Colorado State University

Machine Learning and Accelerators: Neural Network Based Modeling and Control at ASTA

Auralee L. Morin

2nd Annual ASTA Users Meeting

9-10 June, 2014

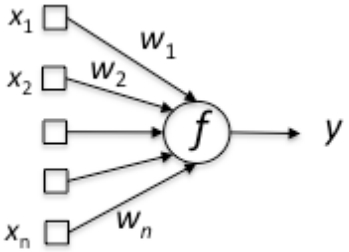
Electrical & Computer ENGINEERING

CSU: Auralee Morin (Graduate Student), Sandra Biedron, Stephen Milton (Faculty)

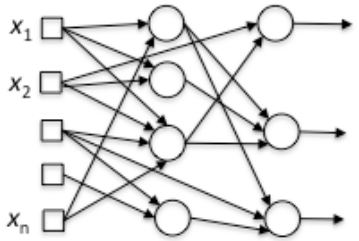
Fermilab: Brian Chase, Pierpaolo Stabile, and many others

Nonlinear Modeling Using Neural Networks

- **What are neural networks?**



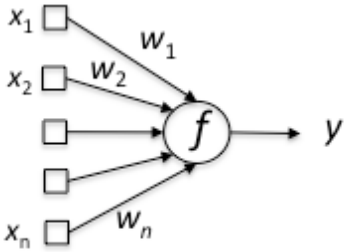
a neuron (node)



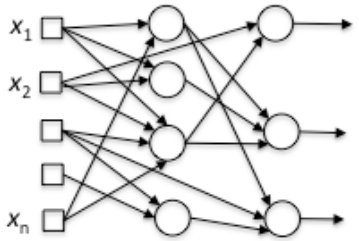
a neural network

Nonlinear Modeling Using Neural Networks

- **What are neural networks?**



a neuron (node)

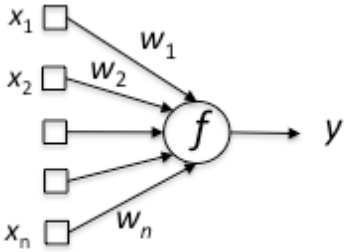


a neural network

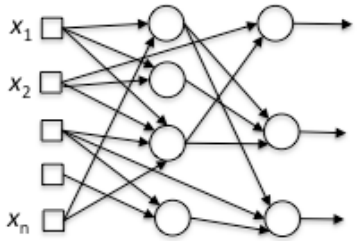
- **When are they useful?**
  - When there is a lot of available data
  - When system dynamics are
    - not well-known, or change over time
    - not captured sufficiently well by analytic models
    - are very complicated/time-consuming to model analytically (but could be in principle)
  - When fast I-O is needed for a complicated model

Nonlinear Modeling Using Neural Networks

What are neural networks?



a neuron (node)



a neural network

When are they useful?

- When there is a lot of available data
- When system dynamics are
  - not well-known, or change over time
  - not captured sufficiently well by analytic models
  - are very complicated/time-consuming to model analytically (but could be in principle)
- When fast I-O is needed for a complicated model

What are the disadvantages?

- Poor generalization
- Computationally intensive to train
- Long, iterative design procedure
- Good design requires substantial expert knowledge

Control Using Neural Networks

- **Two major classes of approaches**
  - NN model based
  - Reinforcement learning/ adaptive critic

Control Using Neural Networks

- **Two major classes of approaches**
  - NN model based
  - Reinforcement learning/ adaptive critic
- **Where are they being pursued?**
  - Lots of interest in industrial process control
  - Lots of academic papers on simple experiments, but still rarely used

Control Using Neural Networks

- **Two major classes of approaches**
  - NN model based
  - Reinforcement learning/ adaptive critic
- **Where are they being pursued?**
  - Lots of interest in industrial process control
  - Lots of academic papers on simple experiments, but still rarely used
- **Major hurdles**
  - Achieving stability requires significant tradeoffs (and is usually not mathematically guaranteed)
  - Adaptive NN-based control is computationally intensive
  - Often, the most appealing solutions rely on controls techniques that have not yet reached maturity
  - Lack of prior track record on complicated systems

Control Using Neural Networks

- **Two major classes of approaches**
  - NN model based
  - Reinforcement learning/ adaptive critic
- **Where are they being pursued?**
  - Lots of interest in industrial process control
  - Lots of academic papers on simple experiments, but still rarely used
- **Major hurdles**
  - Achieving stability requires significant tradeoffs (and is usually not mathematically guaranteed)
  - Adaptive NN-based control is computationally intensive
  - Often, the most appealing solutions rely on controls techniques that have not yet reached maturity
  - Lack of prior track record on complicated systems

So, why bother?

Control Using Neural Networks

- **Two major classes of approaches**
  - NN model based
  - Reinforcement learning/ adaptive critic
- **Where are they being pursued?**
  - Lots of interest in industrial process control
  - Lots of academic papers on simple experiments, but still rarely used
- **Major hurdles**
  - Achieving stability requires significant tradeoffs (and is usually not mathematically guaranteed)
  - Adaptive NN-based control is computationally intensive
  - Often, the most appealing solutions rely on controls techniques that have not yet reached maturity
  - Lack of prior track record on complicated systems

So, why bother?

