

NARMA-L2 Controller

NARMA discrete-time model:

$$y(k+d) = M[y(k), y(k-1), \dots, y(k-n+1), u(k), u(k-1), \dots, u(k-n+1)]$$

Non Linear

Auto-Regressive

Moving Average

Narendra e Mukhopahyay, 1997: NARMA L2 Norm Model

$$y(k+d) = f[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-m+1)] \\ + g[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-m+1)] \cdot u(k)$$

Control Law:

$$u(k) = \frac{y_r(k+d) - f[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-n+1)]}{g[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-n+1)]}$$

Applying the Control Law to the NARMA-LW model results:

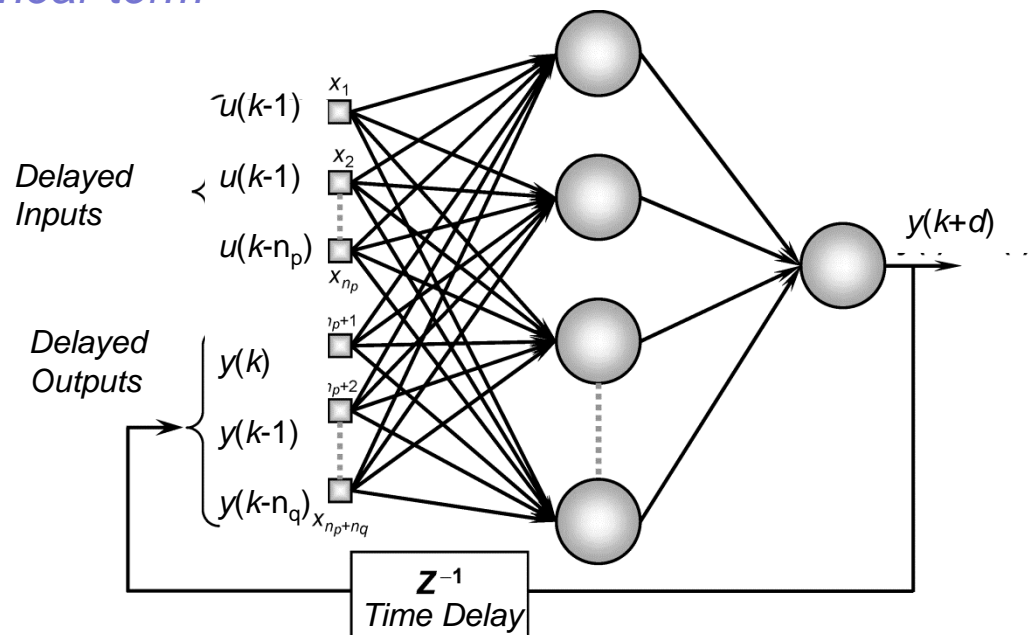
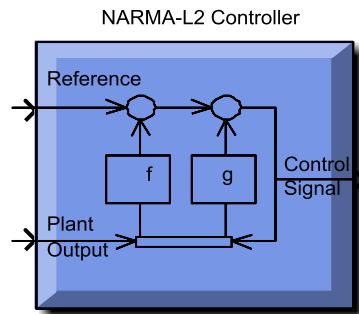
$$y_r(k+d) \equiv y(k+d)$$

y follows y_r exactly!!

