composed of,

- Read Database.
- Generate Synthetic Data.
- Verify Synthetic Data.
- Validate Real Data.
- Single Time Series Analyses.
- Inspectors Data.
- Model Training.

The following subsections provide details about how to use each of the toolboxes above.

3.1 Read Database

This toolbox is essential to prepare the raw visual inspections database for analyses in MATLAB. The toolbox inputs are,

- Element inspections database.
- Structures database.
- Responsible engineer database.
- Realized interventions.
- Element details database (optional).
- Budget database (optional).

The columns to be imported from each database are similar to those defined in §2. The following subsections describe the UI components and inputs of each tab.

3.1.1 Load Data

This tab has two groups of UI components, the first set of components is responsible for importing the database files, while the second is responsible for filtering the imported data. Figure 5 shows the UI for the load data tab with components labeled with the numbers (1-8). The input labels 1-3 correspond to the element inspections database, structures database and the responsible engineer database, respectively. The input label 4 corresponds to the column associated with the structural element category, while the input label 5 corresponds to the structural category name. The input label 6 corresponds to extracting the inspections while excluding/including or isolating the structural elements that underwent an intervention. For the deterioration analyses, the user should exclude the structural elements that underwent an intervention, while for interventions analyses, the user should isolate the elements that

underwent an intervention. Depending on the value selected from the dropdown menu, a prompt window will appear and ask the user to load the interventions database. The columns to be imported in this case are in accordance with §2. The input labels 7-8 are optional and correspond to filtering the database according to the status of the structure. The status of the structure mainly refers to the structure being currently in service or demolished. After filling the options in the tab *Load Data*, the user can move on to the tab *Attributes*, which mainly helps the platform to identify the information that the user wants to import from the database.

3.1.2 Attributes

This UI tab helps associating the pre-defined variables in the software with the information imported by the user from the database. The input in this tab involves the construction date of each structure, the inspection dates, the structural element number within the span, the span number and the material category for each element. It should be noted that for some structural categories, the material can be specified in a different column (e.g., type of element column). Finally, the attributes tab takes also the structural attributes associated with the structure, such as the longitude and latitude information. Based on the information provided in the tabs *Load Data* & *Attributes*, the user can generate analyses-ready data by clicking on the button *Export Analyses Ready Data*. The output of this UI is a set of ".mat" files, which represent,

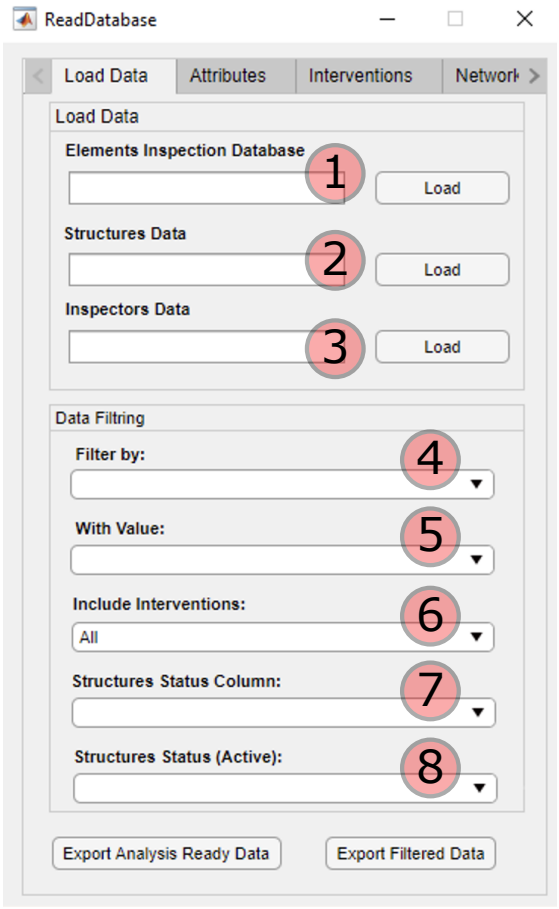


Figure 5: UI components for load data tab in read database toolbox

- SortedData_: a MATLAB cell-array variable that contain all the inspections and structural attributes.
- StructrualAttributesLabels_: meta file which contain the labels of the structural attributes selected by the user.
- ElementsandStructuresData_: supplementary file that shows all the extracted data in single table.
- StructuresID: meta file for the numerical auto generated structure ID.

The inspections and structural attributes are stored in a MATLAB cell-array variable which is organized as shown in Table 1. Each array `T` × `V` in the cell-array is composed of `T`

Table 1: Data structure for the MATLAB cell-array variable that stores the data of elements and structures.

	e_1	...	$e_{\mathtt{E}}$
structure #1	$[_{\mathtt{T}\times\mathtt{V}}]$	...	$[_{\mathtt{T}\times\mathtt{V}}]$
⋮	⋮	⋮	⋮
structure #B	$[_{\mathtt{T}\times\mathtt{V}}]$	...	$[_{\mathtt{T}\times\mathtt{V}}]$

rows representing the total number of inspections performed on the structural element and V columns representing the element's information imported by the user, The data in the columns from left-to-right are, the aggregated observation $\tilde{y}$, inspector I_i, inspection year t, structure unique identifier, element's material identifier, structure's age and structural attributes. In the case when the user select to extract structural elements with intervention, two additional columns are added to the pervious list. These columns are the intervention year and the intervention code which refers to the type of action performed on the structural element [1]. It should be noted that the button *Export Filtered Data* only extracts the element's inspection data according to the selected structural element category.

3.1.3 Interventions

This tab is designed to help in merging interventions reports, extracting interventions per structural element category and extracting interventions costs. The merging functionality is necessary in the case when some intervention reports have non-overlapping information. For example, an intervention is reported in the budget database but is non-existent in the realized interventions database. In order to use the merging functionality, the user has to specify the number of files to be merged and the last year with an intervention. Clicking the *Merge Files* button will result in opening a loading window to load each file separately. For each file, the user has to select the columns associated with Structure ID and the intervention starting date. The outcome of the merged files is a single file that contain a complete list of all structures that underwent an intervention along with the interventions dates.
In the same tab, it is possible also to extract intervention data as a ".csv" file based on the structural element category. This can be done by loading the *realized interventions database*, thereafter, selecting the necessary columns as described in §2.
The last section in this tab is related to costs associated with interventions actions, which are reported in a database called *planned interventions* database. The required columns to be loaded by the user are the same as described in §2.

3.1.4 Network-Scale

The network-scale tab is designated for the network-scale analyses framework, such that the data extracted is for multiple structural elements picked by the user. This tab requires the user to load the *Elements Details* database which contain the material quantity information as described in §2.1.6. After assigning the input in the dropdown menus and selecting the structural elements involved in the network-scale analyses, the data can be extracted by clicking the button *Extract Data*. The output from this tab involves multiple ".mat" files, which includes,

- NetworkData: contains the structures' IDs, the annual average traffic of daily traffic

and columns representing each structural element category selected by the user.

- NetworkQuantity: the structural elements quantity associated with each element in each bridge.
- StructrualAttributesLabels: meta file for the labels of selected of structural attributes.
- Elements: meta file for the selected structural elements.
- StructuresID: meta file for the numerical auto generated structure ID.

3.2 Generate Synthetic Data

The synthetic database is generated to be quantitively and qualitatively representative of the real inspection database. The main advantage of performing analyses with synthetic data is that the true deterioration state is known, which allows verifying the deterioration model performance. Figure 6 shows the user interface for the toolbox. This toolbox has an

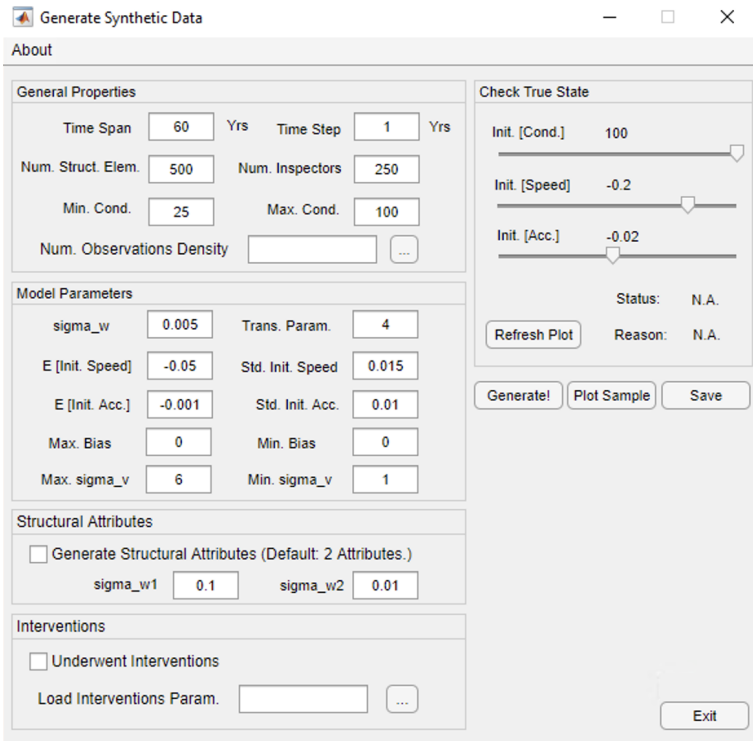


Figure 6: UI for the toolbox generate synthetic data.

interactive mode that allows checking the output based on the configurations provided by the user. This mode is accessible by interacting with the user interface components under the group *Check True State* or by clicking the button *Referesh Plot*. The following subsections describe the main steps for generating synthetic visual inspection data.

3.2.1 True Deterioration State

The true deterioration state for each structural element is represented by a continuous quantity, generated based on the transition model in Equation 4. The true deterioration state

is ensured to have resemblance for the qualitative characteristics of a realistic deterioration by passing through several criteria. A synthetic true state is excluded from the database if,

1. The deterioration is slow over time: $x_{\frac{T}{2}} > 0.85 \times x_1$.
2. Positive speed without an intervention: $\dot{x}_1 > 0$.
3. Positive or fast acceleration without an intervention: $10^{-4} < \ddot{x}_1 < -0.09$.
4. Plateau in the deterioration curve: $x_T > 0.5 \times x_1$.
5. Speed threshold: $\dot{x}_1 < 0.01 \times x_1 - 1.3$.
6. Acceleration threshold: $\ddot{x}_1 < 0.001 \times x_1 - 0.13$.