→ **Starting to look promising for improving the control of complicated MIMO processes**

→ **Not a panacea, but very useful when applied intelligently**

Initial Studies at ASTA

- **Start simple → NN temperature control of the RF gun**

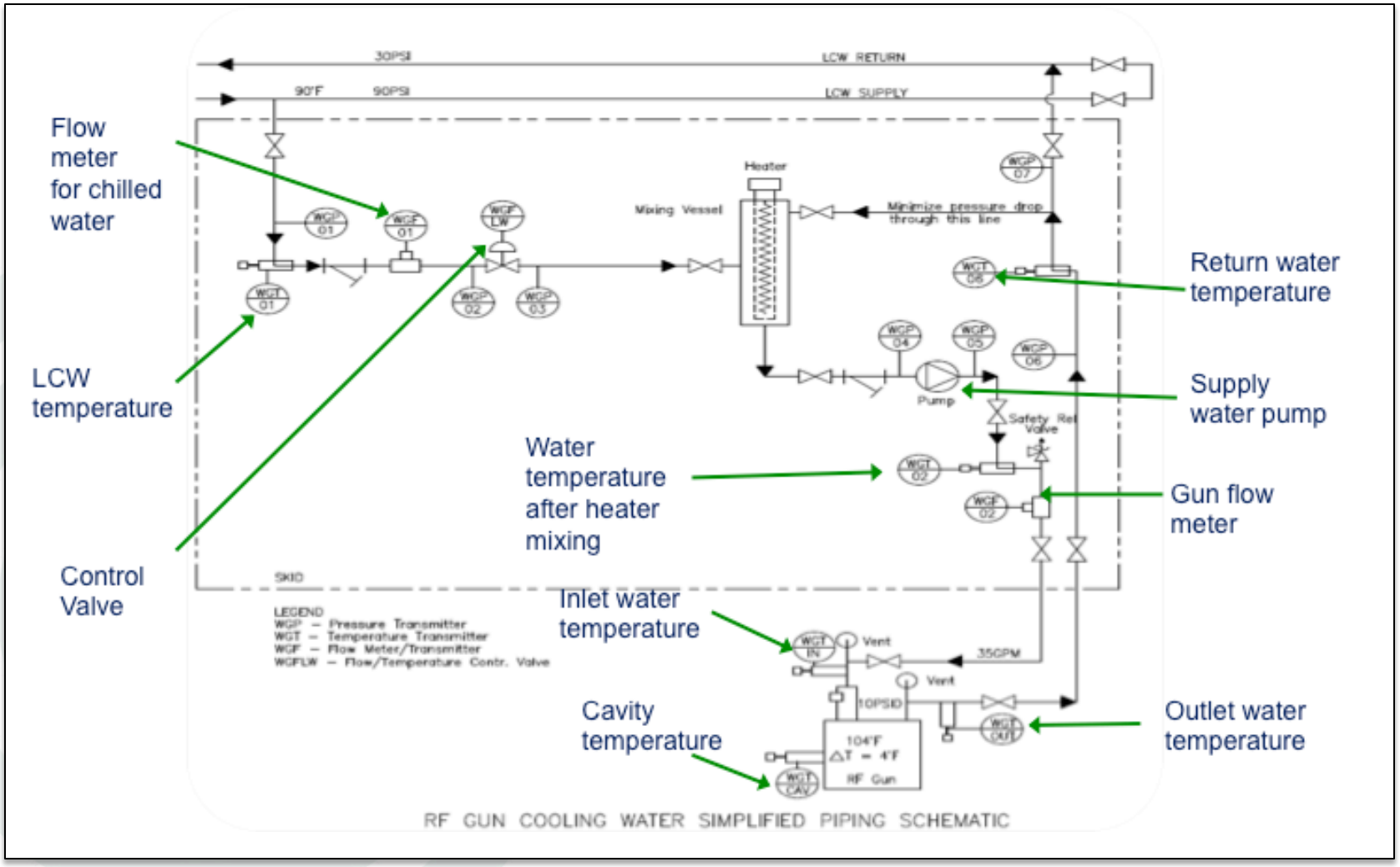


Figure courtesy P. Stabile

Model Design: What is Involved

Model Design: What is Involved

- **Design training/validation/testing data**

Model Design: What is Involved

- Design training/validation/testing data
- Determine what input parameters to include

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**
- **Design an appropriate adaptation procedure, if applicable**

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**
- **Design an appropriate adaptation procedure, if applicable**

→ **15-fold validation: especially important if re-training!**

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**
- **Design an appropriate adaptation procedure, if applicable**

→ **15-fold validation: especially important if re-training!**

→ **For a NN model to be useful in control, also need to keep real-time implementation in mind**

  - Speed of communication channels and actuators
  - Speed of adaptation procedures, speed of NN I-O processing
  - Data sampling rate vs. execution speed

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**
- **Design an appropriate adaptation procedure, if applicable**

Number of Candidates	Property
25	dead time
12	embedding
55	hidden layers
10	sampling rate
10	alternate model inputs

112 candidates x 15-fold validation
→ 1680 NN to train

→ 15-fold validation: especially important if re-training!

→ For a NN model to be useful in control, also need to keep real-time implementation in mind

- Speed of communication channels and actuators
- Speed of adaptation procedures, speed of NN I-O processing
- Data sampling rate vs. execution speed

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**
- **Design an appropriate adaptation procedure, if applicable**

Number of Candidates	Property
25	dead time
12	embedding
55	hidden layers
10	sampling rate
10	alternate model inputs

112 candidates x 15-fold validation
→ 1680 NN to train

40 minutes to train (on average)

→ 15-fold validation: especially important if re-training!