NARMA-L2 Controller

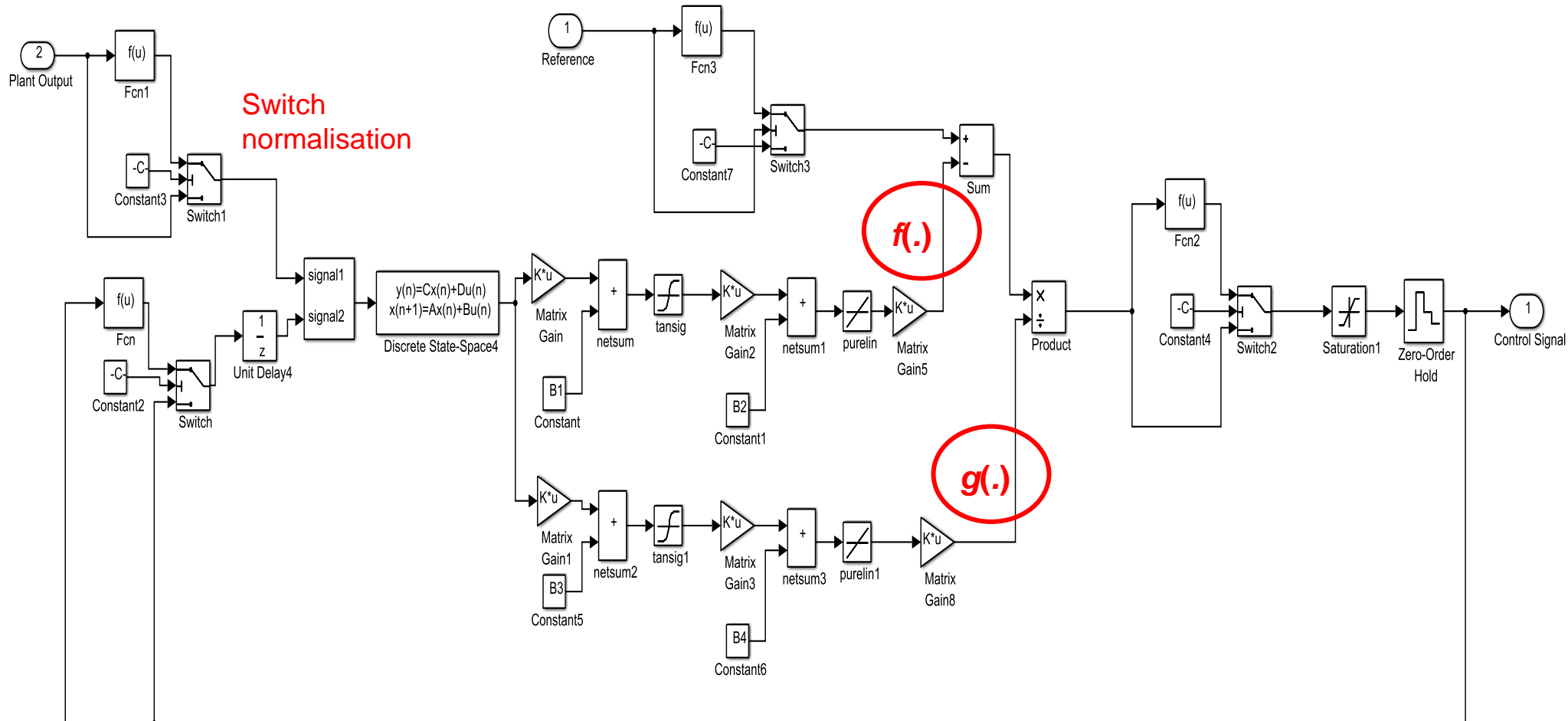
$$u(k) = \frac{y_r(k+d) - f[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-n+1)]}{g[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-n+1)]}$$

ANN can be used to learn $f(\cdot)$: the additive non-linear term and $g(\cdot)$ the multiplicative non-linear term