Generating structural attributes for the synthetic dataset is done according to two models, where in each model it is assumed that the true deterioration speed exhibits a relation with each attribute z_j^1 and z_j^2 as in,

$$z_j^1 = \log(|\dot{x}_0^j|) + w_0^1 : W_0^1 \sim \mathcal{N}(w_0; 0, \sigma^2_{w_0^1}), \tag{1}$$

$$z_j^2 = \exp(\dot{x}_0^j) + w_0^2 : W_0^2 \sim \mathcal{N}(w_0; 0, \sigma^2_{w_0^2}). \tag{2}$$

Furthermore, it is important to take into consideration the diversity in the deterioration condition of the structural elements in the database. However, the relations among the deterioration condition, speed and acceleration are unknown in the real case, thus, all structural elements in the synthetic case are assumed to start in a perfect condition with a value close to zero for the deterioration speed and acceleration. Thereafter, a fixed time window is defined to sample the starting point for each time series. Figure 7 demonstrates the concept of sampling an initial state for each time series. It should be noted that the true

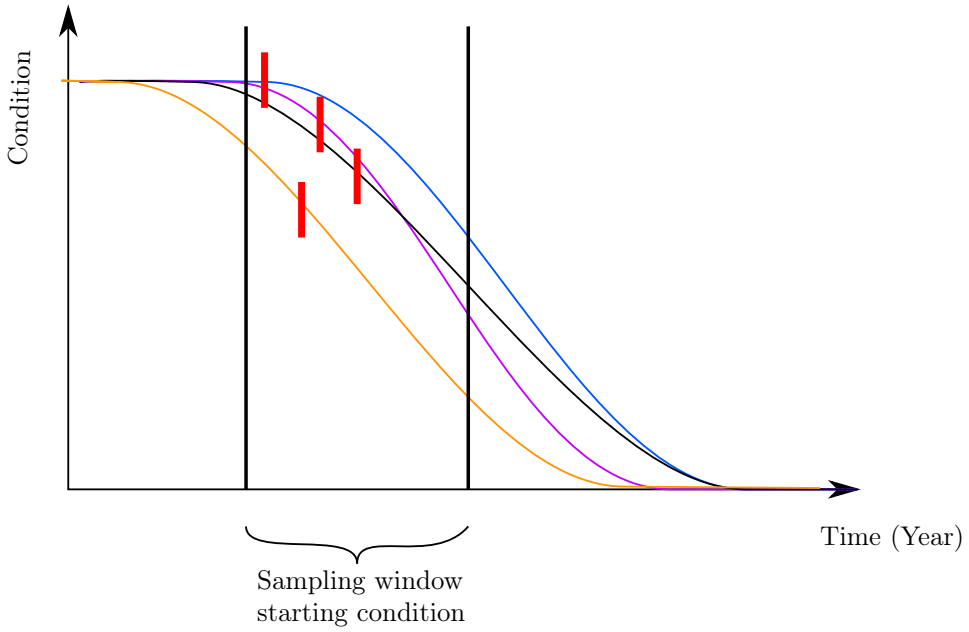


Figure 7: Sampling an initial state for each time series using a fixed time window.

state and the observations for the synthetic data are generated in the transformed space.

3.2.2 True Deterioration State with an Intervention

When generating true states with interventions, the above qualitative conditions are modified and augmented to include,

1. The slow deterioration after an intervention: $x_{\frac{\mathrm{T}}{1.5}} > 0.85 \times x_\tau$.
2. Plateau in the deterioration curve after an intervention: $x_\mathrm{T} > 0.75 \times x_\tau$.
3. There is no observation before or after the intervention.

In this context, simulating interventions is done according to two factors; the intervention priority, and the deterioration state. The priority factor is randomly assigned to structures, based on a uniform distribution $\Omega \sim \mathcal{U}(1, 3)$. This factor emulates the inspector's recommendation for performing an intervention in the real case. The type of intervention is determined based on a synthetic decision making system defined by if-then rules as shown in Table 2. These rules have two inputs and one output, the inputs are the health condition and the priority index of the bridge, while the output is the type of the intervention. In order to limit the number of rules, the deterioration condition and the priority index are discretized into categories as shown in Figure 8. In total, four synthetic intervention actions are defined, h_0: do nothing, h_1: preventive maintenance, h_2: repairs and h_3: major repairs. Whenever one of the actions $h_{1:3}$ is applied, the timing of the synthetic intervention τ is recorded. Furthermore, the health condition category "V.D." refers to a very damaged state of which a replacement action is required.

Table 2: Table of synthetic interventions h_r applied for a given health condition and a priority index.

		Health Condition		
		Damaged	Good	Excellent
Priority	High	h_3	h_2	h_1
	Medium	h_3	h_2	h_0
	Low	h_2	h_1	h_0

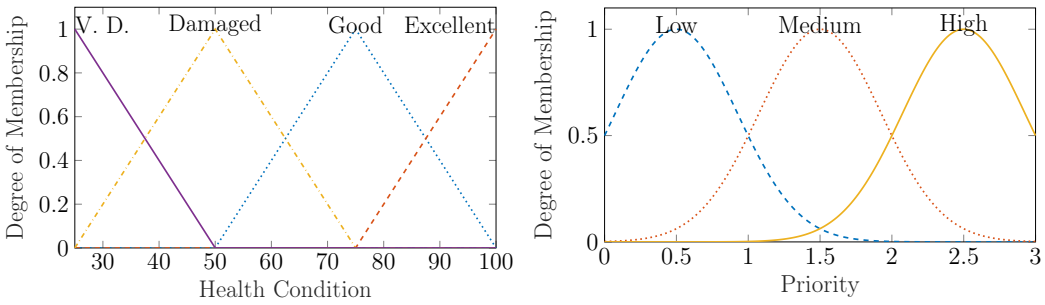


Figure 8: Categories for the health condition and the priority index.

3.2.3 Inspectors & Observations

The observations are generated from the true state using the observation model in Equation 5. Similarly to real cases, the observations are associated with a set of synthetic inspectors $\mathcal{I} = \{I_1, I_2, \ldots, I_\mathrm{I}\}$. Each synthetic inspector is assumed to have an observation error with

$v_t : V \sim \mathcal{N}(\mu_V(I_i), \sigma_V^2(I_i))$. The standard deviation $\sigma_V(I_i)$ and bias $\mu(I_i)$ are generated for each synthetic inspector using a uniform distribution and according to the user defined values in the UI. Moreover, the number of observations in each time series can be determined using the option *Num. Observations Density* in the UI. This option takes as an input a ".mat" file which contain a vector of values that sum up to 1. The interpretation of this input can be demonstrated by the example shown in Figure 9

0	0	0	0.5	0.3	0	0.1	0.1	0

Figure 9: Example for a vector input that contains the density for the number of observations per structural element in the database.

From Figure 9, each structural element will have 9 years of inspections, the inspection times are assigned randomly and the majority of structural elements have a total 4 observations. More specifically, a structural element with 4 inspections account to approximately 50% of the total number of structural elements, while there are no structural elements with a single inspection point. It should be noted that the synthetic inspector is assumed to revisit the same structural element 4-5 times throughout the lifespan of the structural element.

3.2.4 Output

The output of this toolbox is a set of ".mat" files, which can be obtained after clicking the button *Generate*. The list of files is composed of,

- Generated Data: contains all the information utilized to generate the synthetic inspection data.
- Full and Short: contains the structural element complete true state and the corresponding sampled observations.
- InspectorID: has the unique identifier (ID) for each synthetic inspector. This file is required as an input for the training toolbox.
- Observed: contains the synthetic inspections data that can be used to train the deterioration model.
- TrueInspector: includes the true value for the inspectors $\sigma_V(I_i)$, bias $\mu_V(I_i)$ and the synthetic inspector ID, which collectively can be utilized to verify the estimation procedure.

3.3 Verify Synthetic Data

The goal of this toolbox is to provide general verification analyses on synthetic data by comparing the model performance with the true parameters with the model performance with the estimated parameters. Figure 10 shows the main UI for the toolbox with the labeled check-boxes representing the output analyses of the toolbox.
The following subsections describe the inputs and outputs of the toolbox.

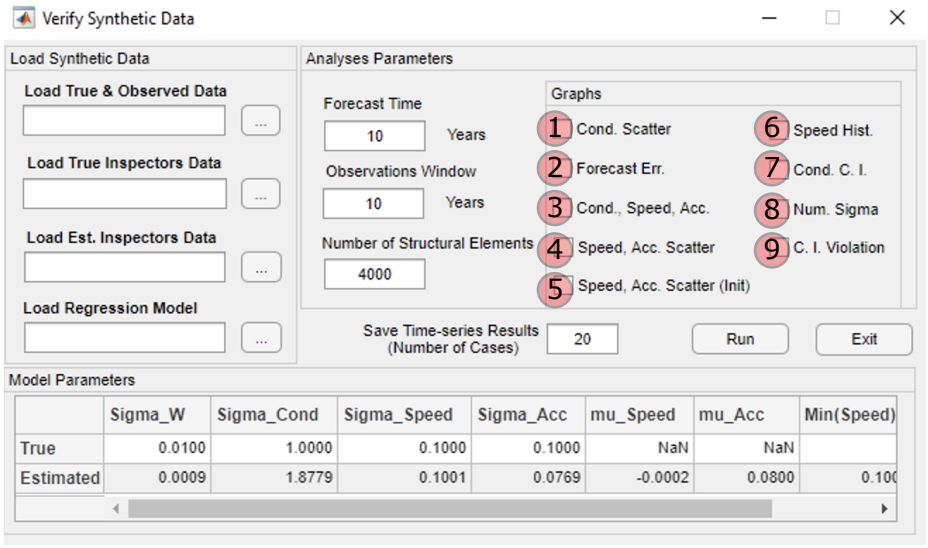


Figure 10: UI for the toolbox verify synthetic data, with the labeled components representing the output analyses/plots performed by the toolbox.

3.3.1 Input

The toolbox takes as input the true and estimated model parameters in addition to the synthetic observations and the true deterioration states. The section *Load Synthetic Data* require loading the following files with same order,

- Full and Short: contains the structural element complete true state and the corresponding sampled observations. This file is automatically generated from the *Generate Synthetic Data* toolbox.
- TrueInsp: includes the synthetic inspector ID and the true value for the parameters associated with the inspectors, which can be utilized to verify the estimation procedure. This file is automatically generated from the *Generate Synthetic Data* toolbox.
- Inspectors Data: contains the estimated inspectors parameters, which is generated from the *Model Training* toolbox.
- Regression Model: contains the estimated regression model parameters, which is also generated from the *Model Training* toolbox.

In addition to the list of files above, the user should provide the estimated and true model parameters in the table, size of the subset of structural elements involved in the analyses from the synthetic database, the number of forecast years and the size of time-window where observations are available.

3.3.2 Output

The output of this toolbox is mainly controlled by a list of checkboxes, with each checkbox corresponding to a different verification assessment. The following is a list for the output graphs associated with each checkbox,

1. Scatter plot for the model estimate of the condition compared to the true condition at pre-defined time stamps. Figure 11 shows an illustration for the verification results obtained for a number of structural elements $\mathtt{E} = 3250$. Each point in this scatter plot corresponds to a single structural element, and the alignment with the diagonal implies a good performance for the model.

