CML: Model and SE

Darren Craig Roos

Nov 24, 2019

CONTENTS

1 Model 3

2 State Estimator 5

3 Labview 7

4 Fake Inputs 9

5 Labview Inputs 11

6 Fake State Update 13

7 Labview State Update 15

8 Plotting 17

Index 19

This code can be used to test and simulate a model in real-time. The model is of a fumaric acid producing bioreactor. The code interacts with Labview and also implements UKF state estimation for the glucose, ethanol and fumaric acid HPLC inputs.

The source code is available here.

CHAPTER

ONE

MODEL

class Model.**Model**(*X0*, *inputs*, *t=0*, *pH_calculations=False*)

A nonlinear model of the system

Parameters

- **X0** (*array_like*) – Initial states
- **inputs** (*callable*) – Must take in a parameter t (the current time) and return an array_like of the current inputs
- **t** (*float, optional*) – Initial time. Defaults to zero
- **pH_calculations** (*bool, optional*) – If *True* then pH calculations are made. Defaults to *False*

X

Array of current state

Type array_like

inputs

Must take in a parameter t (the current time) and return an array_like of the current inputs

Type callable

t

Initial time

Type float

pH_calculations

If *True* then pH calculations are made

Type bool

rate_matrix_inv

The inverse of the rate matrix. Placed here so that it is only calculated once

Type 2d array_like

DEs(*t*)

Contains the differential and algebraic equations for the system model. The rate equations defined in the matrix *rate_matrix* are described by:

1) glucose + 2*CO2 + 6*ATP –> 2*FA + 2*water
2) glucose –> 6*CO2 + 12*NADH + 4*ATP (TCA)

3) NADH + 0.5*O2 -> 7/3 ATP (Respiration)

4) glucose -> 2*ethanol + 2*CO2 + 2*ATP

5) glucose + gamma*ATP –> 6*biomass + beta*NADH

where the unknowns are: rFAp, rTCA, rResp, rEp, rXp

Parameters **t** (*float*) – The current time

Returns **dX** – The differential changes to the state variables

Return type array_like

step(*dt*)

Updates the model with inputs

Parameters **dt** (*float*) – Time since previous step

calculate_pH()

Calculates the pH in the vessel.

Returns **pH** – The pH of the tank

Return type float

outputs()

Returns all the outputs (state and calculated)

Returns **outputs** – List of all the outputs from the model

Return type array_like

get_Xs()

Gets all the states that are stored

get_data()

Gets all relevant information from the object

CHAPTER

TWO

STATE ESTIMATOR

class StateEstimator.**StateEstimator**(*X0*, *inputs*, *t_predict*)

Implements the UKF state estimator for the system

Parameters

X0 [array_like] The initial states of the system

inputs [callable] Must take in a parameter t (the current time) and return an array_like of the current inputs

t_predict [float] The period between state estimator predictions

inputs

Must take in a parameter t (the current time) and return an array_like of the current inputs

Type callable

ts

List of times that the state estimator has been run

Type array_like

Q

A matrix of state covariances

Type 2d array

R

A matrix of measurement covariances

Type 2d array

fx

A function that can be used for state transition

Type callable

nx

The number of states

Type int

sigmas

A sigma point generating object

Type MerweScaledSigmaPoints

ukf
A UKF implementation

Type filterpy.kalman.UnscentedKalmanFilter

t
The current time

Type float

t_next_predict
The next time at which prediction should take place

Type float

t_predict
The period between state estimator predictions

Type float

t_next_predicts
An array of all past prediction times

Type array_like

static hx(*x*)