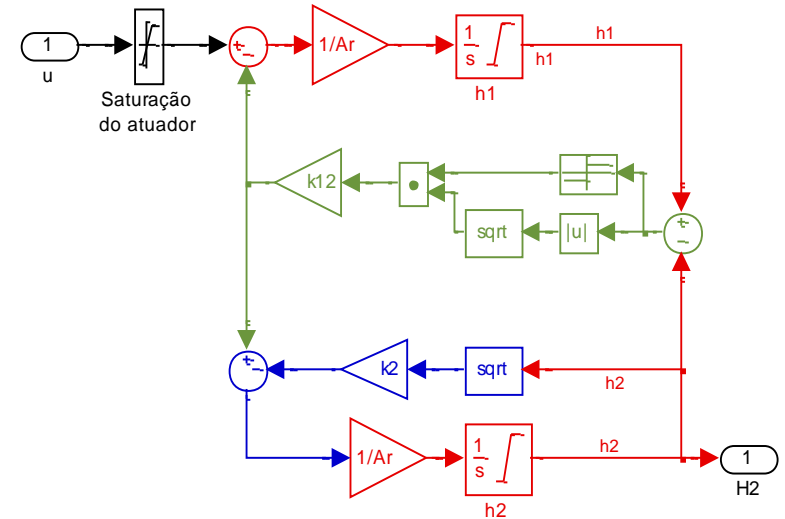
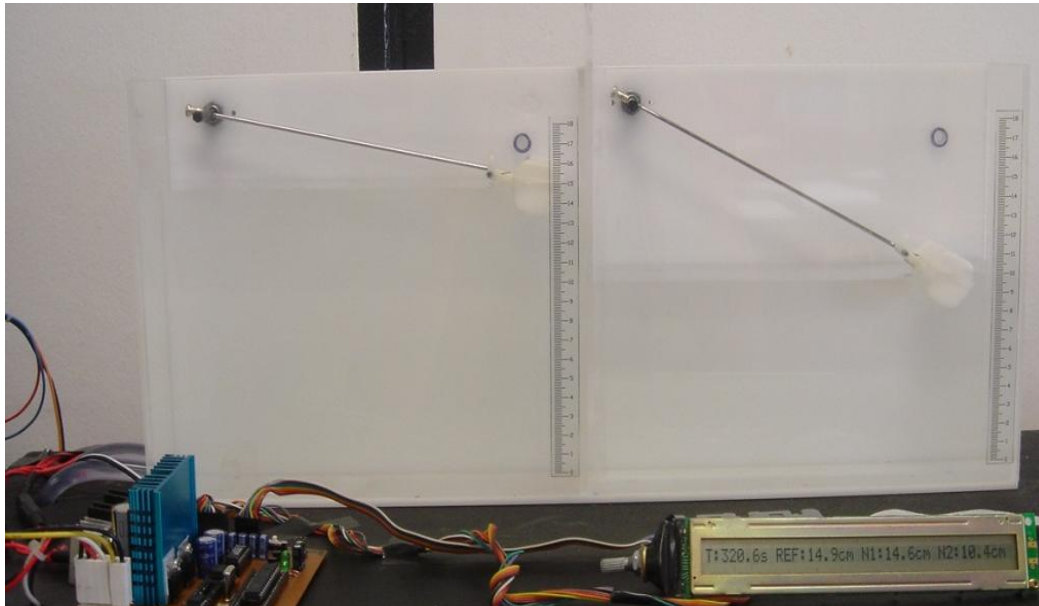