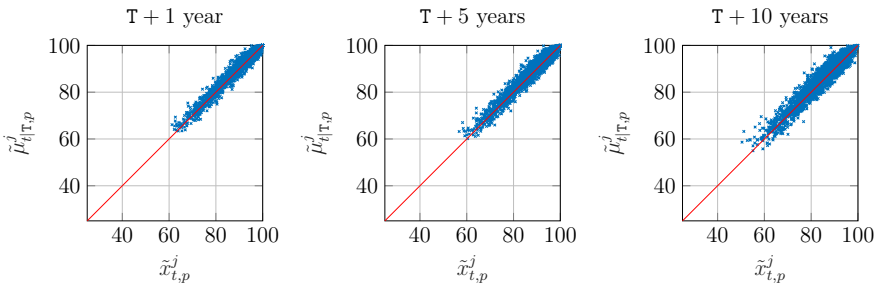


Figure 11: Scatter plot for the model estimate of the condition $\tilde{\mu}^j_{t|T,k}$ vs. the true condition $\tilde{x}^j_{t,k}$ at forecast years 1, 5 and 10.

2. Forecast error checkbox generates two graphs in order to assess the average model performance during forecast time. The first graph shows the absolute average error for the model forecast of the deterioration condition, speed and acceleration as shown in Figure 12.

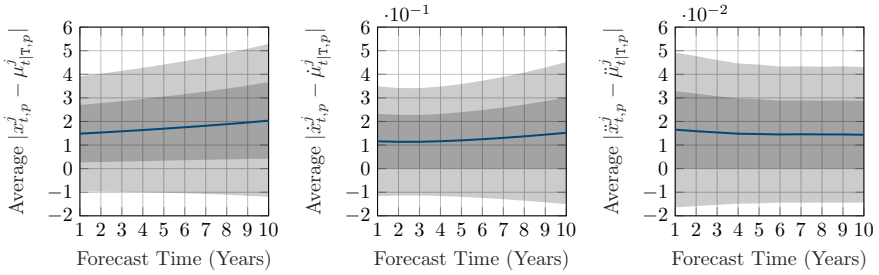


Figure 12: Absolute average error in forecast time for the expected condition, speed and acceleration based on the true condition, speed and acceleration respectively, with the 95% confidence interval ($\pm 2\sigma$) for each error.

The second graph shows the average error for the model forecast of the deterioration condition, speed and acceleration as shown in Figure 13.

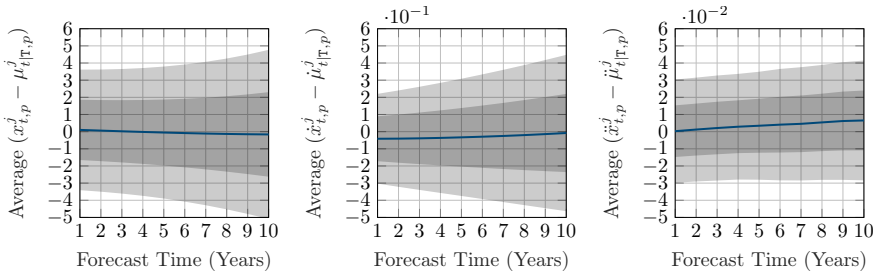


Figure 13: Average error in forecast time for the expected condition, speed and acceleration based on the true condition, speed and acceleration respectively, with the 95% confidence interval ($\pm 2\sigma$) for each error.

3. Shows the graphs for the element-level estimation along with the true state for the deterioration condition, speed and acceleration. Figure 14 shows an example results for the analyses performed in this option.

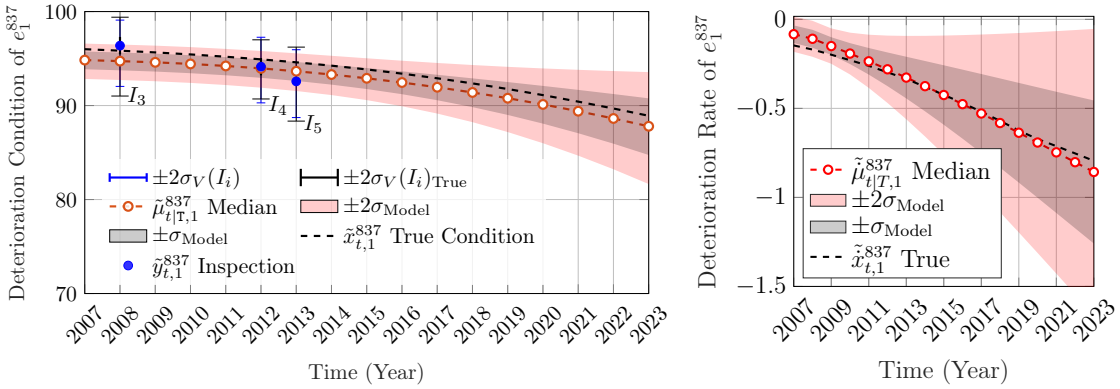


Figure 14: Deterioration state analysis for the condition and the speed based on the observations $\tilde{\boldsymbol{y}}^{837}_{t,1} \in [25, 100]$ of the synthetic structural element e_1^{837} with the error bars representing the inspectors' uncertainty estimates.

4. Scatter plots for the state estimate compared to the true state at the timestamp associated with the last observation in each time series. The first scatter plot is for the deterioration speed and the second scatter plot is for the deterioration acceleration.

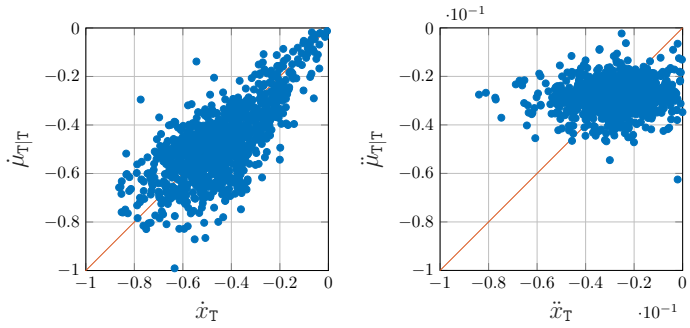


Figure 15: Scatter plots for the expected value v.s. the true value at time $t = \mathtt{T}$, with the first graph (left) representing the deterioration speed and the second graph (right) representing the acceleration.

5. Scatter plots for the state estimate compared to the true state at the first timestamp in each time series. The first scatter plot is for the deterioration speed and the second scatter plot is for the deterioration acceleration.

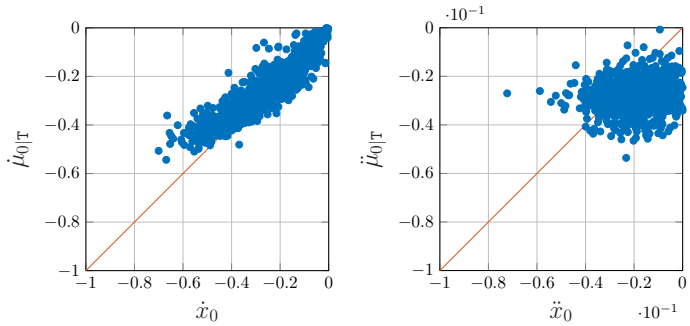


Figure 16: Scatter plots for the expected value v.s. the true value at time $t = 0$, with the first graph (left) representing the deterioration speed and the second graph (right) representing the acceleration.

6. This option compares the estimated deterioration speed with the true deterioration speed at the timestamp after the last observation from each time series. The results are shown on a histogram for the average error as illustrated in Figure 17.

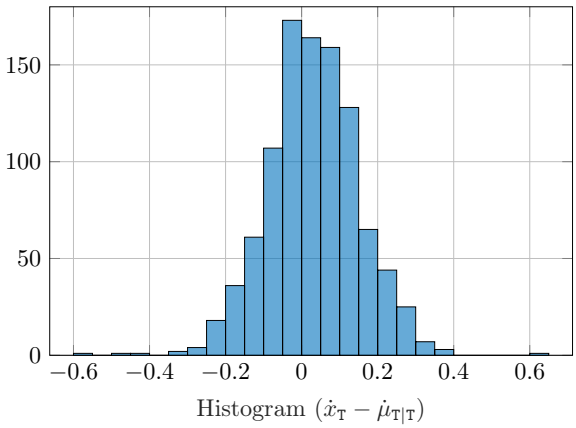


Figure 17: Histogram for the average error in estimating the deterioration speed after the last observation `T`.

7. Performs an assessment on the confidence interval (CI) of the model estimates, by examining the probability of the true deterioration condition being within the 95% confidence interval (i.e. $\mu \pm 2\sigma$). An example of results are shown in Figure 18, for the average probability of the true deterioration being within CI while using the estimated model parameters (left) and while using the true model parameters (right).

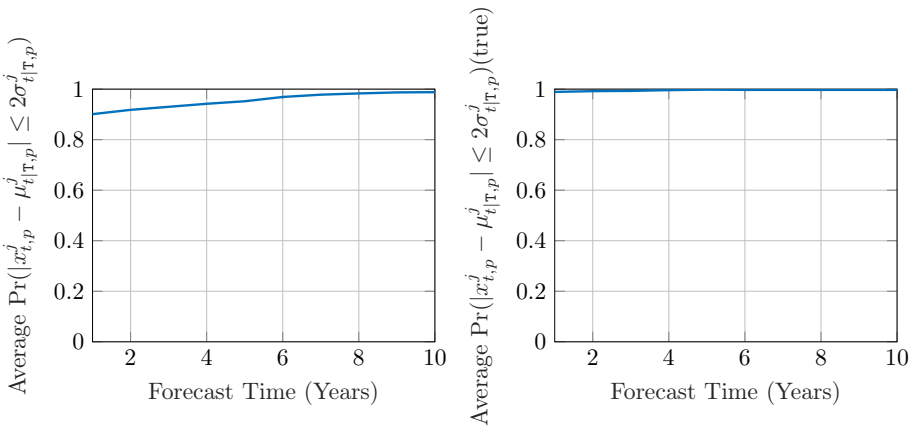


Figure 18: The probability of the true condition being within the 95% confidence interval of the model predicted state for the model with the estimated parameters (left) and true parameters (right).

8. Computes the average absolute error normalized by the standard deviation. This assessment shows how many standard deviations the model estimate is far from the true value of the condition. Figure 19 shows an example for the output of this analysis.

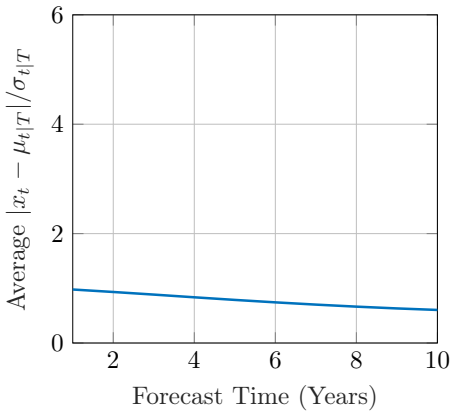


Figure 19: The normalized average absolute error for the model performance during forecast time.

