Time Delay

(Report #2)

Date 2018-1-14

# 

# **Objectives**

# The objective of this experiment is to find the best method for calculating the time delay between series of datasets. There were six different type of datasets tested, each of them had been added 5, 15 and 50 sample delays during experiments. All of them were tested by five delay finding method. After collecting all the result, and calculating the Mean Square Error, we can find the best method for calculating the delay.

**Design**

There are five different types of methods for finding delay:

1. Cross-correlation method (CORR)
2. Coherence method (CSD)
3. Auto Regressive model with eXtra input (ARX)
4. Output Error model (OE)
5. MET

The cross-correlation method is to find the maximum cross-correlation of an input dataset and its output dataset with ith delay:

For the coherence method, both of input and output are transferred into frequency domain values. After that, correlation for frequency domain (f\_z) can be calculated also the single-side spectrum of datasets can be found. Then the estimated frequency is determined by 1. The frequency with largest amplitude on single-side spectrum and 2. The value where the magnitude squared coherence has the largest or second largest value. At last the delay is equal to the angle of the frequency where the f\_z has the largest value divide by 2pif. (The angle can be found by calculating the cross power spectral density of datasets):

For the Auto Regression model with eXtra input, its model can be represented by:

The nk denotes different time-delay, Prediction Error Method is here equal to Least Squares Estimation and the delay is whose model has the lowest loss function.

The principle of the method Output Error is similar as the ARX but the OE model:

are estimated instead of ARX models. The estimation of OE is much more computationally demanding than ARX model.

The MET method is aiming to imitate OE method but with much lower computational demands.

* Pseudo codes of each method is provided in Appendix.

To test the capability of above methods, datasets were designed from easiest random noise to more complexed cases. For this report, six different types of datasets were used to test delay methods:

1. A random data set input passed through a delay and a noise adder.
2. A sinewave input gets decayed overtime and passed through a delay and a noise adder
3. Similar with second case, but the input was changed to be addition of multiple sinewaves.
4. Fourteen sets of TE output data collected from the Simulink, and add delay on them.
5. A step input passed through an open loop system with delay.
6. Similar with the fifth case, but the closed loop system.

(please see Appendix for simulations)

For open and close loop system (dataset 5 and 6), a step input passed through slow, medium and fast PI controller then delay and a system to generate dataset.

**Test results and Discussion**

Each of the testing cases having three different time-delay added: 5, 15 and 50 samples of delay. Because of the noise added are different every time runs the program, for consistency, 10 times of the iteration have run for any of the case. The results are following.