$$y(k+d) = M[y(k), y(k-1), \dots, y(k-n+1), u(k), u(k-1), \dots, u(k-n+1)]$$

NARMA-L2 controller

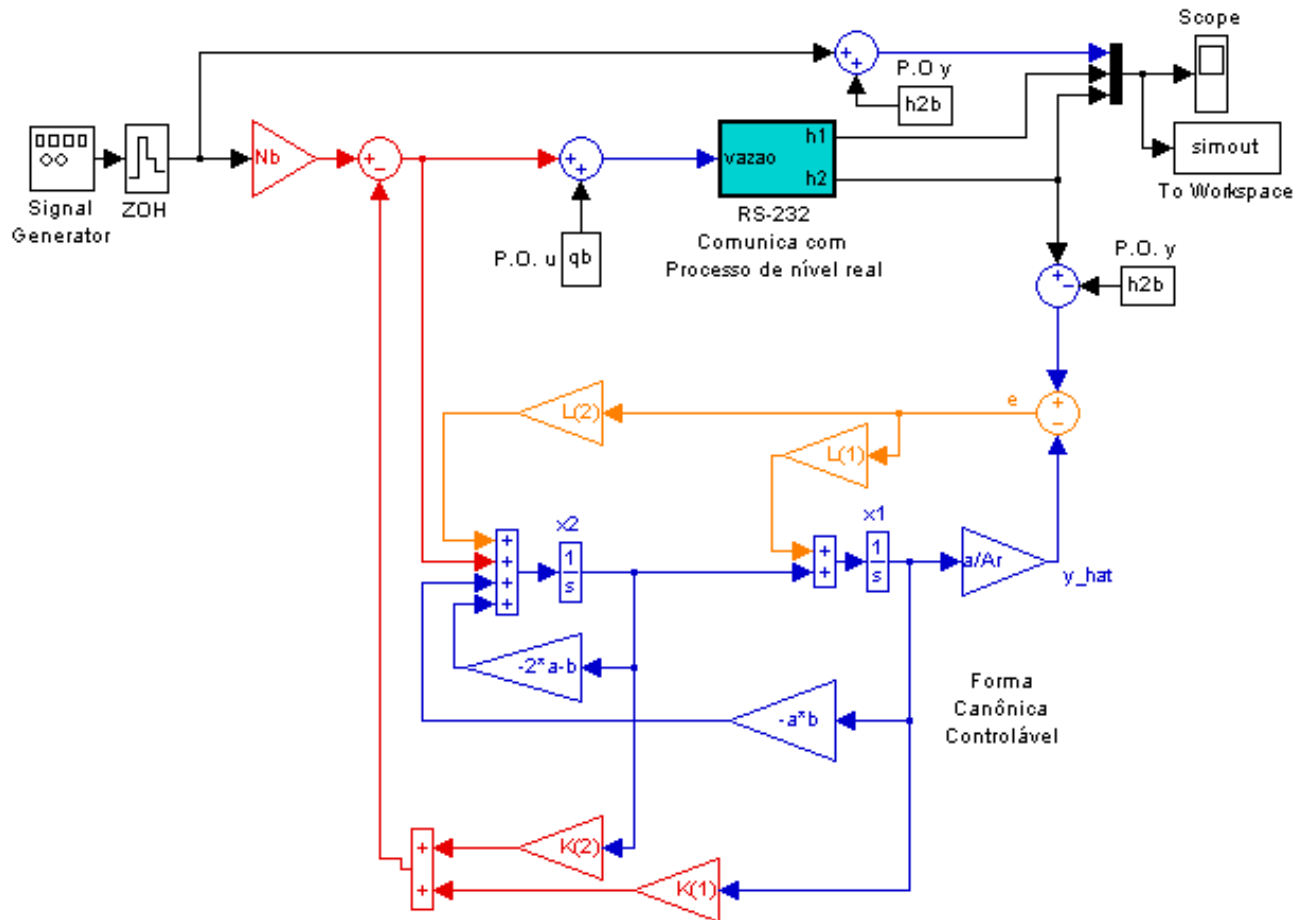


Ex: Liquid Level Control



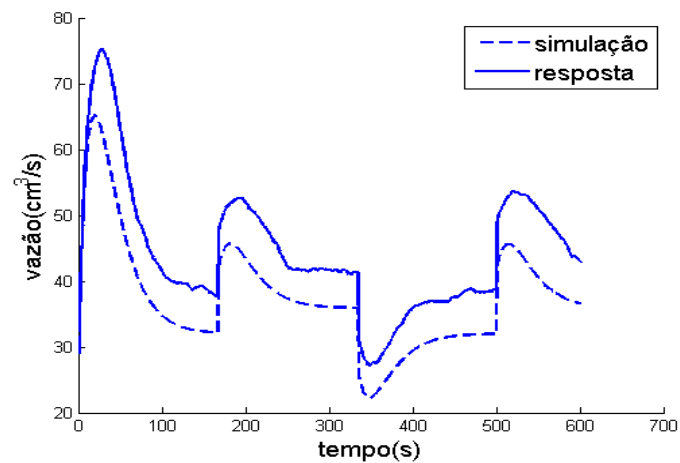
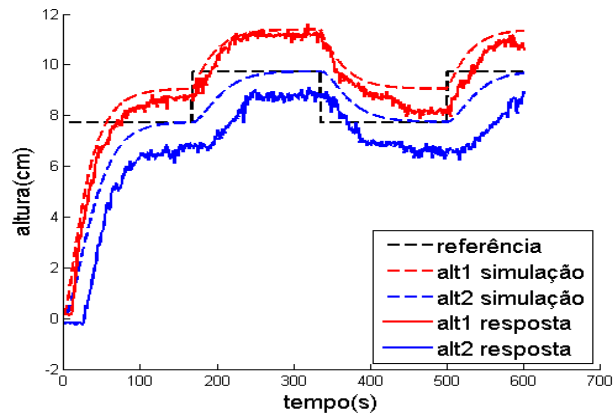
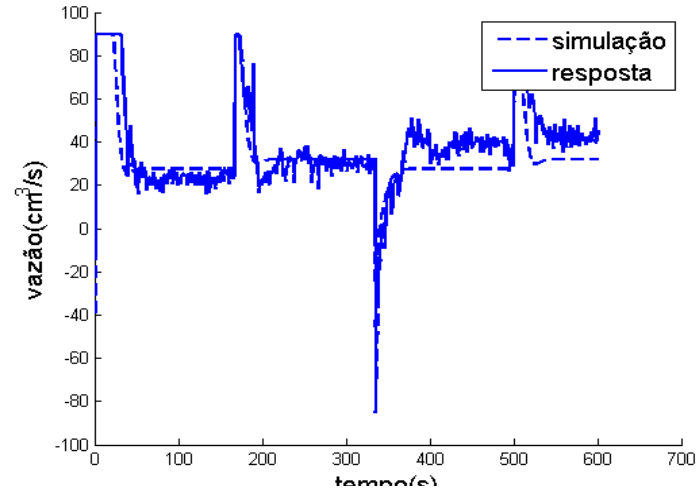