9. Performs a check on the consistency of the confidence interval associated with the condition estimates. This is done by examining if the CI suggests positive improvement, on-average, for the condition in cases without interventions. If the CI is consistent, the results show a straight line parallel to the x-axis with $y = 0$.

3.4 Validation Using Real Data

This toolbox, shown in Figure 20, performs validation analyses on the real database. The inputs and outputs of this toolbox are discussed in the following subsections.

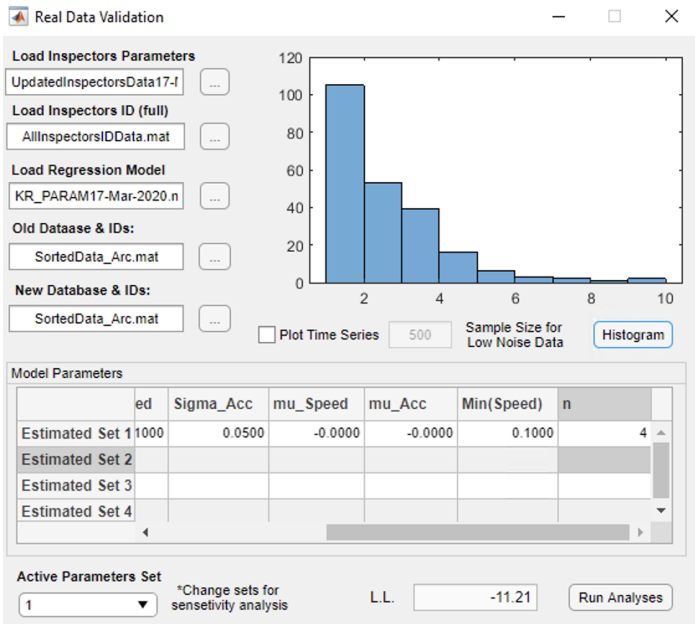


Figure 20: UI for real data validation toolbox.

3.4.1 Input

The main input to this toolbox are,

- the estimated inspectors parameters,
- the full list of inspectors IDs,
- regression model parameters (optional),
- an old visual inspections database file + the structures IDs file,
- the new visual inspections database file + the structures IDs file,
- a table of model parameters.

The old and new visual inspection databases are a reference to two databases with the same structural element category, however, the old database has an overall lower number of inspections.

3.4.2 Output

The output of this toolbox can be obtained after providing the input and clicking the button *Run Analyses*. The output includes the *log-likelihood* for the observations in the validation set along with multiple graphs that allows assessing the bias. The list of graphs is,

1. scatter plot for the model forecast versus new inspection data points, which refers to observations that were never used in training the deterioration model. Each point represent a model forecast $\tilde{\mu}_{t|\mathtt{T}-1}$ versus a new inspection $\tilde{y}_{t=\mathtt{T}}$ at time t for a population of structural elements.

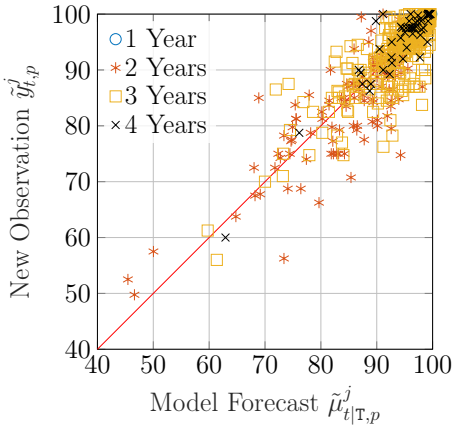


Figure 21: Model forecast $\tilde{\mu}_{t|\mathtt{T}}$ vs. new observations $\tilde{y}_{t=\mathtt{T}}$ with different symbols representing different forecast duration.

2. scatter plot for the model forecast versus new inspection data points with the symbol size representing the uncertainty associated with each new observation.

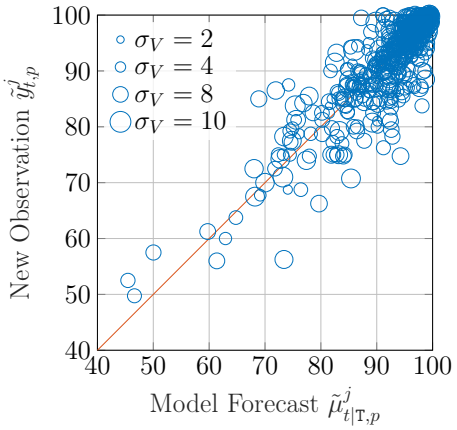


Figure 22: Model forecast $\tilde{\mu}_{t|\mathtt{T}}$ vs. new observations $\tilde{y}_{t=\mathtt{T}}$ with different symbols sizes representing the observation uncertainty.

3. histogram for the normalized residuals compared to a standard Normal distribution.

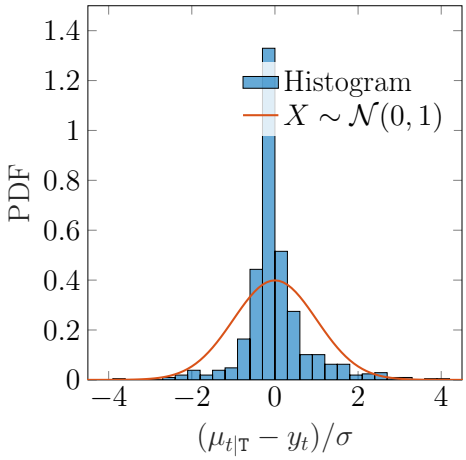


Figure 23: Normalized histogram for $(\mu_{t|\mathtt{T}} y_{t=\mathtt{T}})/\sigma_{t|\mathtt{T}}$.

4. model forecast compared to the new observation for a single structural element. This plot is obtained after activating the check-box *Plot Time Series*.

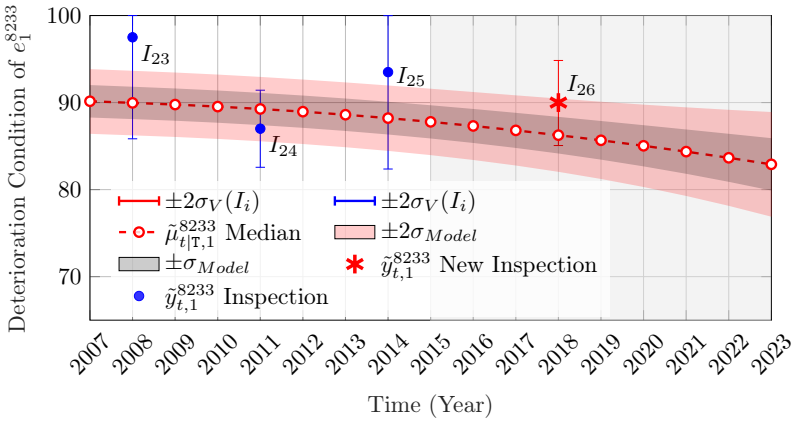


Figure 24: Condition deterioration analysis based on observations of a real structural element with error bars representing the inspectors estimated uncertainties.

3.5 Single Time Series

This toolbox represents a sandbox developed to test hypotheses and methods involved in the analyses of a single time series. Figure 25 shows the user interface, which include interactive components that actively react to the user configurations. The configurations can be stored by clicking the button *Save Case* which output a ".csv" file. The graphs can be also stored as ".pdf" file by clicking the button *Save Graph*.

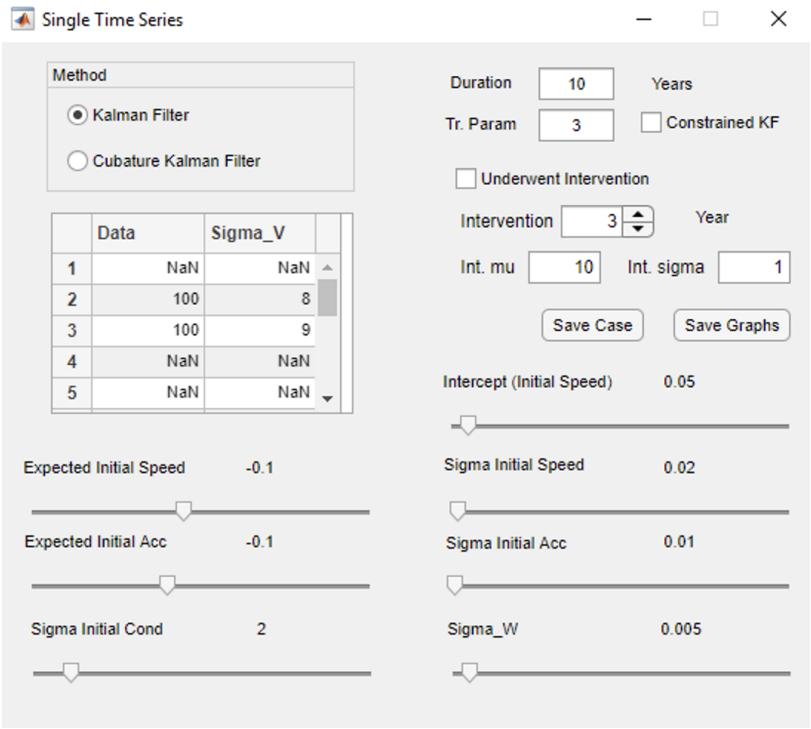
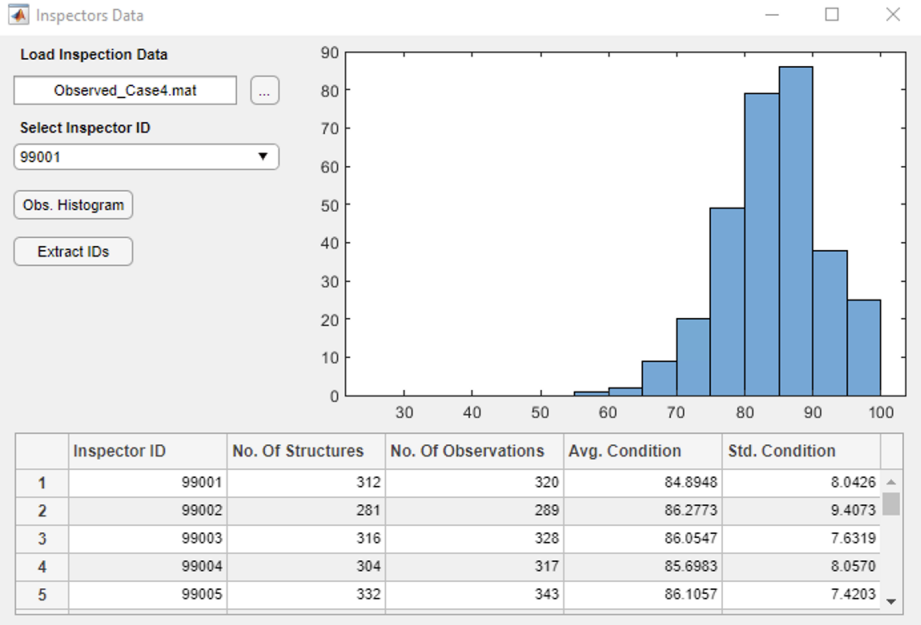


Figure 25: UI for single time series analyses toolox.

