System Identification Tool: Software Specification

1 Introduction

Four years ago I listened to one student presenting a work on Volterra Series based process model. This work was not destined to any future implementation, as the monopoly of convolution based input-output model used by QP based linear MPCs would not use it.

So the goal of this work is to have a framework, which can let a process control researcher who has thought of a new form of model specification for MPC, to quickly configure that model from plant data file and start using it in MHEMPC. APC implementers spend most of their time interacting with process model identification software.

Inputs shall be pure text files, output shall be text files containing related model parameters.

How that generic behavior can be implemented will be explored in the future and it may take few years to evolve a system.

Current work shall begin with convolution based input-output model, as it is the simplest. This will help develop a robust User Interface (UI) protocol, based on mouse and keyboard inputs, to create such models that can address all possible real-life situations that an implementer faces.

Subsequently this is referred to as 'Id tool'. Every type of MPC will have a corresponding Id tool. Switching between different types of MPCs is easy if the Id tool can handle all mathematical forms of process modeling in the same framework/workbench. This is the goal of this specification.

2 Use Cases

User can use this Id tool in various ways as documented here. Use cases help manage design of big software.

2.1 Definitions

User has a plant with **n** outputs, **m** inputs. They are also known as CVs and MVs/DVs, dependent variables and independent variables, effects and causes.

Inputs are of two types, manipulable and disturbance. A signal from computer can change a manipulative input in a deterministic way, like a setpoint to a flow control. Any other cause that cannot be commanded for proper behavior is a disturbance, like the ambient temperature. Sometimes, a manipulable input loses its good behavior and changes to a disturbance, like a choked slurry flow.

Regulatory controls are not plant control. Instrument air pressure can be changed to vary the opening of a control valve, which in turn changes the flow and thus the orifice meter reading changes. This is the classic regulatory control which is single input single output (SISO) in nature. To install, configure and put it on closed loop is the function of an instrumentation/electrical engineer. The resulting regulatory controller is given to the chemical engineer to fix its setpoint.

Several such regulatory controllers are considered by the chemical engineer to run the plant by choosing the best combination of their individual setpoints, so that stable and optimal production is achieved. Here the concept of multiple inputs come in, their combined effects are observed and this makes the plant a multiple input multiple output (MIMO) system.

MIMO systems are not SISO instrumentation. They are plant control. Sometimes, the same physical plant may have serial sections, each having a set of inputs and associated outputs disjoint from other sections making smaller MIMO systems. Reaction and separation systems like FCCU are examples. Each section should have a separate MIMO plant control. Though it is mathematically possible to combine them into one large super-MIMO, it is illogical.

Regulatory controls are much faster than plant controls, so never eliminate any. MIMO does not substitute SISO, they are two different layers.

Tests are the only way to confirm that there exists a deterministic and reproducible relationship between a cause and an effect, so that plant control can be configured based on this relationship, often called model. Obtaining this model by parameter estimation is Process Identification.

Step test is conducted on the real physical plant by changing one cause and watching all effects. In a multivariable way, any sequence of cause-changing can be accommodated, signal-noise ratio of measurement can restrict choices. Disturbance inputs change on their own, manipulable inputs are changed by changing their respective regulatory control setpoints.

A change in a regulatory control setpoint changes the actuator in due course of time causing a lag dynamics between manipulable input setpoint and its process. So both measurements are required. In the case of a disturbance input that does not have a setpoint, just pad with a zero placeholder column.

Lags also exist between process input and outputs. Unique to chemical processes are dead-time. Inverse response happens when sudden bubbles form inside vessels from increase in heat input and the liquid level goes up. After a time these bubbles collapse and net change level is opposite to initial response. At different operating points same process shows different process gains and dynamics, making the system nonlinear.

2.2 Use Case 0

- User lists inputs U1 to Um, outputs Y1 to Yn tentative. A plant with a given name is this bi-list.
- Step tests conducted and data saved in a text file. First line: noOfSamples and plantName, Second line: sampleNo, column headers, Y1 Y2 … Yn U1 U2 … Um US1 US2 … USm
- Data lines correspond to the headers from sampleNo=1 to noOfSamples
- Filename is auto-generated combining date-time-PlantName
- Id tool begins with ReadFile. File Name is displayed at right top corner.
- Start sample and end sample can be changed in editbox
- One click identification writes to scratch
- Scratch is viewed, chosen to make final model, save in a model text file
- Model file can be opened by controller

2.3 Use Case 1

- In the middle of Use Case 0, a scratch file created, cannot be model for few bad relations
- Make another test, use subset plant to re-find bad portions, save as another model
- Read one to scratch, then copy to model, then the other. Combine both models

2.4 Use Case 2

- Same as Use Case 1, during 2nd test something else unwanted moved, use subset with ‘model backoff’