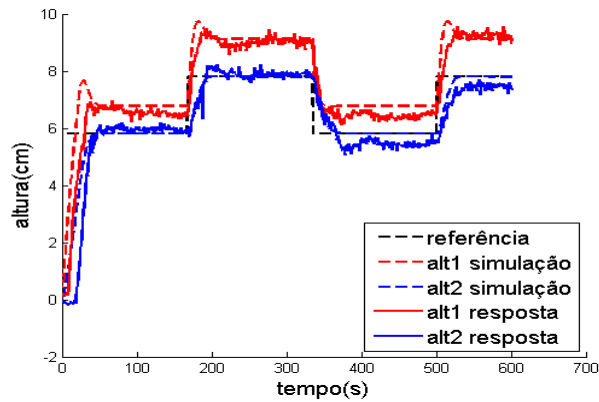
$$\begin{cases} A \frac{dh_1}{dt} = q_i - k_{12} \sqrt{h_1 - h_2} \\ A \frac{dh_2}{dt} = k_{12} \sqrt{h_1 - h_2} - k_2 \sqrt{h_2} \end{cases}$$

Integrated Interface to Process



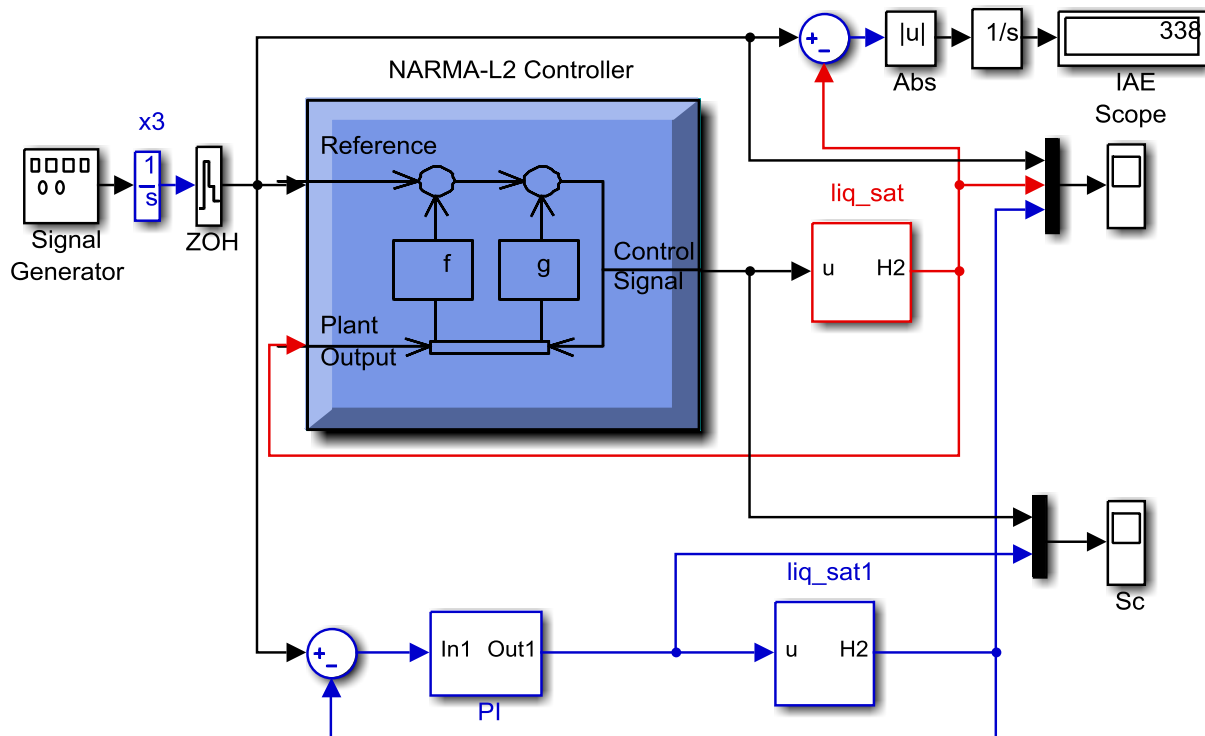
State Space Controller – Designed for a specific Operating Point

