1 General Instruction

1.1 General usage notes

1. This release of software is intended to complement a paper submission to the 53rd Annual Conference on Decision and Control (CDC);

2. This software is intended to be run on Eclipse. It was developed using Eclipse 4.3.0, and the development environment is Java SE Development Kit 7u25.

1.2 How to use the software

1. Download the **DLKCF_Observable** and **DLKCF_Consensus**, and import the two projects into Eclipse. The folder *MatlabPlot* includes matlab code to generate plots for the true densities and the estimates;

2. To generate Fig. 1b in the paper, run *Test.java* in **DLKCF_Observable** and plot the generated *trueState.csv* in the folder *results* using *Main.m* in the folder *MatlabPlot*. Note that *trueState.csv* should be put in the folder *Result_Plot*, same for the rest of the plots generated using *Main.m*;

3. To generate Fig. 2a in the paper, run *Test.java* in **DLKCF_Observable** and plot the generated *writerAvr.csv* using *Main.m* in the folder *MatlabPlot*;

4. To generate Fig. 2b in the paper, run *Test.java* in **DLKCF_Observable** and generate the solid line by the data in the first column of *Lyapfun.csv*. For the dashed line, set the variance of model error on the boundary cells from 0.0009 to 0.09 (line 174, 180, 184, 185 in *RoadModel.java*), and plot the first column of the generated *Lyapfun.csv*;

5. To generate Fig. 3a in the paper, run *Test.java* in **DLKCF_Consensus** and plot the generated *trueState.csv* using *Main.m* in the folder *MatlabPlot*;

6. To generate Fig. 3b in the paper, run *Test.java* in **DLKCF_Consensus** and plot the generated *writerAvr.csv* using *Main.m* in the folder *MatlabPlot*;

7. To generate Fig. 3c in the paper, run *Test.java* in **DLKCF_Consensus** and plot the first columns of *Disagreement.csv* and *Disagreement1.csv*, to generate the solid and dash dot line, respectively;

8. The csv files in the folders *results* are generated by the supplementary code, which can be considered the same as the figures in the paper subject to some random Gaussian noise from the initial guess noise and the sensor noise.

2 Package List

2.1 DLKCF_Observable

1. *DoubleMatrix*, some operations on matrix and *GaussianGenerator.java* for generating Gaussian (and other types of) noise;

2. *trueSolution*, true solution in each section computed by the original switching mode model with true boundary conditions;
3. *section*, used to specify the mode, and to compute the matrices A, B^{ρ}, and B^{q} in the SMM;
4. *model*, used to specify parameters used for estimation (e.g. model/measurement error covariance matrix, output matrix, initial guess of the estimation, and system dynamics used in estimation);
5. *filters*, the distributed Kalman filter and the consensus filter for each agent, the entire estimation process is in **Estimation**, where a static method *exportResult* includes all the steps in estimation and functions to export result.

2.2 DLKCF_Consensus

The packages and their functions in **DLKCF_Consensus** resembles those in **DLKCF_Observable**, and here we only list packages not included in Section 2.1 and packages that have different functions.

1. *trueSolutionCTM*, where true solution of the entire road network is computed by the Godunov scheme in equation (3) of the paper;
2. *trueSolution*, true solution in each section obtained by selecting the part of the true solution computed by *trueSolutionCTM* that is within the local section.

3 Parameters Changing Instruction

1. *Initial guess of the estimates* can be changed in the method *defineVarAndMean()* in *RoadModel.java*;
2. *Model error covariance matrix* is initially defined in the method *defineVarAndMean()* in *RoadModel.java*, and can be updated at each time step by *updateModelVar()* in *RoadModel.java*;
3. *Model error covariance matrix* is initially defined in the method *defineVarAndMean()* in *RoadModel.java*, and can be updated at each time step by *updateModelVar()* in *RoadModel.java*;
4. *Measurement error covariance matrix* (applicable only in **DLKCF_Consensus** now) is initially defined in the method *initialThis()* in *D.java*. To update the measurement error covariance matrix in each step, create another method by coping *updateModelVar()* in *RoadModel.java* to update the measurement error covariance and apply it in the method *Estimations.setSection*;
5. *Parameters in the system dynamics* used by local agents in estimation can be ρ_m, ρ_c, and v_m can be changed in the method $TrueSolution.setSections$ (applicable only in **DLKCF_Consensus** now), where each section observe true boundary densities and set its mode based on its own parameters;
6. *Boundary conditions* can be changed in the method *Estimation.exportResult* (starting from line 421). There are three boundary types to choose now (specified in the comment lines), and boundary type number can be changed in *Test.java* (line 76) as a parameter of *Estimation.exportResult* (now it is set as 2).