User Guide for "Estimating post-disaster traffic conditions using real-time data streams" MATLAB source code

Reece Otsuka, Daniel Work, and Junho Song

August 25, 2014

Abstract

This document describes the implementation of the ensemble Kalman filter algorithm for estimating post-disaster traffic conditions, introduced in the article "Estimating post-disaster traffic conditions using real-time data streams" by Otsuka, Work, and Song, submitted for publication in Structure and Infrastructure Engineering: Maintenance, Management, Life-Cycle Design and Performance. A preprint of the article is available for download on the second author's website. The source code is hosted at `https://github.com/rotsuka/postDisaster_EnKF`.

1 License

This software is licensed under the *University of Illinois/NCSA Open Source License*:

2 General Instruction

This document explains how to explicitly reproduce the results of the article "Estimating post-disaster traffic conditions using real-time data streams." It also provides general instruction on how to use the code to create and analyze a user-designed bridge transportation network. This software is intended to run on MATLAB R2012a or newer. It may run on older versions of MATLAB, but it must have the *object-oriented programming* (OOP) capabilities. Please contact Reece Otsuka at reece.otsuka@gmail.com if you have any questions. Thank you.

3 Replicating results

1. To generate the graphical results of Figure 8, open **Results_Midwest_HI.mat** in the folder *Results_SingleRun* and plot using *plotResults_indiv.m*. Figures 8(a)-(e) in the paper are respectively Figures 1, 6, 8, 2, and 4 of the MATLAB output.

   The **.mat** file mentioned above may be found at

   `https://www.dropbox.com/sh/z44z1ovyhb20gtl/AACRcqK_ILC5rJVq9qN1qx91a?dl=0`.

2. To generate the BEEQ results in Table 7, open the folder *Results_100SimTests* and run one of the following depending on the damage state desired:

   - *medium*: **Results_100SimTests_MD.mat**
   - *high*: **Results_100SimTests_HI.mat**
   - *total*: **Results_100SimTests_TO.mat**

   The BEEQ values are stored in the variable *trueBEEQ*. The variable *ratioMat* is a matrix whose columns contain the computed ratio values for each simulation run.

3. To generate BEEQ results in Table 8, open the folder *Results_100SimTests_DiffEns* run one of the following depending on the number of ensembles desired:

   - 100: **Results_100SimTests_HI_100Ens.mat**
   - 150: **Results_100SimTests_HI_150Ens.mat**
   - 200: **Results_100SimTests_HI_200Ens.mat**
   - 300: **Results_100SimTests_HI_300Ens.mat**
   - 500: **Results_100SimTests_HI_500Ens.mat**
   - 1000: **Results_100SimTests_HI_1000Ens.mat**

   The computation cost is obtained by using *sum(simtVec)* and is given in seconds. The BEEQ values in Table 8 are taken from the third column of the variable *trueBEEQ* for each **.mat** file. The geometric standard deviation, σ_g, is obtained by using the *geostd.m* function in the folder *PostProcessing*.

4 Creating a new simulation scenario

This section briefly explains how to make a new simulation setup. In order to run a simulation, the input parameters in one of the following files in the folder *Mains* must be edited:

- **EQTrafficModel_Main_EQandNoEQ_singleRun.m**
- **EQTrafficModel_Main_EQandNoEQ_multiRun.m**
- **EQTrafficModel_Main_EQandNoEQ_multiRun_parallelCompfn.m**

To do a single run, edit the first file. To do multiple runs with the same input parameters, edit the second or third files. You must use the third file if you intend to use parallel computing. Please refer to the comments in the code itself for further explanation on how to use and edit these files. Some instruction will be provided in the following sections.

4.1 Creating the road network geometry

1. Determine the number of nodes, links, traffic sensors, and bridges the network has and give integer values to *numNodes*, *numLinks*, *numSensors*, and *numBridges*, respectively.

2. *inpNodes.m*, *inpLinks.m*, and *inpSensors.m*: these functions create cell arrays of the values and IDs of the associated objects. Make sure the IDs have no spaces (e.g. 'clarkSt' not 'Clark St'). You must also create start (s) and end (e) links, which will serve as the boundary conditions. All internal links should be given type i.

3. *inpBridges.m* Bridges are given site conditions, material type, and the desired damage state. The bridge fragility parameters are adapted from Nielson and DesRoches (2007). Currently, the code is hardcoded to handle only three fragility curves (limit states).

4. *inpEQ.m*: creates the earthquake by specifying a magnitude.

NOTE: at this point, it might be beneficial to create a save state (**.mat** file) so the road inputs do not need to be done again if another simulation is to be run on a network with the same road geometry. Currently, there are three road network geometry files pertinent to the results of the paper in the folder *PreloadedGeoms*:

1. **Geom__Midwest__SensNoise10__9Sensors__MD.mat**

2. **Geom__Midwest__SensNoise10__9Sensors__HI.mat**

3. **Geom__Midwest__SensNoise10__9Sensors__TO.mat**

4.2 Other input parameters

1. Simulation variables: choose a time of simulation, length of time step, and time of earthquake occurrence (in seconds).