→ For a NN model to be useful in control, also need to keep real-time implementation in mind

- Speed of communication channels and actuators
- Speed of adaptation procedures, speed of NN I-O processing
- Data sampling rate vs. execution speed

Model Design: What is Involved

- **Design training/validation/testing data**
- **Determine what input parameters to include**
- **Determine what preprocessing is needed**
  - Standardization and scaling range (e.g. -1 to 1, -0.5 to 0.5)
  - Dimensionality reduction (PCA, CCA), dead-time removal (if desired)
- **Determine which activation functions to use**
- **Determine an appropriate sampling rate for the inputs**
- **Find the appropriate lag space**
- **Find an appropriate topology**
  - number of hidden layers
  - number of nodes in each layer
  - interconnections between nodes
- **Determine which training algorithm to use (and parameters—objective function, learning rate, etc.)**
- **Design an appropriate adaptation procedure, if applicable**

Number of Candidates	Property
25	dead time
12	embedding
55	hidden layers
10	sampling rate
10	alternate model inputs

112 candidates x 15-fold validation
→ 1680 NN to train

40 minutes to train (on average)

47 days

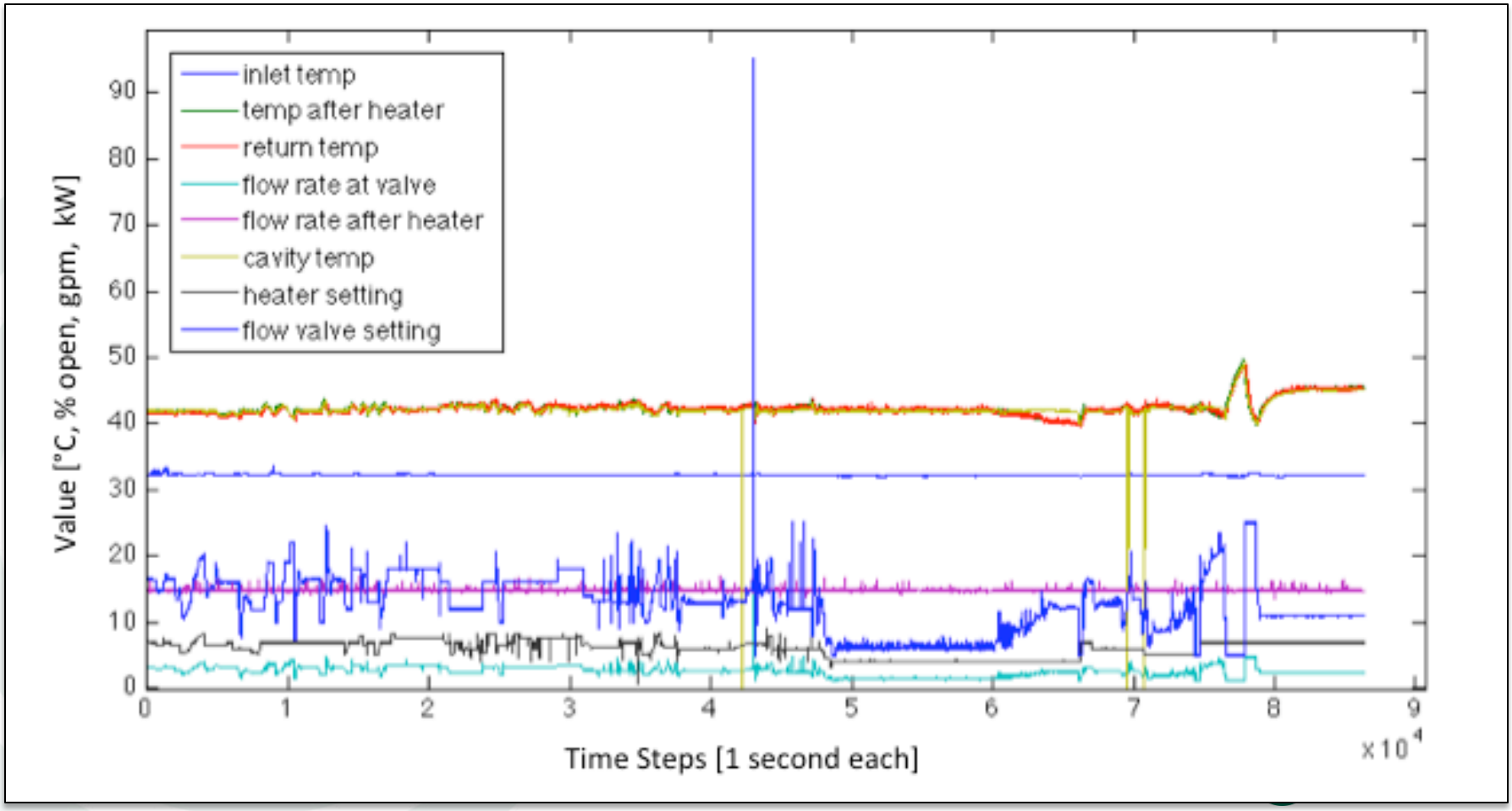
→ 15-fold validation: especially important if re-training!

→ For a NN model to be useful in control, also need to keep real-time implementation in mind

- Speed of communication channels and actuators
- Speed of adaptation procedures, speed of NN I-O processing
- Data sampling rate vs. execution speed