Table1: Delay calculating result

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | corr | csd | oes | arx | met1 | csd1 | csd2 |
| delay for datasets | delay rand noise | |  |  |  |  |  |
| 5 | 5 | 0.903723 | 5 | 2.1 | 5 | 1.191243 | 1.191243 |
| 15 | 20 | 1.970935 | 20 | 13.7 | 20 | 4.188987 | 4.188987 |
| 50 | 70 | 4.990073 | 70 | 60.5 | 36.4 | 5.470975 | 5.470975 |
| MSE | 6.871843 | 15.67882 | 6.871843 | 3.656805 | 4.82999885 | 15.32688 | 15.32688 |
|  | delay multi sin | |  |  |  |  |  |
| 5 | 5 | 16.73761 | 3.5 | 10.6 | 1 | 59.60164 | 57.22902 |
| 15 | 20 | 39.7283 | 17.3 | 25.3 | 2 | 120.7273 | 124.5463 |
| 50 | 70 | 98.54553 | 66.3 | 77.3 | 21.5 | 197.2013 | 184.1517 |
| MSE | 6.871843 | 18.57695 | 5.50989 | 9.903647 | 10.52642178 | 63.09414 | 60.30011 |
|  | delay sin damp | |  |  |  |  |  |
| 5 | 5 | 4.834708 | 34.8 | 10.9 | 0 | 5.00337 | 5.00337 |
| 15 | 20 | 20.5914 | 77.3 | 28.4 | 0 | 22.42023 | 22.42023 |
| 50 | 70 | 37.27376 | 102.2 | 79.8 | 0 | 39.05048 | 39.05048 |
| MSE | 6.871843 | 4.633791 | 28.85629 | 11.06752 | 17.48014747 | 4.408979 | 4.408979 |
|  | **openloop with step input,slow PI** | | | |  |  |  |
| 5 | 19.5 | Inf | 10.5 | 53.2 | 16.9 | 0 | 0 |
| 15 | 52.9 | Inf | 28.8 | 97 | 34.8 | 0 | 0 |
| 50 | 90.8 | Inf | 47 | 153.1 | 53.2 | 0 | 0 |
| MSE | 19.1813 | #VALUE! | 5.051842 | 46.75807 | 7.773816452 |  |  |
|  | **openloop with step input,medium PI** | | | |  |  |  |
| 5 | 29.5 | Inf | 2.5 | 51.5 | 13 | 0 | 0 |
| 15 | 73 | Inf | 4 | 95 | 29.2 | 0 | 0 |
| 50 | 112.4 | Inf | 19.3 | 134.7 | 47.9 | 0 | 0 |
| MSE | 29.54847 | #VALUE! | 10.90229 | 41.81486 | 5.477732702 |  |  |
|  | **openloop with step input, fast PI** | | | |  |  |  |
| 5 | 17.1 | Inf | 20.7 | 47.9 | 16.8 | 0 | 0 |
| 15 | 45.1 | Inf | 27.5 | 91.4 | 34.8 | 0 | 0 |
| 50 | 83.1 | Inf | 32.9 | 144 | 54.2 | 0 | 0 |
| MSE | 15.44895 | #VALUE! | 8.788566 | 42.83479 | 7.809680602 |  |  |
|  | **closeloop with step input, slow PI** | | | |  |  |  |
| 5 | 16.1 | Inf | 7 | 46.4 | 16.2 | 0 | 0 |
| 15 | 41.1 | Inf | 21.2 | 99.3 | 32.7 | 0 | 0 |
| 50 | 81.3 | Inf | 33.2 | 142.7 | 51.7 | 0 | 0 |
| MSE | 14.07958 | #VALUE! | 6.006293 | 43.98704 | 7.004918907 |  |  |
|  | **closeloop with step input, medium PI** | | | |  |  |  |
| 5 | 25.1 | Inf | 3.1 | 31.8 | 12.8 | 0 | 0 |
| 15 | 66.5 | Inf | 6.9 | 77.6 | 27.4 | 0 | 0 |
| 50 | 106.8 | Inf | 17.6 | 116.5 | 45.1 | 0 | 0 |
| MSE | 26.42074 | #VALUE! | 11.15039 | 31.7267 | 5.149002061 |  |  |
|  | **closeloop with step input, fast PI** | | |  |  |  |  |
| 5 | 11.3 | Inf | 7.5 | 33.2 | 15.6 | 0 | 0 |
| 15 | 36.1 | Inf | 16.1 | 82.7 | 32.7 | 0 | 0 |
| 50 | 75.4 | Inf | 21.1 | 126.1 | 50.3 | 0 | 0 |
| MSE | 11.20546 | #VALUE! | 9.67626 | 35.229 | 6.877822653 |  |  |
|  |  |  |  |  |  |  |  |
| average MSE | 15.16667 | #VALUE! | 10.31263 | 29.66427 | 8.103282387 |  |  |

Table 2: fourteen TE datasets with 5 samples delay



Table 3: fourteen TE datasets with 15 samples delay



Table 4: fourteen TE datasets with 15 samples delay



Table 5: averaging the result from OE and MET

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ave oe met | |  |  |  |
| open slow | | 5.777453 | Systems open and close | MSE |
| open medium | | 5.568587 | OE | 8.597054397 |
| open fast |  | 7.389839 | MET | 6.682162229 |
| close slow | | 5.200053 | Average OE MET result | 6.063068302 |
| close medium | | 6.334627 |  |  |
| close fast |  | 6.10785 |  |  |
| close slow | |  |  |  |

There are three different csd columns, because there are three ways to estimate frequency by csd method, but csd method does not performing well, so they can be ignored. In addition, the reason that result contains INF and zeros in table one, because for those datasets inputs are step input.

The Correlation method worked well for some of cases, because for those cases, delays are applied directly on to the input data therefore Correlation method has good performance. While looking at the open-loop and closed-loop cases, the Correlation method is not reliable.

OE and MET methods, both are having decent performance, the OE had closer result while delay is small, and the MET had the better result while the delay is large. Since OE method is very time consuming, MET should be the choice if the computation time is considered. If the computation time is not considered as problem, calculating both methods and find it average will be recommended. From the Table 5, OE and MET method had MSE for open and close loop systems datasets 8.59 and 6.68. However, while averaging the delay calculated by OE and MET than do the MSE calculation, the result got better, which is 6.06.

* Future experiments:

1. As the transfer function can be anything, there can be much more experiments be done later by choosing different systems.
2. After numbers of closed loop systems get tested, it is more likely to say which method works best for a typical kind of data set.