Parameters **x** (*array_like*) – A list of the states

Returns **z** – A list of the observations in measurement space

Return type array_like

step(*dt*)
Steps the object through time

Parameters **dt** (*float*) – The amount of time since the previous call

update(*z*, *t=nan*)
Performs an update step

Parameters

- **z** (*array_like*) – A list of the observations
- **t** (*float*) – The time at which the observations took place

get_Xs()
Get the _Xs array

get_deviations()
Get the _deviations array

get_data()
Get all the data from the object

CHAPTER

THREE

LABVIEW

class labview.**Labview**

Contains the required objects to store state for labview functions

t

The current time

Type float

ts

List of times

Type array_like

inputs

Input object that stores and retrieves inputs

Type *inputters.LabviewInputs*

su

State update object that stores and retrieves HPLC input data

Type *stateUpdaters.LabviewStateUpdate*

X0

The initial state

Type array_like

m

The system model

Type *Model.Model*

Xs

List of previous states

Type array_like

t_predict

The period between state predictions

Type float

se

State estimator object

Type *StateEstimator.StateEstimator*

live_plot
If *True* then a live plot of the run is shown

Type bool

labview.**init**()
Initialises the labview interface. Called before the while loop in labview

labview.**finalise**()
Ends the labview interface. Called after the while loop in labview

labview.**update_inputs**(*t*, *inputs*)
Passes inputs into the system to the model.

Parameters

- **t** (*float*) – Current time
- **inputs** (*array_like*) – The values of the inputs

labview.**step**(*t*)
Steps the labview object through time

Parameters **t** (*float*) – Current time

labview.**update_state**(*t*, *z*)
Sends update to the state updater

Parameters

- **t** (*float*) – Timestamp of the update
- **z** (*array_like*) – Update values

labview.**get_glucose_graph**(*confidence=0.95*)
Passes outputs to labview from the model

Parameters **confidence** (*float*) – The confidence probability for the plots

CHAPTER

FOUR

FAKE INPUTS

class inputters.**FakeInputs**(*glucose_data_file*)

Creates fake inputs for the glucose feed from past data

Parameters **glucose_data_file** (*string*) – The name of the csv file that contains the glucose data

glucose

An object containing the info from the glucose file

Type pandas.Dataframe

CgFg(*t*)

Interpolates the value from the glucose file

Parameters **t** (*float*) – The value of time at which the input should be looked up

CHAPTER

FIVE

LABVIEW INPUTS

class inputters.**LabviewInputs**

Stores and looks up input values from Labview

ts

Stores the time stamp information about the inputs

Type array_like

inputs

Stores the inputs

Type array_like

Cg_in

The glucose feed concentration

Type float, constant

G_rpm_to_ml_min

The conversion factor between rpm and ml/min for the glucose pump

Type float, constant

Cco_in

The percentage CO2 in the CO2 feed

Type float, constant

Co_in

The percentage O2 in the O2 feed

Type float, constant

Cn_in

The concentration urea in the nitrogen feed

Type float, constant

N_rpm_to_ml_min

The conversion factor between rpm and ml/min for the nitrogen pump

Type float, constant

B_rpm_to_ml_min

The conversion factor between rpm and ml/min for the sodium hydroxide pump

Type float, constant

Cb_in
The concentration of sodium hydroxide in the base feed

Type float, constant

M_rpm_to_ml_min
The conversion factor between rpm and ml/min for the mineral pump

Type float, constant

T_amb
The ambient room temperature

Type float

Q_fact
A multiplier for the heater gain

Type float, constant

update(*t*, *data*)
Update the current inputss

Parameters

- **t** (*float*) – Current time
- **data** (*array_like*) – Current inputs

get_data()
Get all the input data

Returns **out** – All the input data

Return type array_like

CHAPTER

SIX

FAKE STATE UPDATE

class stateUpdaters.**FakeStateUpdate**(*update_data_file*, *t=0*, *backdate=0*)

Creates fake state updates from past data

Parameters

- **update_data_file** (*string*) – The name of the csv file that contains the data
- **t** (*float, optional*) – Initial time. Defaults to zero
- **backdate** (*float, optional*) – Sets a time to back date samples. Simulates the fact that HPLC readings are not instant. Defaults to zero

concentration

An object containing the info from the data file

Type pandas.Dataframe

t

Current time.