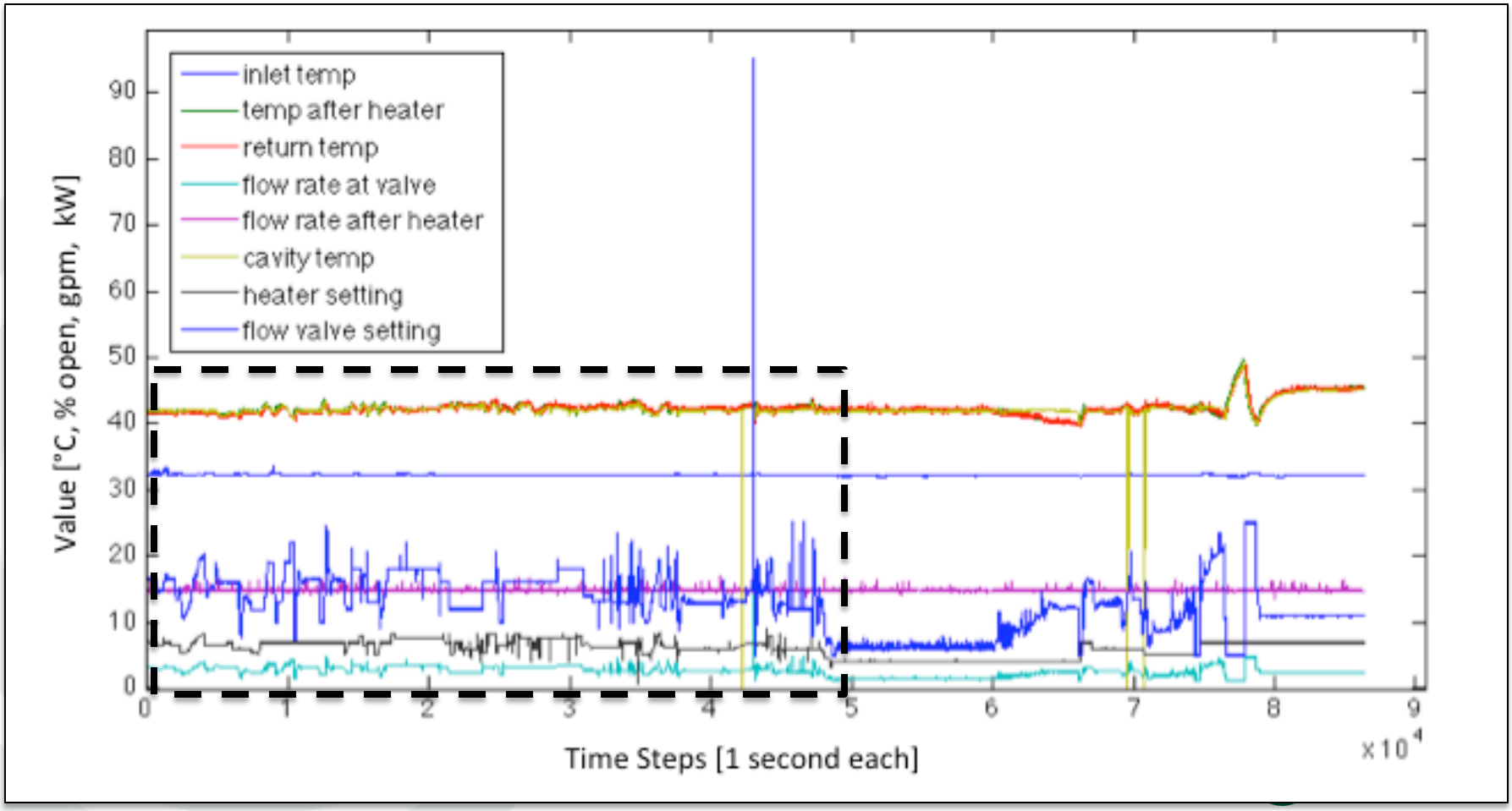
Data Obtained for the Initial Model Design

- ~24 hours of data total



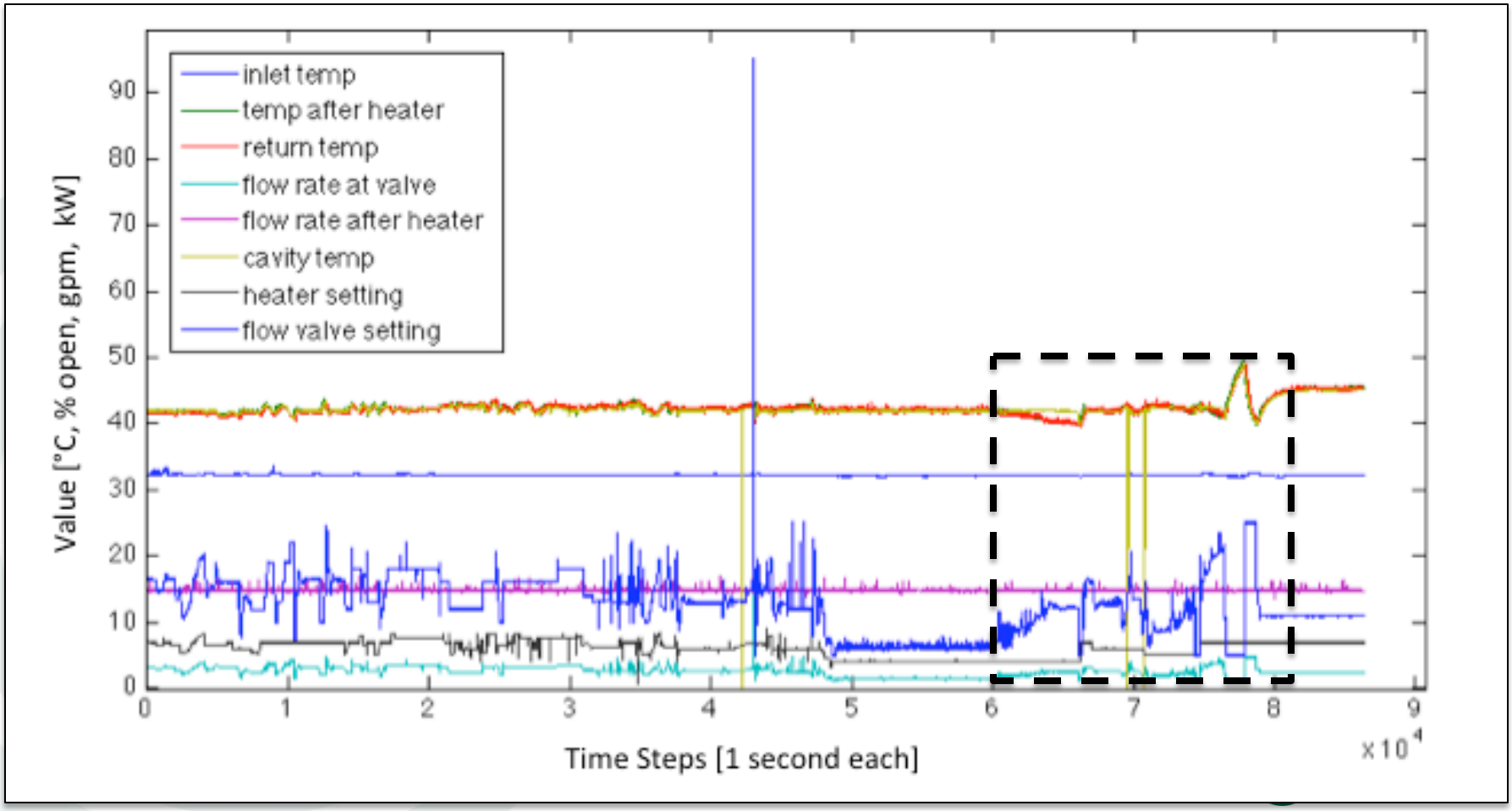
Data Obtained for the Initial Model Design