**Conclusions**

In this experiment, five different method of calculating time-delay was applied on six different types of datasets. The OE and MET method had very good performance, but combining their result would get the best result overall. If any time a new dataset is asked to find the time-delay, trying to calculate the average of OE and MET would be the best way, but if the time consumption is considered, MET method will also get a decent result

# **Appendix**

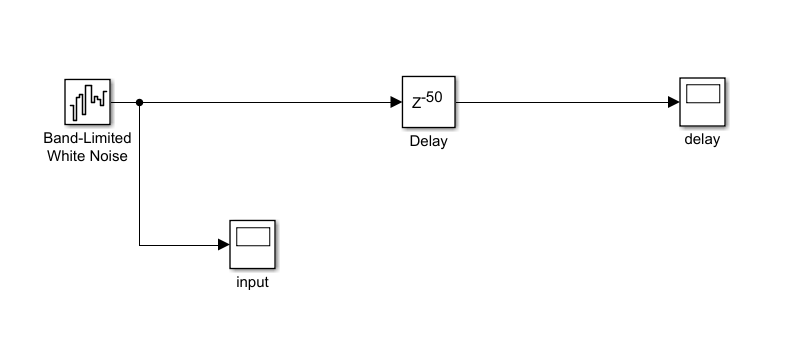


Figure 1: Random noise Data generator

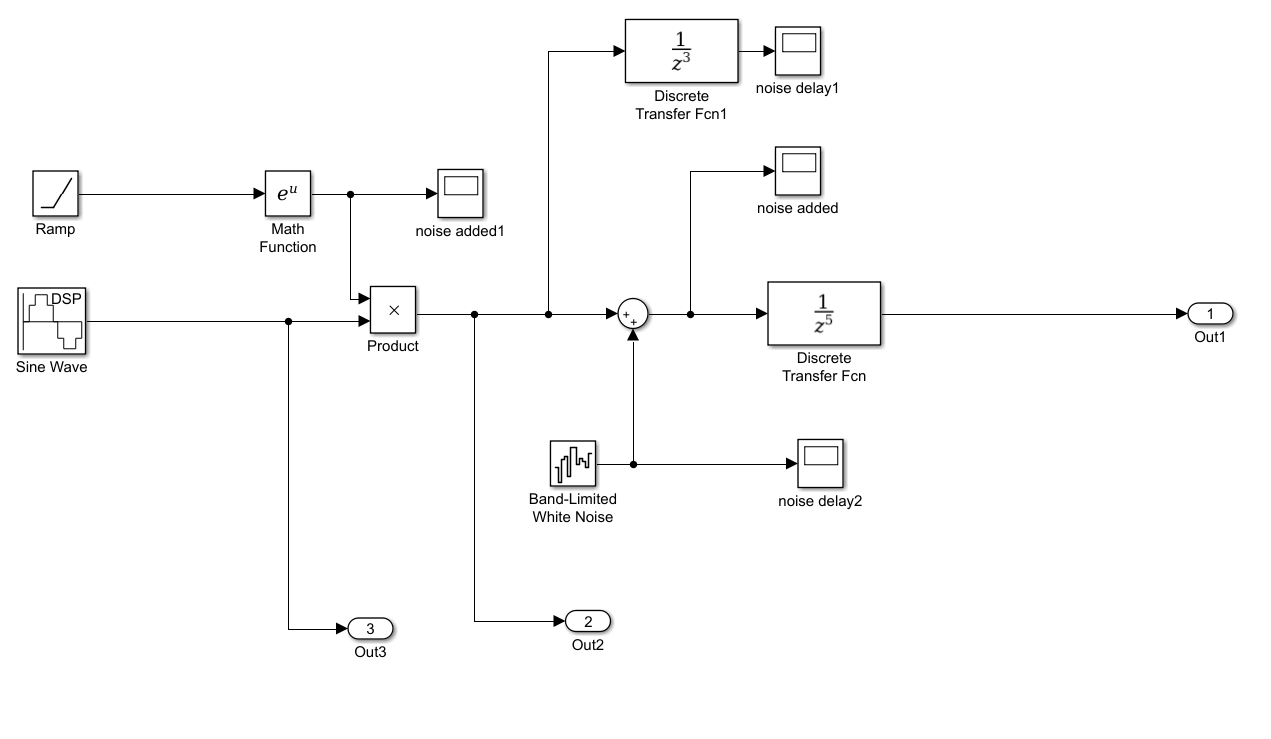


Figure 2: Sin Damping Data generator

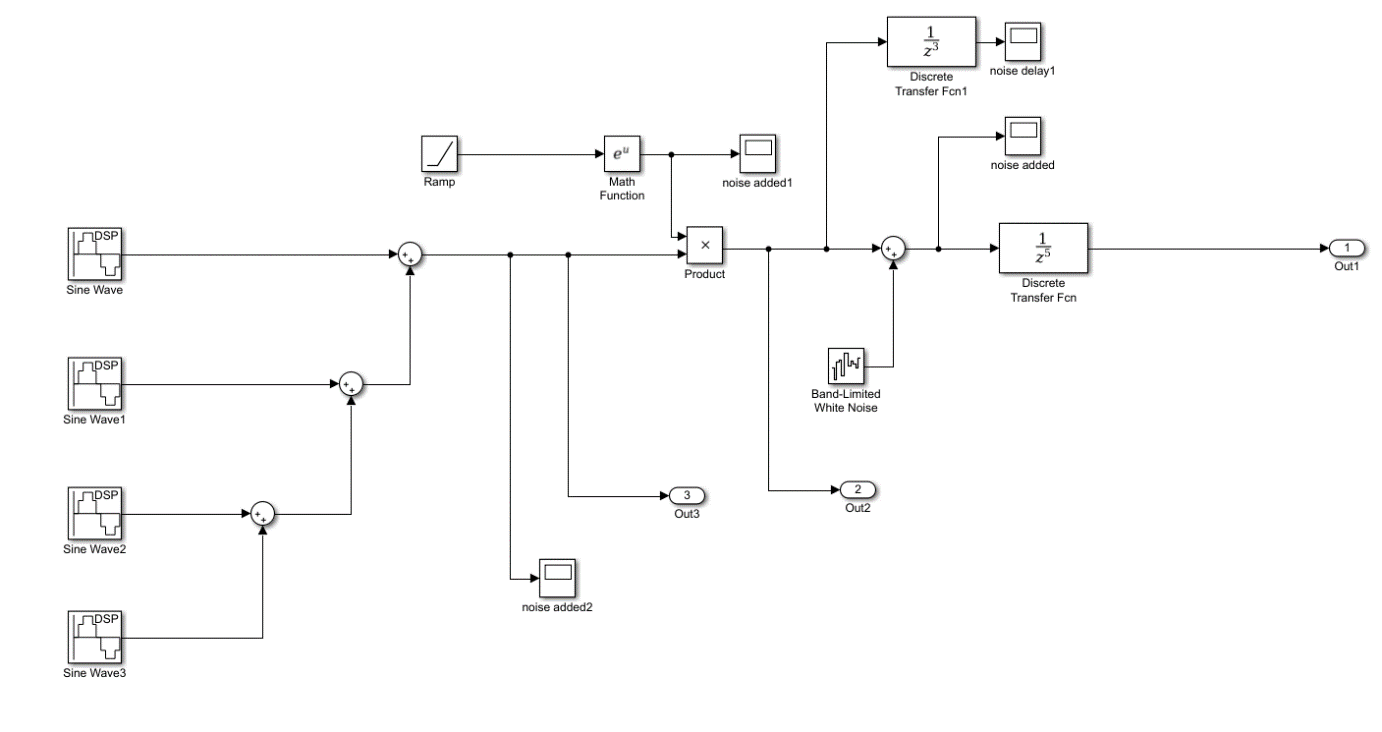


Figure 3: Multiple sin damping Data generator

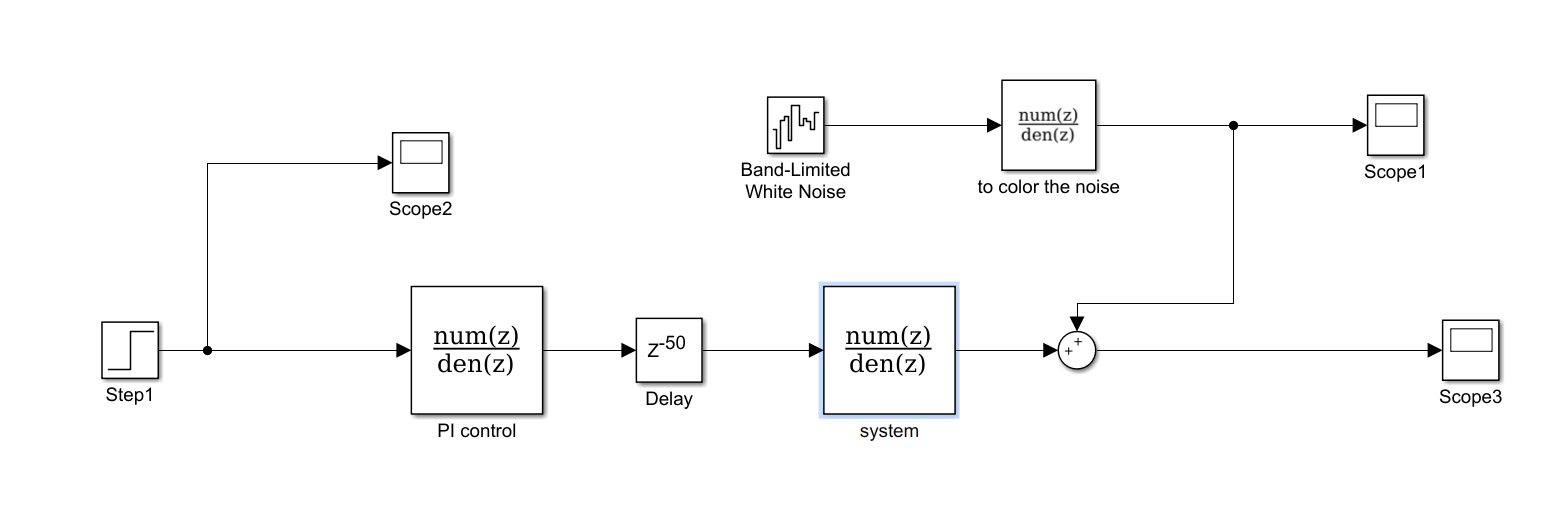


Figure 2: Open loop Data generator

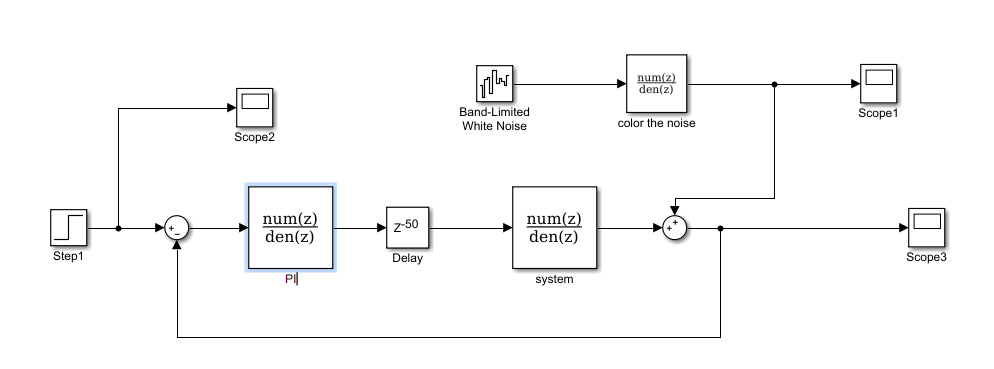