2. EnKF variable, *tolerance*: tells how much the initial discrepancy between the true model and estimation model matters (i.e. initial error covariance matrix).

3. Fragility model: the user is free to choose the degree of accuracy of which to map computed PGA values to. A value of $delta = 0.001$ should be adequate but that is up to the user.

4. Initial conditions, *rho0Acc*, *rho0App*: these are the initial density values of the *true solution* and *estimation solution*. They must be ordered in the same order as the links were created.

5. Boundary condition noise: give *startSig* and *endSig* values for the standard deviation on the upstream and downstream boundary conditions in terms of percentages of the true values.

6. Sending and receiving function and maximum flow region errors: give the means and standard deviations of the desired normal distributions of the noise parameters.

7. Probability of sensor failure, *pf__sens*: by default, set to 0, but can be changed to reflect that traffic sensors may be damaged from the earthquake. This changes the shape of $\mathbf{H}$.

8. Parameters of true and approximate models, *isImperfParams*: set to 0 to give the true and approximate models the same road parameters ($q_{\max}$, $v_{\max}$, ρ_{j}) or 1 to draw realizations of these parameters for each ensemble.

5 Running the model

At this point, the model can be executed. To do a single run, simply run **EQTrafficModel__Main__-EQandNoEQ__singleRun.m**. To do multiple runs, run **multiRun__Main.m** or **multirun__Parallel__-Main.m** for parallel computing. Descriptions of the functions of the model can be obtained using the built-in *help* function of MATLAB.

6 Post-processing capabilities

After the main code is run, there are many post-processing options available. These cannot be run without first executing the main code. They may be found in the folder named *PostProcessing*.

1. *plotResults_indiv.m*: plots the true, open, prior, and posterior filter solutions in separate windows for both cases. The first four graphs are the solutions with the earthquake input and the second four are the solutions without the earthquake input. The true solution is the same in both cases. This is not a function; it is a script that calls the function *densPlotter.m*.

2. *plotResults_subplots.m*: plots the true, open, prior, and posterior filter solutions in two subplots. The first set of graphs are the solutions with the earthquake input and the second set are the solutions without the earthquake input. The true solution is the same in both cases. This is not a function; it is a script that calls the function *densPlotter.m*.

3. *computeBEEQ.m*: computes the BEEQ value for a given estimator.

4. *plotCompCovar.m*: plots various solutions with standard deviation bands for the desired time steps.

5. *plotCovarEvo.m*: plots the contours of the covariance matrices for the open loop, prior, and posterior solutions for the desired time steps.

6. *plotPriorPostEns.m*: plots the prior and posterior ensembles with the true measurements and ensemble measurements for the desired time steps.

7. *plotQmax.m*: plots the distribution of q_{max} at the desired time steps for the open loop and filter solutions. It also shows the bridge flow capacities of the *true solution*.

8. *plotError.m*: plots the difference errors ($true - prior$, $true - posterior$, and $posterior - prior$) as a contour plot.

9. *plotFundDiaEns.m*: plots the fundamental diagram for the true and approximate models for each ensemble in a desired ensemble range (time-invariant).

10. *extremumCheck.m*: gives the minimum and maximum of the densities considering both the time and space axes.

7 Diagnostics

These functions are intended to help the user run diagnostic tests and may be useful in debugging future modifications to the code. They are not necessary for viewing simulation results. They may be found in the folder named *Diagnostics*.

1. *findBadEns.m*: finds ensembles that are out of range and returns an array that tells the time steps, ensemble numbers, and cell locations of the bad ensembles.

2. *fundDiaPlot.m*: plots fundamental diagram and sending and receiving diagrams with noise. This is not a function.

3. *uniFragTest.m*: create a deterministic fragility model with multiple fragility curves and simulate probabilistic draws from the distribution. This is not a function.

4. *X5_test.m*: checks each column of an **X5** matrix to determine if it follows the numerical properties given in Evensen (2009).

5. *plotFDEnsNoise.m*: plots the true fundamental diagram and sample points drawn from distributions of the true parameters of the fundamental diagram. This is not a function.

8 Running a simulation with or without an earthquake

It is also possible to run a simulation where the model either has the earthquake input or does not. To do a single run in this fashion, execute **EQTrafficModel_Main_EQorNoEQ.m**. In this script, there is a variable *isEQinp* which will either be given the value 1 (earthquake input) or 0 (no earthquake input). To analyze the graphical results, run *plotResults.m*. All other post-processing techniques above can also be used in this instance. Running a simulation in this way may be useful if quick results are desired. However, the initial ensemble draws in the earthquake and no earthquake scenarios are not preserved, so the results are not necessarily ideal for comparison.

References

G. Evensen. *Data Assimilation: The Ensemble Kalman Filter*. Springer, 2009.

B. G. Nielson and R. DesRoches. Analytical seismic fragility curves for typical bridges in the Central and Southeastern United States. *Earthquake Spectra*, 23(3):615–633, 2007.