State-Space Results



NARMA-L2Liquid Level Control

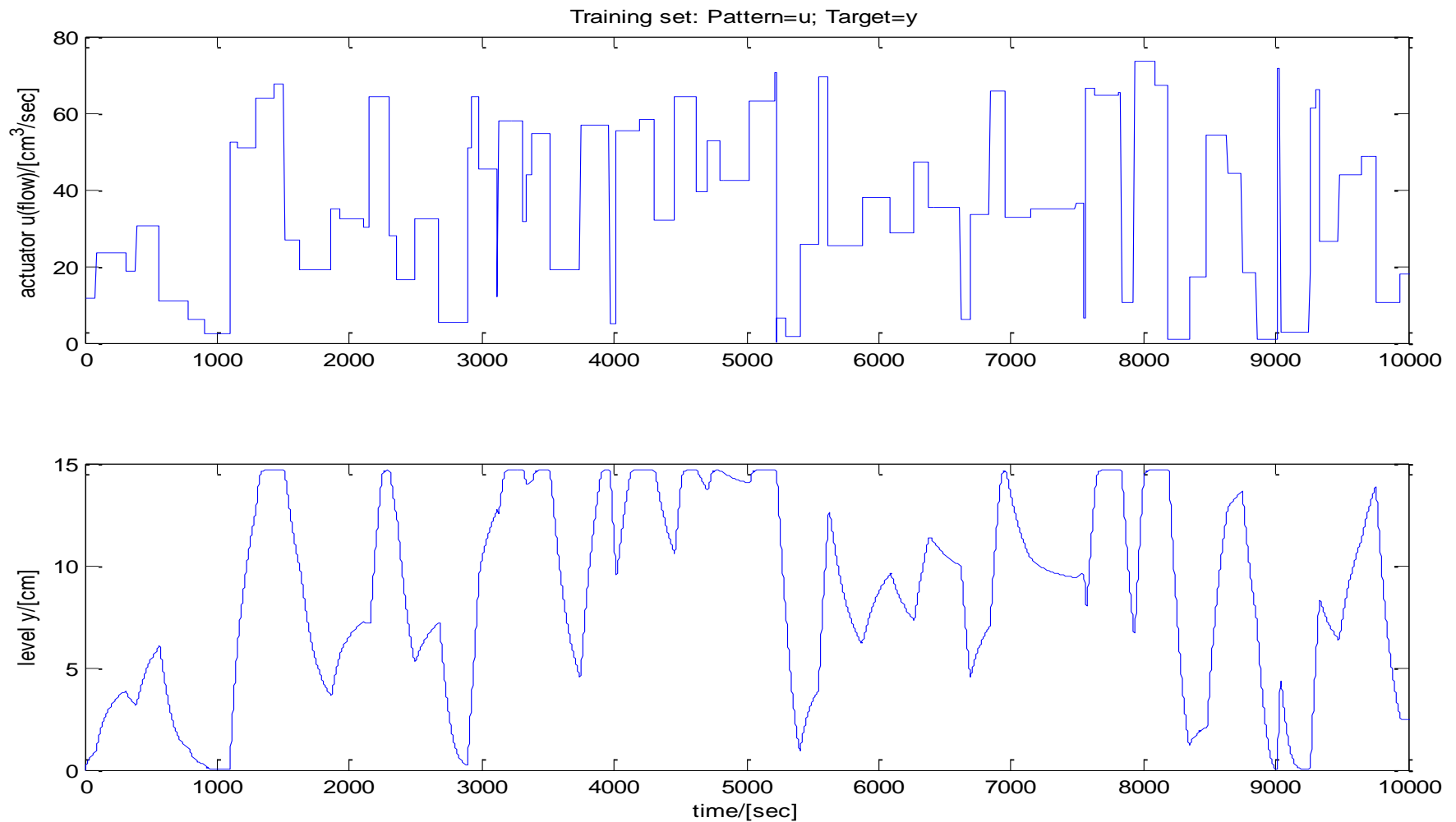
Purpose: compare PI and NARMA-L2 control.