Type float

backdate

Sets a time to back date samples

Type float

ts_meas, Cg_meas, Cfa_meas, Ce_meas

List of HPLC readings and time stamps

Type array_like

Cis

Adjusted HPLC readings

Type array_like

step(*dt*)

Steps the updater through time :param dt: Time since the previous step :type dt: float

update_ready()

Returns *True* if there is an update ready for the state estimator

get_update()

Get the state update for the state estimator

Returns **z** – The update

Return type array_like

get_times()
Gets the times of the state updates

get_data()
Gets all data from the object

CHAPTER

SEVEN

LABVIEW STATE UPDATE

class stateUpdaters.**LabviewStateUpdate**

Handles the state updates from Labview

update

True if there is an update ready for the state estimator

Type bool

update_values

List of previous update values

Type array_like

update_value

List of current update value

Type array_like

update_time

Time at which the readings were taken

Type float

ts

List of previous times

Type array_like

step(*dt*)

Steps the updater through time

update_ready()

Returns *True* if there is an update ready for the state estimator

get_update()

Get the state update for the state estimator

Returns

- **update_time** (*float*) – Time at which the readings were taken
- **update_value** (*array_like*) – The update

get_times()

Gets the times of the state updates

get_data()
Gets all data from the object

CHAPTER

EIGHT

PLOTTING

plotting.**plot_all**(*file_name*, *confidence=0.95*, *show=True*)
Plots all the graphs from a file

Parameters

- **file_name** (*string*) – The name of the file in which all the data is stored
- **confidence** (*float, optional*) – The confidence probability for the plots Defaults to 95%
- **show** (*bool, optional*) – If *True* then the plt.show method is called at the end. Useful to turn off when you want to add additional things Defaults to *True*

plotting.**plot_live**(*ts*, *model_obj: Model.Model*, *se_obj: StateEstimator.StateEstimator*, *su_obj: {<class 'stateUpdaters.LabviewStateUpdate'>, <class 'stateUpdaters.FakeStateUpdate'>}*, *confidence=0.95*)

Parameters

- **ts** (*array_like*) – List of times
- **model_obj** (Model.Model) – Model object
- **se_obj** (StateEstimator.StateEstimator) – State estimation object
- **su_obj** (*{stateUpdaters.FakeStateUpdate, stateUpdaters.LabviewStateUpdate}*) – State updating object
- **confidence** (*float, optional*) – The confidence probability for the plots Defaults to 95%

plotting.**plot_data**(*file_name*, *show=True*)
Plots state update data from a file

Parameters

- **file_name** (*string*) – The name of the file in which all the data is stored
- **show** (*bool, optional*) – If *True* then the plt.show method is called at the end. Useful to turn off when you want to add additional things Defaults to *True*

plotting.**plot_model**(*file_name*, *show=True*)
Plots state update data and model data from a file

Parameters

- **file_name** (*string*) – The name of the file in which all the data is stored

- **show** (*bool, optional*) – If *True* then the plt.show method is called at the end. Useful to turn off when you want to add additional things Defaults to *True*

This documentation was last updated on Nov 24, 2019.

INDEX

B

B_rpm_to_ml_min (*inputters.LabviewInputs attribute*), 11
backdate (*stateUpdaters.FakeStateUpdate attribute*), 13

C

calculate_pH() (*Model.Model method*), 4
Cb_in (*inputters.LabviewInputs attribute*), 11
Cco_in (*inputters.LabviewInputs attribute*), 11
Cg_in (*inputters.LabviewInputs attribute*), 11
CgFg() (*inputters.FakeInputs method*), 9
Cis (*stateUpdaters.FakeStateUpdate attribute*), 13
Cn_in (*inputters.LabviewInputs attribute*), 11
Co_in (*inputters.LabviewInputs attribute*), 11
concentration (*stateUpdaters.FakeStateUpdate attribute*), 13

D

DEs() (*Model.Model method*), 3

F

FakeInputs (*class in inputters*), 9
FakeStateUpdate (*class in stateUpdaters*), 13
finalise() (*in module labview*), 8
fx (*StateEstimator.StateEstimator attribute*), 5