- ~24 hours of data total
- ~14 hours of pseudo-random input (without regulation)



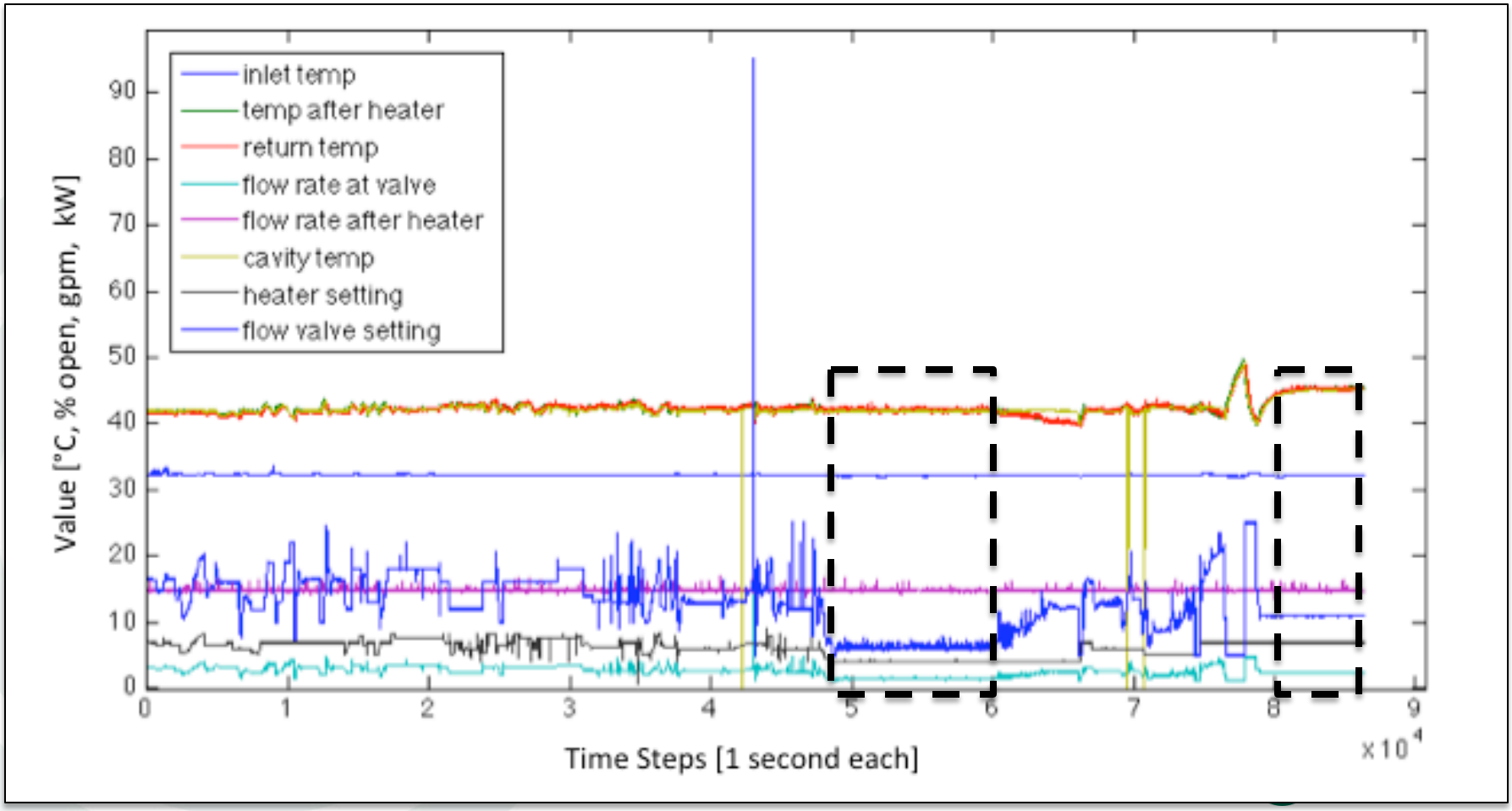
Data Obtained for the Initial Model Design

- ~24 hours of data total
- ~14 hours of pseudo-random input (without regulation)
- ~5 hours with changing RF power to the gun (with regulation) → cavity conditioning