3.6 Inspectors Data

This toolbox provide information about each inspector involved in the database. The information include the number of inspections performed, the number of structures inspected and the average health condition. The input to this toolbox is the *Observed.mat* file, which contain the organized dataset for the inspection data 3.2.4. Figure 26 shows the user interface for the toolbox while displaying a histogram for all the observations collected by the selected inspector. The button *Extract IDs* outputs a ".mat" file with all the inspectors IDs involved



	Inspector ID	No. Of Structures	No. Of Observations	Avg. Condition	Std. Condition
1	99001	312	320	84.8948	8.0426
2	99002	281	289	86.2773	9.4073
3	99003	316	328	86.0547	7.6319
4	99004	304	317	85.6983	8.0570
5	99005	332	343	86.1057	7.4203

Figure 26: UI for the toolbox inspectors data.

in the inspections. It should be noted that the numbers shown for each inspector are reported without excluding any data points, therefore, in the case of removing noisy time series data, the numbers associated with each inspector are subject to change.

3.7 Model Training

The model training toolboxes are designated for estimating the deterioration and interventions model parameters, which includes configuring the parameters' bounds and selecting structural attributes. There are 3 model training toolboxes, the first toolbox is for estimating the deterioration model parameters for a single structural category, the second toolbox is for estimating the interventions model parameters and the third toolbox combines the aforementioned two toolboxes for network-scale parameter estimation. The following subsections describe the functionality of each toolbox.

3.7.1 Model Training - Generic

This toolbox is designated for estimating the deterioration model parameters for a single structural category. Figure 27 shows the main UI components for this toolbox. The input for this toolbox is the *Observed.mat* file, which contain the organized dataset for the inspection data and the *InspectorsID* which contain the inspectors ID for the inspectors involved in

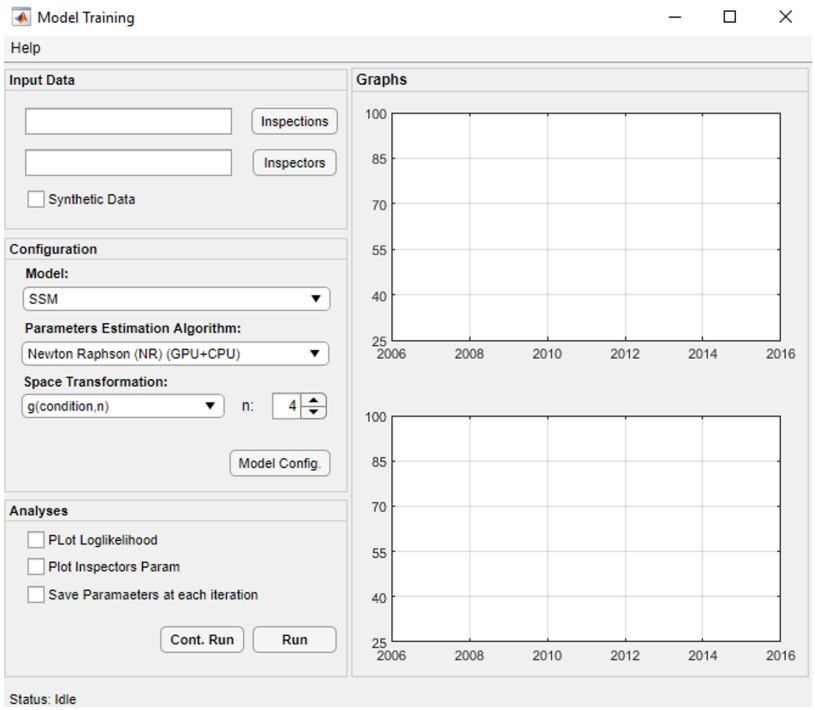


Figure 27: UI for the generic model training toolbox.

the inspections. The *Synthetic Data* checkbox is determine whether the dataset is synthetic or real. The main difference between a synthetic dataset and a real dataset is that the synthetic dataset does not require space transformation, because it is generated in the transformed space §3.2. The *Configuration* section in this toolbox includes specifying the type of deterioration model, the parameter estimation algorithm and the space transformation function. The button *Model Config* allows the user to specify the parameters upper and lower bounds for the deterioration model, in addition, it allows viewing and organizing structural attributes when regression analyses are involved. Figure 28 shows the interface for the model configuration window, after selecting the SSM-KR deterioration model. In this UI, the user can specify the deterioration model constraints and parameters bounds, in addition to the size of the independent test set. The role of a test set is to evaluate the model performance as the training process is carried out. It should be noted that for small inspections datasets, it is recommended to reduce the size of the test set. Reducing the independent test set to 0% would imply that the model will rely on the last observation from each time series as a test point. Furthermore, the *Start Year* refers to the year of which the first inspection point is performed. This date input helps the model identify time window for the time series analyses. The button *View Attributes* opens a new window for viewing/organizing/modifying the structural attributes data. The modifications includes inferring missing data or applying transformation on attributes, in addition to changing the order of the structural attributes in the list.

The last section in this toolbox is the *Analyses* section, which include checkboxes for plotting the Log-likelihood of the test set, the inspectors parameters $\sigma_V(I_i)$ and storing the model parameters, which are stored in a folder named "ParametersLog". The button *Run* starts the parameter estimation process using the initial values and configurations specified by the user, while the button *Cont. Run*, starts the parameter estimation process using pre-stored

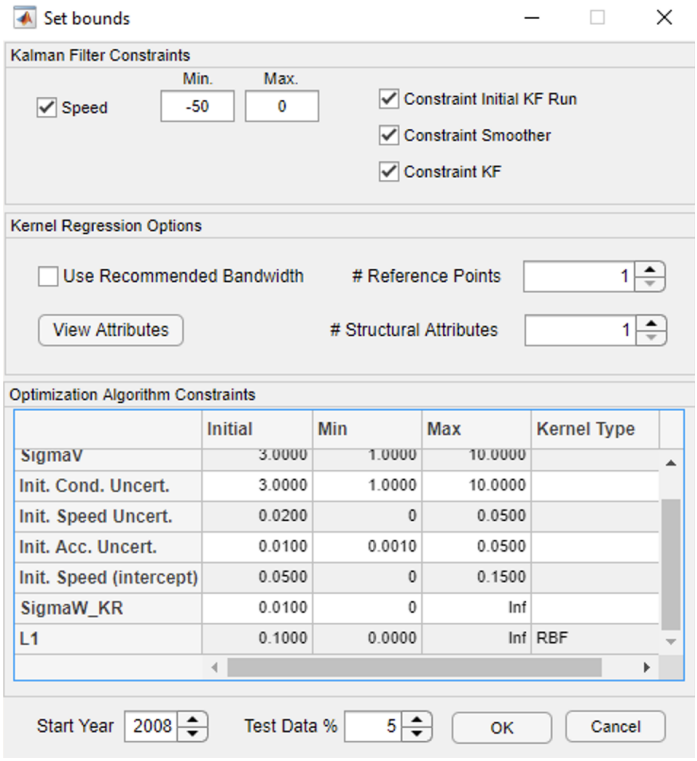


Figure 28: Model configuration window for the SSM-KR deterioration model.

parameters from the toolbox. The output of this toolbox is a set of ".mat" files,

- *Param*: which contains the all the SSM model parameters except the $\sigma_V(I_i)$.
- *KR_ Param*: contains the kernel regression model parameters.
- *UpdatedInspectorsData*: contains the inspectors IDs and the estimated parameters $\sigma_V(I_i)$ for each inspector (I_i).

3.7.2 Model Training - Intervention

The training toolbox for interventions has a simple interface (shown in Figure 29), which is utilized to learn the parameters of the interventions framework. This training framework assumes that each structural element has a single intervention. The interventions here are codified according to the Manual of Inspections from MTQ [1].
The input to this toolbox includes the dataset of inspections with interventions which can be obtained from the *Read Database* after opting to extract interventions only 3.1. This toolbox also takes as an input the deterioration model configurations and parameters. The *Model Config* tab in this toolbox contains the upper and lower bounds for the intervention framework parameters. The outputs of this toolbox are,

- *Network_ scale_ effect*: contains the expected values and the variance for the network-scale effect of the selected type of intervention.
- *Interventions_ params*: contains the model parameters associated with the selected type of intervention.

The above-mentioned ".mat" files can be obtained after clicking the button *Save Params*.

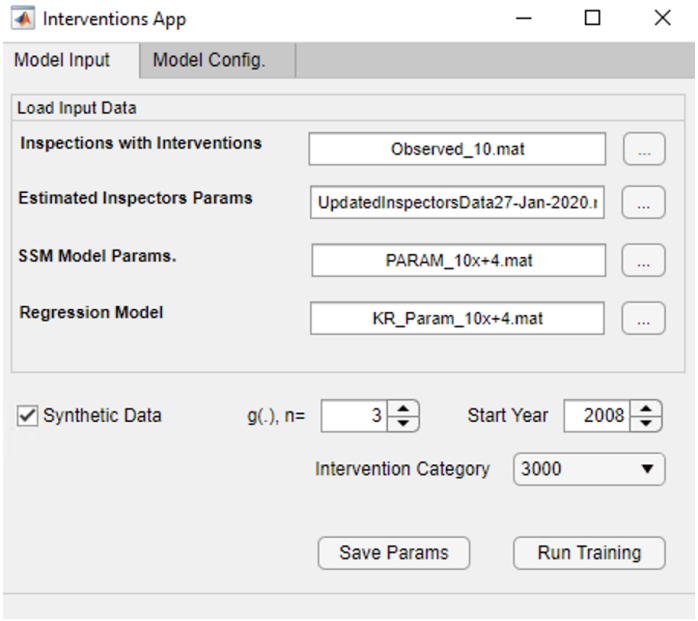


Figure 29: UI for the interventions model training toolbox.

3.7.3 Model Training - Network

This toolbox is designed to perform the parameter estimation for the deterioration and interventions frameworks on a network-scale. The input to this toolbox is similar to OpenIPDM main interface, which is the following set of ".csv" files,

- *Structures Database*: includes information about bridges such as the bridge's identifying number, location and other bridge level information.
- *Inspections Database*: includes the inspection information such as the inspection date and results, in addition to the structural element type and material.
- *Interventions Database* involves information about the interventions such as the intervention date and type as well as the structural element ID.
- *Budget Database*: includes information about records of investments made on each bridge over time.
- *Inspectors Database*: contains information about the inspector responsible for the intervention report.
- *Elements Details*: includes information associated with the characteristics of structural elements, such as the element quantity.