G

G_rpm_to_ml_min (*inputters.LabviewInputs attribute*), 11
get_data() (*inputters.LabviewInputs method*), 12
get_data() (*Model.Model method*), 4
get_data() (*StateEstimator.StateEstimator method*), 6
get_data() (*stateUpdaters.FakeStateUpdate method*), 14
get_data() (*stateUpdaters.LabviewStateUpdate method*), 15
get_deviations() (*StateEstimator.StateEstimator method*), 6
get_glucose_graph() (*in module labview*), 8
get_times() (*stateUpdaters.FakeStateUpdate method*), 14
get_times() (*stateUpdaters.LabviewStateUpdate method*), 15
get_update() (*stateUpdaters.FakeStateUpdate method*), 13
get_update() (*stateUpdaters.LabviewStateUpdate method*), 15
get_Xs() (*Model.Model method*), 4
get_Xs() (*StateEstimator.StateEstimator method*), 6
glucose (*inputters.FakeInputs attribute*), 9

H

hx() (*StateEstimator.StateEstimator static method*), 6

I

init() (*in module labview*), 8
inputs (*inputters.LabviewInputs attribute*), 11
inputs (*labview.Labview attribute*), 7
inputs (*Model.Model attribute*), 3
inputs (*StateEstimator.StateEstimator attribute*), 5

L

Labview (*class in labview*), 7
LabviewInputs (*class in inputters*), 11
LabviewStateUpdate (*class in stateUpdaters*), 15
live_plot (*labview.Labview attribute*), 7

M

m (*labview.Labview attribute*), 7
M_rpm_to_ml_min (*inputters.LabviewInputs attribute*), 12
Model (*class in Model*), 3

N

N_rpm_to_ml_min (*inputters.LabviewInputs attribute*), 11
nx (*StateEstimator.StateEstimator attribute*), 5

O

outputs() (*Model.Model method*), 4

P

pH_calculations (*Model.Model attribute*), 3
plot_all() (*in module plotting*), 17
plot_data() (*in module plotting*), 17
plot_live() (*in module plotting*), 17
plot_model() (*in module plotting*), 17

Q

Q (*StateEstimator.StateEstimator attribute*), 5
Q_fact (*inputters.LabviewInputs attribute*), 12

R

R (*StateEstimator.StateEstimator attribute*), 5
rate_matrix_inv (*Model.Model attribute*), 3

S

se (*labview.Labview attribute*), 7
sigmas (*StateEstimator.StateEstimator attribute*), 5
StateEstimator (*class in StateEstimator*), 5
step() (*in module labview*), 8
step() (*Model.Model method*), 4
step() (*StateEstimator.StateEstimator method*), 6
step() (*stateUpdaters.FakeStateUpdate method*), 13
step() (*stateUpdaters.LabviewStateUpdate method*), 15
su (*labview.Labview attribute*), 7

T

t (*labview.Labview attribute*), 7
t (*Model.Model attribute*), 3
t (*StateEstimator.StateEstimator attribute*), 6
t (*stateUpdaters.FakeStateUpdate attribute*), 13
T_amb (*inputters.LabviewInputs attribute*), 12
t_next_predict (*StateEstimator.StateEstimator attribute*), 6
t_next_predicts (*StateEstimator.StateEstimator attribute*), 6
t_predict (*labview.Labview attribute*), 7
t_predict (*StateEstimator.StateEstimator attribute*), 6
ts (*inputters.LabviewInputs attribute*), 11
ts (*labview.Labview attribute*), 7
ts (*StateEstimator.StateEstimator attribute*), 5
ts (*stateUpdaters.LabviewStateUpdate attribute*), 15

U

ukf (*StateEstimator.StateEstimator attribute*), 5
update (*stateUpdaters.LabviewStateUpdate attribute*), 15
update() (*inputters.LabviewInputs method*), 12
update() (*StateEstimator.StateEstimator method*), 6
update_inputs() (*in module labview*), 8
update_ready() (*stateUpdaters.FakeStateUpdate method*), 13
update_ready() (*stateUpdaters.LabviewStateUpdate method*), 15
update_state() (*in module labview*), 8
update_time (*stateUpdaters.LabviewStateUpdate attribute*), 15
update_value (*stateUpdaters.LabviewStateUpdate attribute*), 15
update_values (*stateUpdaters.LabviewStateUpdate attribute*), 15

X

X (*Model.Model attribute*), 3
X0 (*labview.Labview attribute*), 7
Xs (*labview.Labview attribute*), 7