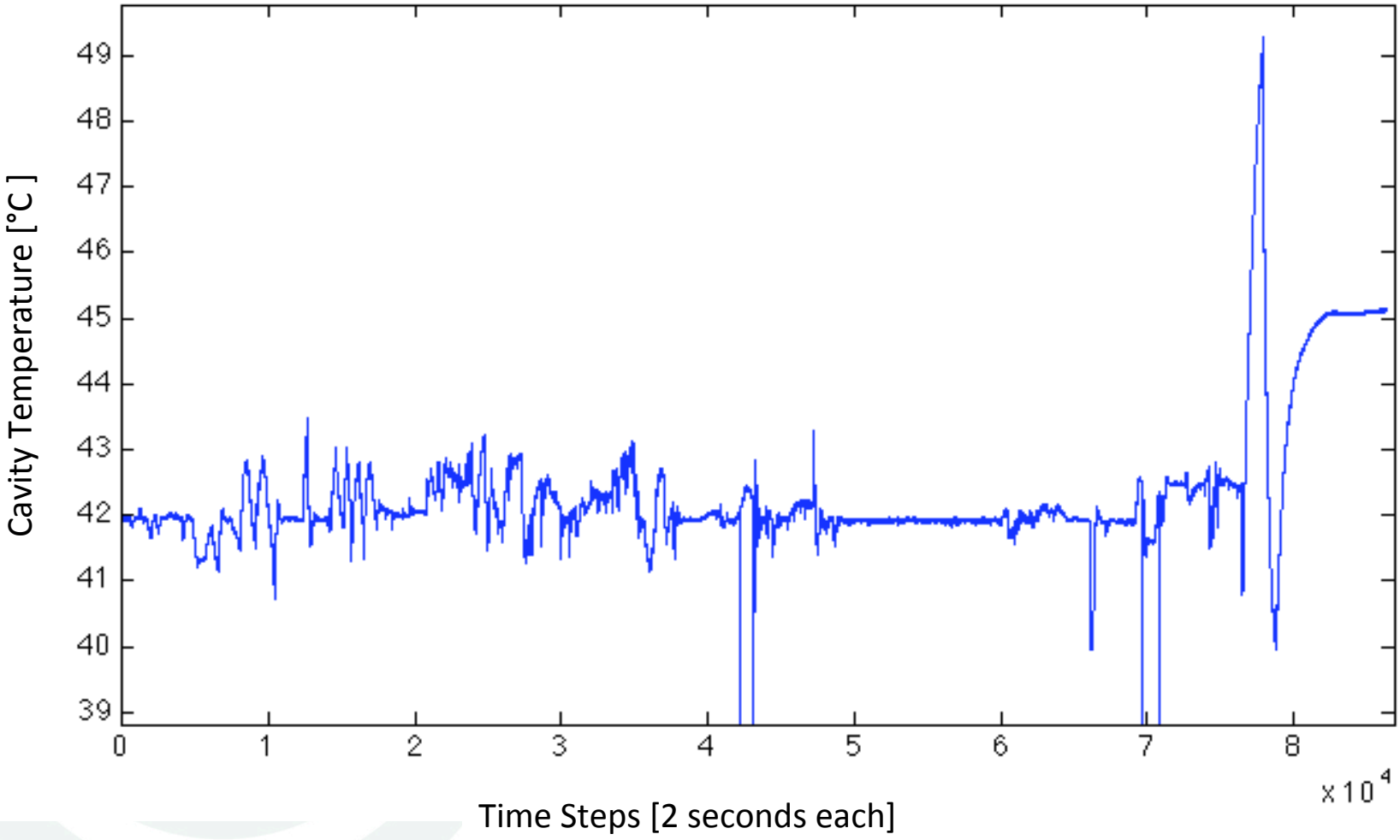


Data Obtained for the Initial Model Design

- ~24 hours of data total
- ~14 hours of pseudo-random input (without regulation)
- ~5 hours with changing RF power to the gun (with regulation) → cavity conditioning
- The rest is steady-state regulation with existing controller