The columns to be imported from each database are pre-selected for the user by default. After loading all the files, the interface will extract the names of the structural elements categories and list them in the UI as shown in Figure 30.
The user can assign each structural category to either the SSM or the SSM-KR deterioration

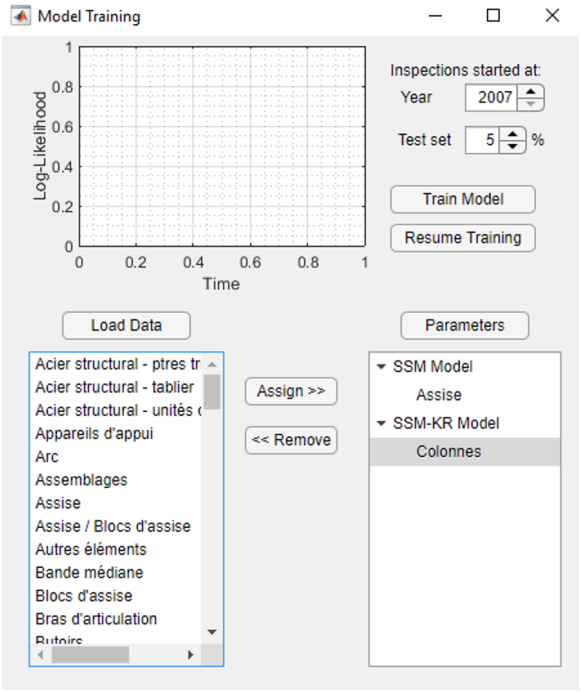


Figure 30: UI for the network-scale model training toolbox.

framework. For each structural category, the user must define the bounds and configurations for the parameter estimation by selecting the structural category and then clicking the button *Parameters*. After specifying the configurations for each structural category, the user can start the parameter estimation process by clicking the button *Train Model*. This process includes estimating the parameters for the deterioration framework as well as the interventions framework. The toolbox stores the parameters values into a single ".mat" file, which is automatically deposited into a folder named "Params". The button *Resume Training* is dedicated for resuming the parameter estimation after an interruption. Clicking the aforementioned button will result in a file loading window, which requires the user to load a ".mat" file. The ".mat" file in this case is a file that was automatically stored during the training process, which can be found in the "Params" folder.

4 Reference Theory

This section presents a summary of the theory behind the network-scale deterioration and interventions analyses of visual inspection data. For in-depth details and examples of analyses, the reader is advised to consult the following references:

Quantifying the Effects of Interventions Based on Visual Inspections from a Network of Bridges
Hamida, Z. and Goulet, J.-A.
Structure and Infrastructure Engineering. 2021 [DOI link]

Network-Scale Deterioration Modelling of Bridges Based on Visual Inspections and Structural Attributes
Hamida, Z. and Goulet, J.-A.
Structural Safety. 2021 [DOI link]

Modeling infrastructure degradation from visual inspections using network-scale state-space models
Hamida, Z. and Goulet, J.-A.
Structural Control and Health Monitoring. 2020 [DOI link]

Stochastic Modelling of Infrastructures Deterioration and Interventions based on Network-Scale Visual Inspections.
Hamida, Z.
Ph.D. Thesis 2020, Polytechnique Montreal [DOI link]

4.1 Deterioration Framework

The deterioration analyses on structural elements are mainly performed using a hybrid deterioration framework SSM-KR, that combines state-space models (SSM) with kernel regression (KR) [4]. The SSM-KR enables modelling the deterioration process using a kinematic model [7], which involves the element's deterioration condition x, speed $\dot{x}$ and acceleration $\ddot{x}$ defined over time by,

$$\underbrace{\begin{bmatrix} x_t \\ \dot{x}_t \\ \ddot{x}_t \end{bmatrix}}_{\boldsymbol{x}_t} = \underbrace{\begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{A}} \cdot \underbrace{\begin{bmatrix} x_{t-1} \\ \dot{x}_{t-1} \\ \ddot{x}_{t-1} \end{bmatrix}}_{\boldsymbol{x}_{t-1}} + \underbrace{\begin{bmatrix} w_t \\ \dot{w}_t \\ \ddot{w}_t \end{bmatrix}}_{\boldsymbol{w}_t}, \tag{3}$$

where $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t-1}$ are the state vector at time t and $t-1$, $\boldsymbol{A}$ describes the model kinematics for transitioning from $\boldsymbol{x}_{t-1}$ to $\boldsymbol{x}_t$ and $\boldsymbol{w}_t$ is the model-error vector. The above equations are applied within the context of *state-space models*, along with problem-specific modifications, which are detailed in the following subsections.

4.1.1 State-space models (SSM)

State-space models characterizes the deterioration behaviour by using two models, the *transition model* and the *observation model*, each defined as in,

$$\overbrace{\boldsymbol{x}_t = \boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{w}_t}^{\text{transition model}},\ \underbrace{\boldsymbol{w}_t : \boldsymbol{W} \sim \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, \boldsymbol{Q}_t)}_{\text{process errors}}, \tag{4}$$

$$\overbrace{y_t = \boldsymbol{C}\boldsymbol{x}_t + v_t}^{\text{observation model}},\ \underbrace{v_t : V \sim \mathcal{N}(v; 0, \sigma_V^2(I_i))}_{\text{observation errors}}. \tag{5}$$

The transition model here represents the model defined in Equation 3, with the deterioration state at time t described by $\boldsymbol{x}_t : \boldsymbol{X} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$, $\boldsymbol{w}_t$ is the process error with $\boldsymbol{Q}_t$ representing the error covariance matrix. From Equation 5, y_t represents the observation, $\boldsymbol{C}$ defines the observation matrix and $v_t : V \sim \mathcal{N}(v; 0, \sigma_V^2(I_i))$ is the observation error with $\sigma_V(I_i)$ being the standard deviation of the error associated with each inspector $I_i \in \mathcal{I}$ responsible for the inspection. The estimation of the hidden states is done using the *Kalman filter* (KF) [8] consisting in the *prediction step* and the *update step*. The prediction step is defined by,

$$\begin{aligned} E[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}] &\equiv \boldsymbol{\mu}_{t|t-1} = \boldsymbol{A}\boldsymbol{\mu}_{t-1|t-1} \\ \text{cov}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}] &\equiv \boldsymbol{\Sigma}_{t|t-1} = \boldsymbol{A}\boldsymbol{\Sigma}_{t-1|t-1}\boldsymbol{A}^\intercal + \boldsymbol{Q}_t, \end{aligned} \tag{6}$$

where $E[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}]$ refers to the expected value of the state vector $\boldsymbol{x}_t$ at time t given all the observations $\boldsymbol{y}_{1:t-1}$ up to time $t-1$, with $\boldsymbol{Q}_t$ defined as,

$$\boldsymbol{Q}_t = \sigma_W^2 \times \begin{bmatrix} \frac{dt^5}{20} & \frac{dt^4}{8} & \frac{dt^3}{6} \\ \frac{dt^4}{8} & \frac{dt^3}{3} & \frac{dt^2}{2} \\ \frac{dt^3}{6} & \frac{dt^2}{2} & dt \end{bmatrix}. \tag{7}$$

If an inspection point is available at time t, the expected value and covariance estimates are updated with the inspection information using the update step, defined by,

$$\begin{aligned} f(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) &= \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}) \\ \boldsymbol{\mu}_{t|t} &= \boldsymbol{\mu}_{t|t-1} + \boldsymbol{K}_t(\boldsymbol{y}_t - \boldsymbol{C}\boldsymbol{\mu}_{t|t-1}) \\ \boldsymbol{\Sigma}_{t|t} &= (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{C})\boldsymbol{\Sigma}_{t-1|t-1} \\ \boldsymbol{K}_t &= \boldsymbol{\Sigma}_{t-1|t-1}\boldsymbol{C}^\intercal G_t^{-1} \\ G_t &= \boldsymbol{C}\boldsymbol{\Sigma}_{t-1|t-1}\boldsymbol{C}^\intercal + \sigma_V^2(I_i)), \end{aligned}$$

where $\boldsymbol{\mu}_{t|t} \equiv E[\boldsymbol{X}_t|\boldsymbol{y}_{1:t}]$ the posterior expected value and $\boldsymbol{\Sigma}_{t|t} \equiv \text{cov}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t}]$ the posterior covariance at time t, conditional to the observations up to time t, $\boldsymbol{K}_t$ is the Kalman gain matrix, $\boldsymbol{I}$ is the identity matrix and G_t is the innovation covariance matrix. The Kalman smoother on the other hand is described by the *RTS Kalman Smoother* [9] equations,

$$\begin{aligned} f(\boldsymbol{x}_t|\boldsymbol{y}_{1:\mathrm{T}}) &= \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_{t|\mathrm{T}}, \boldsymbol{\Sigma}_{t|\mathrm{T}}) \\ \boldsymbol{\mu}_{t|\mathrm{T}} &= \boldsymbol{\mu}_{t|t} + \boldsymbol{J}_t(\boldsymbol{\mu}_{t+1|\mathrm{T}} - \boldsymbol{\mu}_{t+1|t}) \\ \boldsymbol{\Sigma}_{t|\mathrm{T}} &= \boldsymbol{\Sigma}_{t|t} + \boldsymbol{J}_t(\boldsymbol{\Sigma}_{t+1|\mathrm{T}} - \boldsymbol{\Sigma}_{t+1|t})\boldsymbol{J}_t^\intercal \\ \boldsymbol{J}_t &= \boldsymbol{\Sigma}_{t|t}\boldsymbol{A}^\intercal\boldsymbol{\Sigma}_{t+1|t}^{-1}. \end{aligned}$$