Because of the square root in the model a different behaviour is expected for different reference levels!

Generated Training Sets

Cover the time domain range (dynamics) and amplitude range (operating points).



NARMA-L2 tool

Plant Identification - NARMA-L2

File Window Help

Plant Identification - NARMA-L2

Network Architecture

Size of Hidden Layer: 20 No. Delayed Plant Inputs: 3

Sampling Interval (sec): 1 No. Delayed Plant Outputs: 3

Normalize Training Data

Training Data

Training Samples: 30000 Limit Output Data

Maximum Plant Input: 90 Maximum Plant Output: Inf

Minimum Plant Input: 0 Minimum Plant Output: -Inf

Maximum Interval Value (sec): 250 Simulink Plant Model: Browse

Minimum Interval Value (sec): 1 mod_liq2

Generate Training Data Import Data Export Data

Training Parameters

Training Epochs: 500 Training Function: trainlm

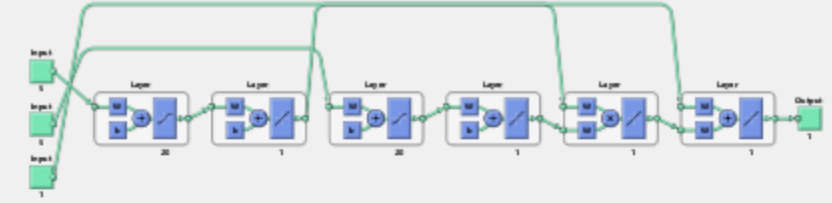
Use Current Weights Use Validation Data Use Testing Data

Train Network OK Cancel Apply

Generate or import data before training the neural network plant.

Neural Network Training (nntraintool)

Neural Network



Algorithms

Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)

Progress

Epoch:	0	34 iterations	500
Time:		0:00:36	
Performance:	1.40e-06	1.34e-06	0.00
Gradient:	0.00773	0.00130	1.00e-10
Mu:	0.00100	1.00e-06	1.00e+10
Validation Checks:	0	0	6