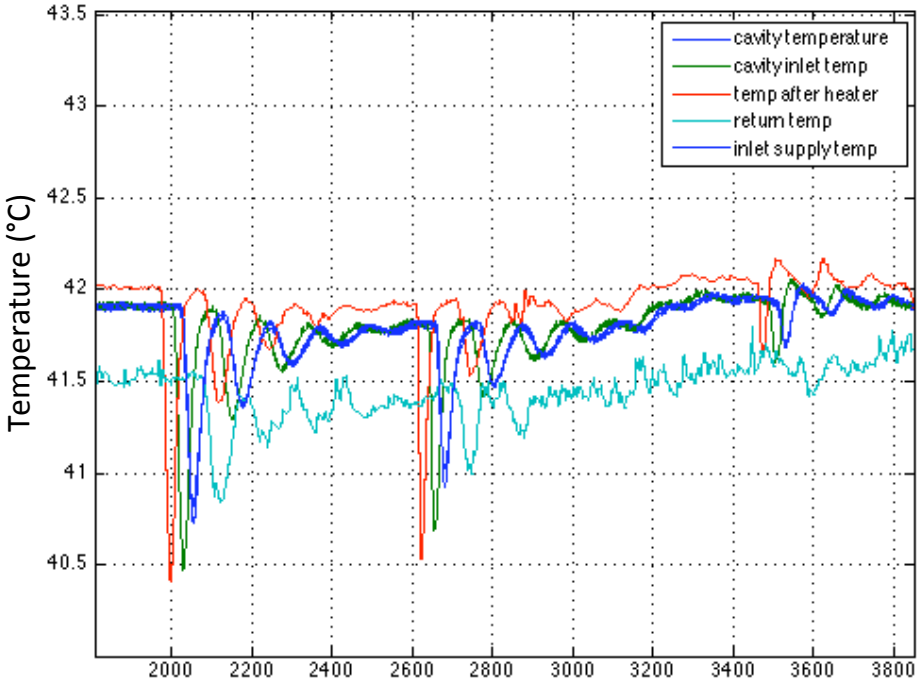
Closer View of Cavity Temperature

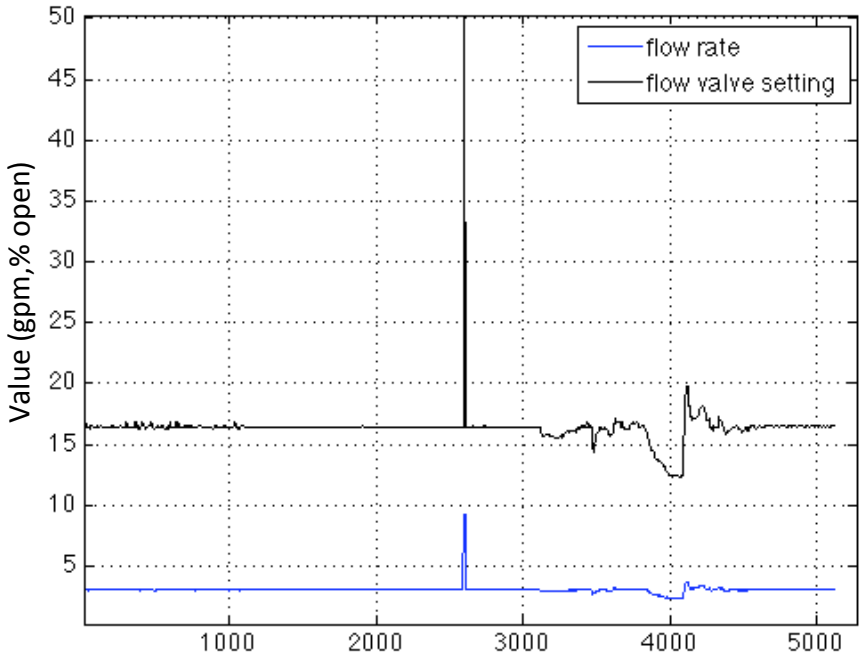


Model Design: Input Parameter Selection

- Choose parameters which make sense *a priori*
- Get rid of redundant parameters
- Where uncertain, investigate candidate models with 15-fold validation

Highly Correlated Parameters





Time Steps (1 second each)

Model Design: Input Parameter Selection

- **Chosen Parameters**
  - Flow Control Valve Setting
  - Heater Setting
  - LCW Temperature
  - → Cavity Temperature

Model Design: Input Parameter Selection

- **Chosen Parameters**
  - Flow Control Valve Setting
  - Heater Setting
  - LCW Temperature
  - → Cavity Temperature

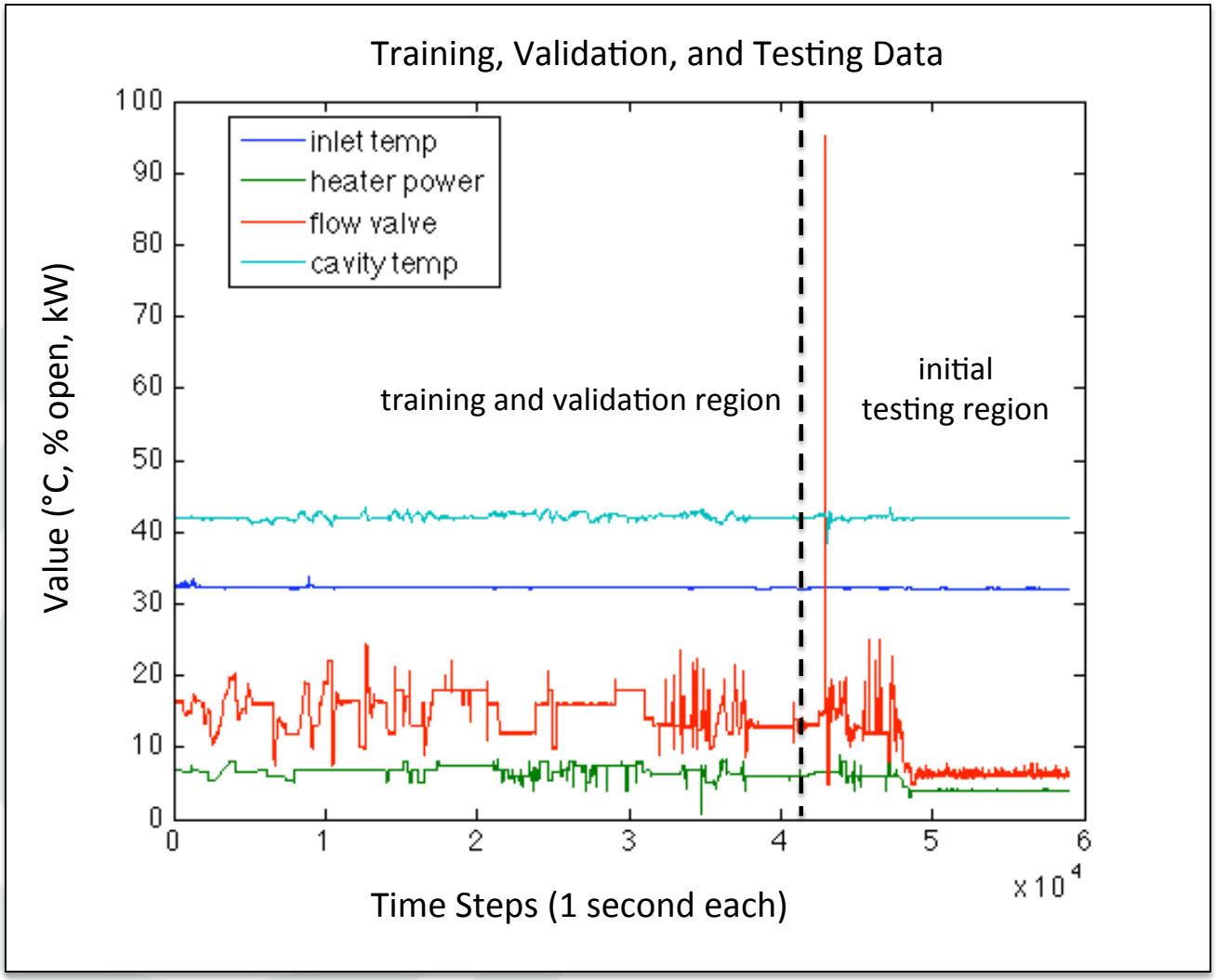
- **Other candidate models examined**
  - Adding return temperature as an input (with appropriate delay)
  - Excluding previous cavity temperatures as an input
  - Reducing the number of previous cavity temperatures used as input