4.1.2 Constrained State Estimate

In order to ensure the monotonicity throughout the deterioration process, the state estimate of the deterioration speed has to be constrained by an upper bound as in $\dot{\mu}^j_{t|t,k} + 2\dot{\sigma}^j_{t|t,k} \leqslant 0$. Applying this constraint can be done by using the PDF truncation method proposed by Simon and Simon [10]. The PDF truncation method implies that for a state vector $\boldsymbol{x}_t$ with an expected value $\boldsymbol{\mu}_{t|t} \in \mathbb{R}^{\mathtt{n}\times 1}$ and a coefficient matrix $\boldsymbol{H} \in \mathbb{R}^{1\times \mathtt{n}}$, the hidden state vector can be constrained by a lower bound l and an upper bound u as in,

$$l \leqslant \boldsymbol{H}\boldsymbol{x}_t \leqslant u. \tag{8}$$

Applying the above constraint require decoupling the states in the state vector, which can be done by applying a space transformation. The transformation would result in transformed state $\bar{x}_t$ and transformed constraints $\bar{l}$ and $\bar{u}$,

$$\bar{l} \leqslant \bar{x}_t \leqslant \bar{u}, \tag{9}$$

Applying the constraints in Equation 9 can be done simply by approximating a truncated PDF with an expected value $\hat{\mu}_{t|t}$ and variance $\hat{\sigma}^2_{t|t}$,

$$\begin{aligned}
\hat{\mu}_{t|t} &= \bar{\alpha}\left[\exp\left(\frac{-\bar{l}^2}{2}\right) - \exp\left(\frac{-\bar{u}^2}{2}\right)\right],\\
\hat{\sigma}^2_{t|t} &= \bar{\alpha}\left[\exp\left(\frac{-\bar{l}^2}{2}\right)(\bar{l} - 2\hat{\mu}_{t|t}) - \exp\left(\frac{-\bar{u}^2}{2}\right)(\bar{u} - 2\hat{\mu}_{t|t})\right] + \hat{\mu}^2_{t|t} + 1,
\end{aligned}$$

where $\bar{\alpha}$, $\bar{l}$ and $\bar{u}$ are computed using,

$$\bar{l} = \frac{l - (\boldsymbol{H}\boldsymbol{\mu}_{t|t})}{\sqrt{(\boldsymbol{H}\boldsymbol{\Sigma}_{t|t}\boldsymbol{H}^\intercal)}},\quad \bar{u} = \frac{u - (\boldsymbol{H}\boldsymbol{\mu}_{t|t})}{\sqrt{(\boldsymbol{H}\boldsymbol{\Sigma}_{t|t}\boldsymbol{H}^\intercal)}},\quad \bar{\alpha} = \frac{1}{\sqrt{\pi/2}\left[\mathrm{erf}\left(\frac{\bar{u}}{\sqrt{2}}\right) - \mathrm{erf}\left(\frac{\bar{l}}{\sqrt{2}}\right)\right]},$$

with erf(.) representing the error function. The constrained expected value of the original state $\boldsymbol{\mu}_{t|t}$ and covariance $\boldsymbol{\Sigma}_{t|t}$ are,

$$\begin{aligned}
\boldsymbol{\mu}_{t|t} &= \boldsymbol{T}\boldsymbol{W}^{1/2}\boldsymbol{S}^\intercal\,[\hat{\mu}\ 0\ \ldots\ 0] + \boldsymbol{\mu}_{t|t},\\
\boldsymbol{\Sigma}_{t|t} &= \boldsymbol{T}\boldsymbol{W}^{1/2}\boldsymbol{S}^\intercal\mathbf{diag}([\hat{\sigma}^2\ 1\ \ldots\ 1])\boldsymbol{S}\boldsymbol{W}^{1/2}\boldsymbol{T}^\intercal,
\end{aligned}$$

where the matrices $\boldsymbol{T}$ and $\boldsymbol{W}$ are obtained from the *Jordan* canonical decomposition of $\boldsymbol{\Sigma}_{t|t}$ and the matrix $\boldsymbol{S}$ is obtained through the *Gram-Schmidt* orthogonalization [11].

4.1.3 Bounded Condition & Space Transformation

One of the main assumptions in the KF framework is that each state is defined by Normal distribution, which implies that the domain of each state is infinitely continuous. In the context of visual inspection data, this property does not hold due to the evaluation being bounded by maximum grade for a perfect condition and minimum grade for a poor condition. In order to resolve for this issue, space transformation is performed. The transformation function $g(.)$ and its inverse $g^{-1}(.)$ are derived in this framework from the CDF of the *Gamma distribution* defined by,

$$F(x;\alpha,\beta) = \frac{1}{\beta^\alpha\Gamma(\alpha)}\int_0^x t^{\alpha-1}e^{\frac{-t}{\beta}}\,dt. \tag{10}$$

where $\Gamma(.)$ is the gamma function and $\{\alpha, \beta\}$ are the Gamma distribution parameters. By assigning the parameter $\beta = 1$, the CDF function becomes the *incomplete gamma* function [12],

$$F(x, a) = \frac{1}{\Gamma(\alpha)} \int_0^x t^{\alpha-1} e^{-t} dt. \tag{11}$$

This function is defined for $x \in (0, \infty]$, which can be modified to be defined for $x \in [-\infty, \infty]$ as in,

$$\tilde{x} = g^{-1}(x) = \begin{cases} \frac{1}{\Gamma(\alpha)} \int_0^{x^{\frac{1}{\alpha}}} t^{\alpha-1} e^{-t} dt, & x > \frac{u+l}{2}, \\ x, & x = \frac{u+l}{2}, \\ -\frac{1}{\Gamma(\alpha)} \int_0^{x^{\frac{1}{\alpha}}} t^{\alpha-1} e^{-t} dt, & x < \frac{u+l}{2}, \end{cases} \tag{12}$$

where $\tilde{x}$ represents the state in the constrained space $\tilde{x} \in [l, u]$. The transformation function $g(.)$ mapping the state $\tilde{x} \in [l, u]$ to $x \in [-\infty, \infty]$ is defined by,

$$x = g(\tilde{x}) = \begin{cases} \left[\frac{1}{\Gamma(\alpha)} \int_0^{\tilde{x}} t^{\alpha-1} e^{-t} dt\right]^{\alpha}, & \frac{u+l}{2} < \tilde{x} \leqslant u, \\ \tilde{x}, & \tilde{x} = \frac{u+l}{2}, \\ -\left[\frac{1}{\Gamma(\alpha)} \int_0^{\tilde{x}} t^{\alpha-1} e^{-t} dt\right]^{\alpha}, & l \leqslant \tilde{x} < \frac{u+l}{2}, \end{cases} \tag{13}$$

where the parameter α is given by: $\alpha = 2^{-n}$, with n is a positive integer that controls the curvature at the transformation function ends.

4.1.4 Kernel Regression (KR)

Kernel regression is an approach that commonly used for pattern detection problems [13]. This approach relies on a kernel function in order to determine the levels of similarity between pairs of covariates. In the context of this project, the purpose of KR is to exploit the structural similarities across bridges in order to improve estimation of the initial deterioration speed $\dot{x}_{0,p}^j$. The KR framework is defined based on the *Nadaraya-Watson* model [14],

$$\dot{x}_{0,p}^j = (\boldsymbol{a}_p^j)^{\intercal} \dot{\boldsymbol{x}}_{\boldsymbol{z}} + w_0 : W_0 \sim \mathcal{N}(w_0; 0, \sigma_{w_0}^2), \tag{14}$$

with the vector $\boldsymbol{a}_p^j$ obtained by,

$$\boldsymbol{a}_p^j = \frac{\boldsymbol{k}\left(\boldsymbol{z}_j, \boldsymbol{Z}_{c(m)}, \boldsymbol{\ell}\right)}{\sum_{m=1}^{\mathtt{M}} \boldsymbol{k}\left(\boldsymbol{z}_j, \boldsymbol{Z}_{c(m)}, \boldsymbol{\ell}\right)}, \; m = 1, \ldots, \mathtt{M}, \tag{15}$$

where $\boldsymbol{z}_j$ is a vector of $\mathtt{Q}$ covariates associated with the j-th bridge and $\boldsymbol{Z}_c$ is a matrix that encodes a $\mathtt{Q}$-dimensional grid of reference points. The $\mathtt{Q}$-dimensional grid is obtained from discretizing the range of the covariates with an equal number of $\mathtt{M}$ reference points, such that $\boldsymbol{Z}_c = \left[\boldsymbol{z}_c^1 \; \ldots \; \boldsymbol{z}_c^{\mathtt{Q}}\right] \in \mathbb{R}^{\mathtt{M}^{\mathtt{Q}} \times \mathtt{Q}}$. The function $\boldsymbol{k}(.)$ is a multivariate kernel function $\boldsymbol{k} : \mathbb{R}^{\mathtt{Q}} \to \mathbb{R}$ representing the multiplicative kernel,

$$\boldsymbol{k}\left(\boldsymbol{z}_j, \boldsymbol{Z}_{c(m)}, \boldsymbol{\ell}\right) = k\left(\frac{z_j^1 - \boldsymbol{z}_{c(m)}^1}{\ell_1}\right) \cdot \ldots \cdot k\left(\frac{z_j^{\mathtt{Q}} - \boldsymbol{z}_{c(m)}^{\mathtt{Q}}}{\ell_{\mathtt{Q}}}\right), \; m = 1, \ldots, \mathtt{M}. \tag{16}$$

where $k(.)$ is the univariate kernel function and $\boldsymbol{\ell} = [\ell_1 \ldots \ell_{\mathtt{Q}}]$ represents the kernel length parameter associated with each covariate. The univariate kernel functions employed in this platform are,

- Aitchison and Aitken kernel function:

$$k^{(\mathrm{AAK})}(z_j, \boldsymbol{z}_c) = \begin{cases} 1-\ell, & z_j = z_c, \\ \frac{\ell}{\mathtt{C}-1}, & z_j \neq z_c, \end{cases}, \ell \in [0, \tfrac{\mathtt{C}-1}{\mathtt{C}}].$$

- Radial basis kernel function:

$$k^{(\mathrm{RBF})}(z_j, \boldsymbol{z}_c) = \exp\left(-\tfrac{(z_j - \boldsymbol{z}_c)^2}{2\ell^2}\right).$$

- Matérn 12 kernel function:

$$k^{(\mathrm{M12})}(z_j, \boldsymbol{z}_c) = \exp\left(-\tfrac{(z_j - \boldsymbol{z}_c)}{\ell}\right).$$

- Matérn 52 kernel function:

$$k^{(\mathrm{M52})}(z_j, \boldsymbol{z}_c) = \left(1 + \tfrac{\sqrt{5}(z_j - \boldsymbol{z}_c)}{\ell} + \tfrac{5}{3}\tfrac{(z_j - \boldsymbol{z}_c)^2}{\ell^2}\right) \exp\left(-\tfrac{\sqrt{5}(z_j - \boldsymbol{z}_c)}{\ell}\right).$$

4.2 Modelling Deterioration and Interventions

The full framework that allows modelling the deterioration and interventions require augmenting the state vector. This is done to include the components representing the change in the condition δ_t, the speed $\dot{\delta}_t$, and acceleration $\ddot{\delta}_t$, as in,

$$\boldsymbol{x}_{p,t}^j = \left[x_{p,t}^j \ \dot{x}_{p,t}^j \ \ddot{x}_{p,t}^j \ \delta_t \ \dot{\delta}_t \ \ddot{\delta}_t\right]^\intercal. \tag{17}$$