Figure 2: Closed loop Data generator

Pseudo code:

1. Correlation method:

Program corr\_method(input, output)

k = length of input and output

for i =1 to k

c[k] = correlations for (Inuput[k:end] and Output)

k\_max = index of the largest c[k]

delay = k\_max

1. Coherence method:

Program csd\_method(input,output)

zero mean the input and output

I\_f = Fourier transferred input

I\_o = Fourier transferred output

k = length of I\_f and I\_o

for i = 1 to k

c[k] = correlations in frequency domain for (I\_f[k:end] and I\_o)

k\_max = index of the largest |c[k]|

phi = angle of the correlation at i\_max

delay = phi/(2\*pi\*frequency)

1. Auto Regressive model with eXtra input

Program arxstructd(input,output)

define: na – Order of polynomial A(q)

nb – Order of polynomial B(q) + 1

nk – range of fixed leading zeros of the B polynomial

nn = [na,nb,nk]

loss function V = arxstruct([output,input],nn)

order of ARX nn\_sel = selstruct(V)

delay = nn\_sel[3]

1. Output Error model

Program oestructd(input,output)

define: nb – Order of polynomial B(q) +1

nf – Order of polynomial F(q)

nk – range of fixed leading zeros of the B polynomial

for i = 1 to length(nk)

nn[i] = [nb,nf,nk[i]]

estimated system: Model = oe([output,input],nn[i])

loss function V[i] = Model.NoiseVariance

k\_min = index of the minimum V[i]

delay = nk[k\_min]

1. MET

Program met1structd(input,output)

define:

order: order of estimated model

modelSs = estimate state space model: n4sig(output,input ..)

[A,B,C,D,E] = coefficient of polynomial modal of modelSs

uFilt = filter the input by C

yFilt = filter the output by C

do arxstructed(uFilt,yFilt)