Plots

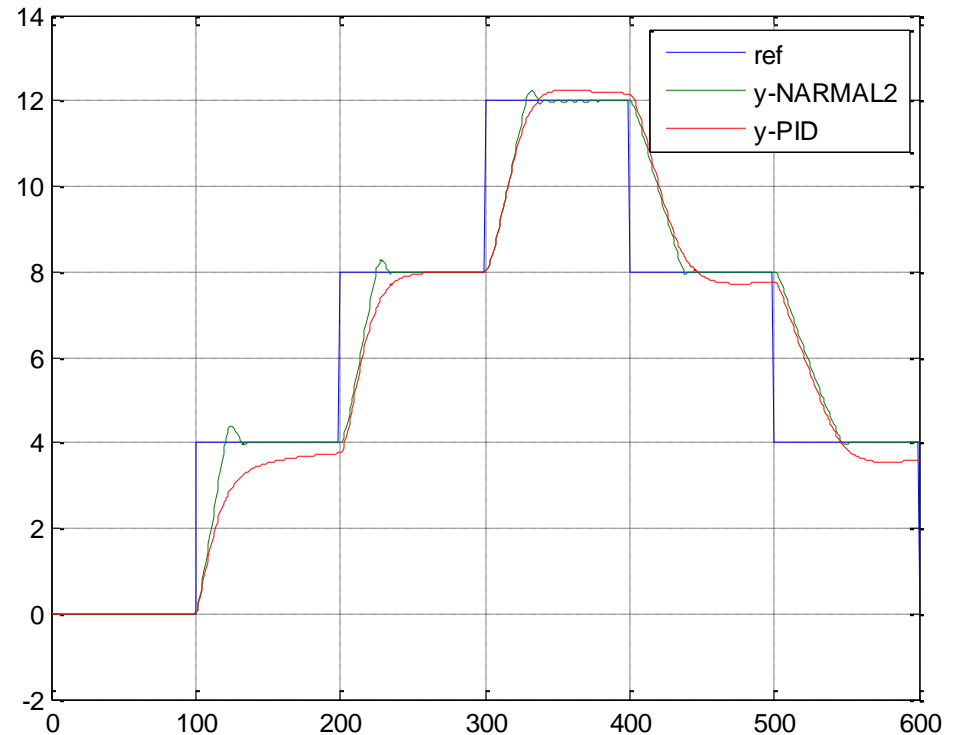
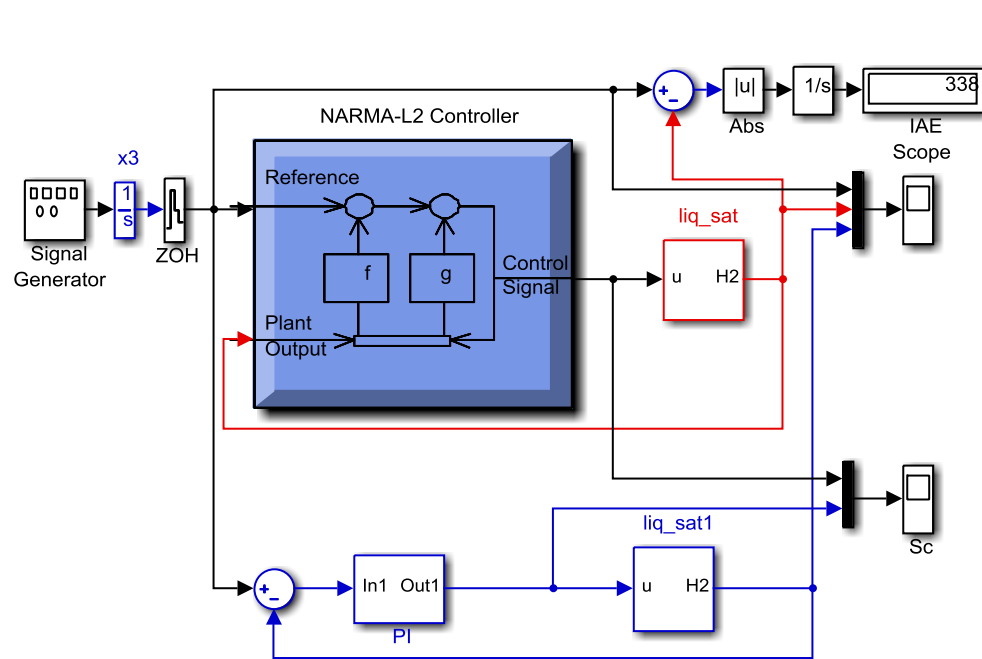
Performance (plotperform)
Training State (plottrainstate)
Regression (plotregression)

Plot Interval: 1 epochs

Training neural network...

Stop Training Cancel

NARMA-L2 Liquid Level Control



- PI gives no error in steady state, but with very different dynamics.
- NARMA-L2 has almost the same behaviour in all operating points.

PI vs NARMA-L2 Control Signal

- NARMA-L2 uses often the maximum available u .
- PI calculate signals that are clamped by the saturation.
- NARMA-L2 tends to “chattering”
- NARMA-L2 design parameters:
 - sampling rate
 - input/output delays
 - training sets
 - training algorithms

NARMA-L2 can be trained with real data!!
Real processes are quite more complex than models!

NARMA-L2 vs PI control Liquid Level