The effect of an intervention h_r on a structural element is incorporated in the deterioration model by applying the following modifications,

$$\boldsymbol{x}_t = \boldsymbol{A}_t \boldsymbol{x}_{t-1} + \boldsymbol{w}_t, \ \boldsymbol{w}_t : \begin{cases} \boldsymbol{W}^{\mathtt{ki}} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_t^{\mathtt{ki}}) \\ \boldsymbol{W}^{r} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_t^{r}) \end{cases} \tag{18}$$

where the transition matrix $\boldsymbol{A}_t$ is defined by,

$$\boldsymbol{A}_{t=\tau} = \begin{bmatrix} \boldsymbol{A}^{\mathtt{ki}} & \boldsymbol{I}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix}, \ \boldsymbol{A}_{t\neq\tau} = \begin{bmatrix} \boldsymbol{A}^{\mathtt{ki}} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix}, \tag{19}$$

with $\boldsymbol{I}$ representing the identity matrix and $\boldsymbol{A}^{\mathtt{ki}}$ defined by,

$$\boldsymbol{A}^{\mathtt{ki}} = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}. \tag{20}$$

The covariance for the transition model errors is described by,

$$\boldsymbol{Q}_{t=\tau} = \begin{bmatrix} \boldsymbol{Q}^{\mathtt{ki}} + \boldsymbol{Q}^{r} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{Q}^{r} \end{bmatrix}, \ \boldsymbol{Q}_{t\neq\tau} = \begin{bmatrix} \boldsymbol{Q}^{\mathtt{ki}} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{0}_{3\times3} \end{bmatrix}, \tag{21}$$

with $\boldsymbol{Q}^r$ and $\boldsymbol{Q}^{\mathtt{ki}}$ defined as,

$$\boldsymbol{Q}^r = \mathrm{diag}\left(\left[\sigma_{w_r}^2 \ \dot{\sigma}_{w_r}^2 \ \ddot{\sigma}_{w_r}^2\right]\right), \ \boldsymbol{Q}^{\mathtt{ki}} = \sigma_w^2 \begin{bmatrix} \frac{\Delta t^5}{20} & \frac{\Delta t^4}{8} & \frac{\Delta t^3}{6} \\ \frac{\Delta t^4}{8} & \frac{\Delta t^3}{3} & \frac{\Delta t^2}{2} \\ \frac{\Delta t^3}{6} & \frac{\Delta t^2}{2} & \Delta t \end{bmatrix}. \tag{22}$$

The matrix $\boldsymbol{Q}^r$ is a diagonal matrix containing the standard deviations describing the element-level intervention errors.

4.3 Aggregating States Estimates

Aggregating the deterioration state estimates is required whenever we want to upscale the deterioration analyses to the bridge level or the network level. The aggregation is done using the Gaussian mixture [15], which is described as in,

$$p(\hat{\boldsymbol{x}}_t) = \sum_{j=1}^{\mathtt{J}} \lambda^j \mathcal{N}(\boldsymbol{x}_t^j; \boldsymbol{\mu}_{t|t}^j, \boldsymbol{\Sigma}_{t|t}^j), \tag{23}$$

where λ^j represents the mixture weight with, $\sum_{j=1}^{\mathtt{J}} \lambda^j = 1$, and $\mathtt{J}$ being the total number of components. In the context of this project, the overall deterioration state of the system can be approximated by the expected value $\hat{\boldsymbol{\mu}}_{t|t}$ and covariance $\hat{\boldsymbol{\Sigma}}_{t|t}$ defined by,

$$\begin{aligned} \hat{\boldsymbol{\mu}}_{t|\mathtt{T}} &= \sum_{j=1}^{\mathtt{J}} \lambda^j \boldsymbol{\mu}_{t|\mathtt{T}}^j, \\ \hat{\boldsymbol{\Sigma}}_{t|\mathtt{T}} &= \sum_{j=1}^{\mathtt{J}} \lambda^j \boldsymbol{\Sigma}_{t|\mathtt{T}}^j + \sum_{j=1}^{\mathtt{J}} \lambda^j (\boldsymbol{\mu}_{t|\mathtt{T}}^j - \hat{\boldsymbol{\mu}}_{t|\mathtt{T}})(\boldsymbol{\mu}_{t|\mathtt{T}}^j - \hat{\boldsymbol{\mu}}_{t|\mathtt{T}})^\intercal, \end{aligned} \tag{24}$$

where the covariance $\hat{\boldsymbol{\Sigma}}_{t|t}$ consists in the summation of the variance resulting from within-elements and the variance resulting from between-elements [16].

4.4 Maximum Likelihood Estimate (MLE)

The MLE approach is intended for maximizing the joint prior probability of observations while assuming the observations to be conditionally independent given the state x_t. Therefore, the likelihood for a set of observations can be described by,

$$f(y_{1:\mathtt{T}}|\boldsymbol{\theta}) = \prod_{t=1}^{\mathtt{T}} f(y_t|y_{1:t-1}, \boldsymbol{\theta}). \tag{25}$$

In order to avoid numerical instabilities due to the multiplication, the natural logarithm is taken for the likelihood estimate. Thus, the *log-likelihood* estimate is given by,

$$\ln f(y_{1:\mathtt{T}}|\boldsymbol{\theta}) = \sum_{t=1}^{\mathtt{T}} \ln f(y_t|y_{1:t-1}, \boldsymbol{\theta}). \tag{26}$$

In the context of this project, the analyses are performed on a network-scale, hence, the *log-likelihood* estimate is taken for the inspections of all the structural elements $e_p^j \ \forall j, p$ combined. Therefore, the network-scale *log-likelihood* is defined by,

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{j=1}^{\mathtt{B}} \sum_{p=1}^{\mathtt{E}_j} \sum_{t=1}^{\mathtt{T}_p} \ln f(y_{t,p}^j|y_{1:t-1,p}^j, \boldsymbol{\theta}), \tag{27}$$

where $\mathtt{B}$ is the total number of bridges, $\mathtt{E}_j$ is the total number of structural elements in the j-th bridge and $\mathtt{T}_p$ is the total number of observations for the p-th structural element.

5 FAQ Troubleshooting

This section contains some of the common issues that the user might encounter when using OpenIPDM platform.

- **An error occurs when the user tries to load a ".csv" file in the toolbox.**

  This is a common problem in databases with a column that contain comments and text data. This problem occurs due to having extra "commas" that exist in the text field data, which create issues in the ".csv" formatted files. Resolving this issue is possible by excluding the column with the text fields before exporting the database from ".accdb" to ".csv" file.

- **The software is crashing during saving data/graphs, or during the execution of a script.**

  This problem can occur due to one of the following reasons,

  - The software main folder is not the current directory of MATLAB. In order to check for the current directory, use the command *pwd*.
  - Conflicting script name, which can occur when using a MATLAB function name, that already exist in the software. Resolving this issue is possible by trying to change the script name or by cleaning the MATLAB path which can be done by using the command *pathtool*.

- **The model parameter estimation is too slow.**

  The speed of computations can be affected by different factors, below is a short-list for some of them:

  - The size of the database and the number of model parameters (or inspectors) can significantly slow down the computation due to needing high computational resources.
  - Alternating the calculations between CPU and GPU can cause slowing down the computational time, especially when the calculations are expected to take a long time. One way to resolve for such an issue is to continuously store the estimated parameters and restart the software with the stored parameters whenever a noticeable slowdown has occurred.
  - Starting from parameters values that are far from being realistic, provided the problem context can extend the computational time for the parameters estimation. In case of confusion about the correct starting point, it is advisable to try different plausible values to assess the time gain and identify the best starting point.

- **The model parameter estimation is crashing.**

  The common causes of this issue could be associated with,

- The parameters upper and lower bounds are not well defined. To resolve this issue, try adjusting the parameters' upper and lower bounds or even the initial values.
- An issue with the GPU card or an issue with the CPU parallel processing toolbox in MATLAB. This problem can be identified from the error message displayed on MATLAB command line. The solutions to such an issue can vary depending on the cause, therefore, it is recommended to consult an IT in this case.

- **Problem in compiling the ".tex" figures.**

  This issue can rise when attempting to compile large ".tex" figures. Resolving this issue can be done by using the Lulatex compiler, which is a compiler that is available in most TEX editing/compiling software.

- **How to cite OpenIPDM ?**

  Stochastic Modelling of Infrastructures Deterioration and Interventions based on Network-Scale Visual Inspections.
  Hamida, Z.
  Ph.D. Thesis 2020, Polytechnique Montreal [DOI link]

Acknowledgements

This project is funded by the Transportation Ministry of Quebec Province (MTQ), Canada. The authors would like to acknowledge the support of René Gagnon for facilitating the access to the inspections database employed in this research project.

References

[1] MTQ. *Manuel d'Inspection des Structures*. Ministère des Transports, de la Mobilité Durable et de l'Électrification des Transports, Jan 2014.

[2] Zachary Hamida and James-A. Goulet. A stochastic model for estimating the network-scale deterioration and effect of interventions on bridges. *Structural Control and Health Monitoring*, 2021 (submitted).

[3] Zachary Hamida and James-A. Goulet. Quantifying the effects of interventions based on visual inspections of bridges network. *Structure and infrastructure engineering*, 2021.

[4] Zachary Hamida and James-A Goulet. Network-scale deterioration modelling based on visual inspections and structural attributes. *Structural Safety*, 2020.

[5] Zachary Hamida and James-A Goulet. Modeling infrastructure degradation from visual inspections using network-scale state-space models. *Structural Control and Health Monitoring*, pages 1545–2255, 2020.

[6] Zachary Hamida. *Stochastic Modelling of Infrastructures Deterioration and Interventions based on Network-Scale Visual Inspections*. PhD thesis, Polytechnique Montreal, 2020.

[7] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley Sons, 2004.

[8] Rudolf Emil Kalman. Contributions to the theory of optimal control. *Bol. Soc. Mat. Mexicana*, 5(2):102–119, 1960.

[9] Herbert E Rauch, CT Striebel, and F Tung. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450

[10] Dan Simon and Donald L Simon. Constrained kalman filtering via density function truncation for turbofan engine health estimation. *International Journal of Systems Science*, 41(2):159–171 0020–7721, 2010.

[11] Dan Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley Sons, 2006.

[12] Armido R DiDonato and Alfred H Morris Jr. Computation of the incomplete gamma function ratios and their inverse. *ACM Transactions on Mathematical Software (TOMS)*, 12(4):377–393

[13] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[14] Elizbar A Nadaraya. On estimating regression. *Theory of Probability Its Applications*, 9(1):141–142 0040–585X, 1964.

[15] Douglas A Reynolds. Gaussian mixture models. *Encyclopedia of biometrics*, 741, 2009.

[16] Andrew R Runnalls. Kullback-leibler approach to gaussian mixture reduction. *IEEE Transactions on Aerospace and Electronic Systems*, 43(3):989–999