Model Design: Input Parameter Selection

- **Chosen Parameters**
  - Flow Control Valve Setting
  - Heater Setting
  - LCW Temperature
  - → Cavity Temperature

- **Other candidate models examined**
  - Adding return temperature as an input (with appropriate delay)
  - Excluding previous cavity temperatures as an input
  - Reducing the number of previous cavity temperatures used as input
  - → Did well, but not as well as the originally proposed model

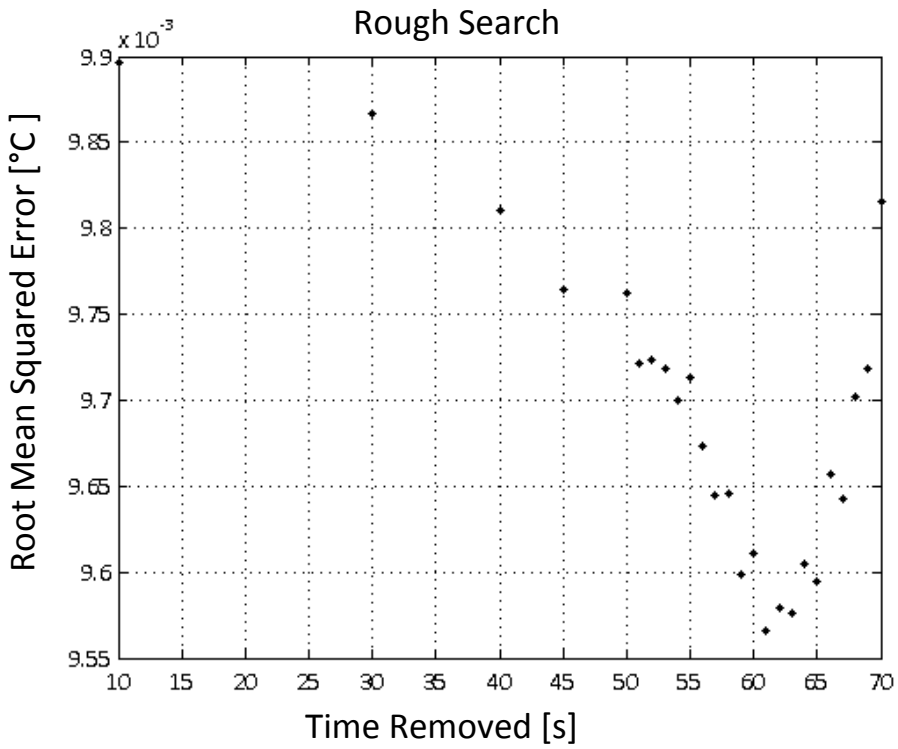
Initial Training, Validation, and Testing Data

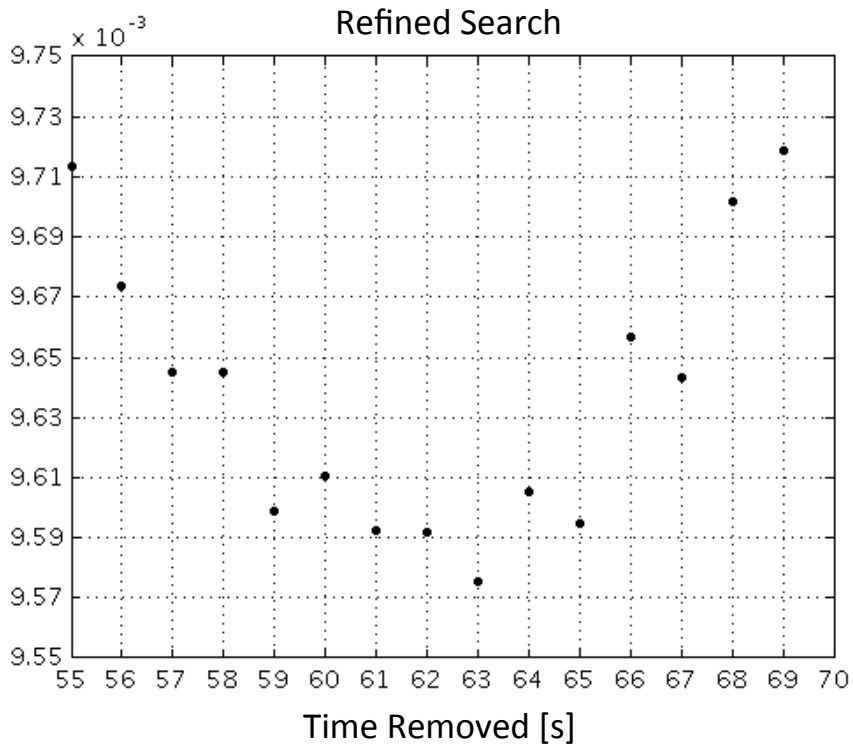


With more challenging test data to follow

Model Design: Variable Dead-Time Removal

- Measurements supply a starting point → but the delays vary
- Empirical study with NN gives an optimal solution
- 25 candidates
- 15-fold validation

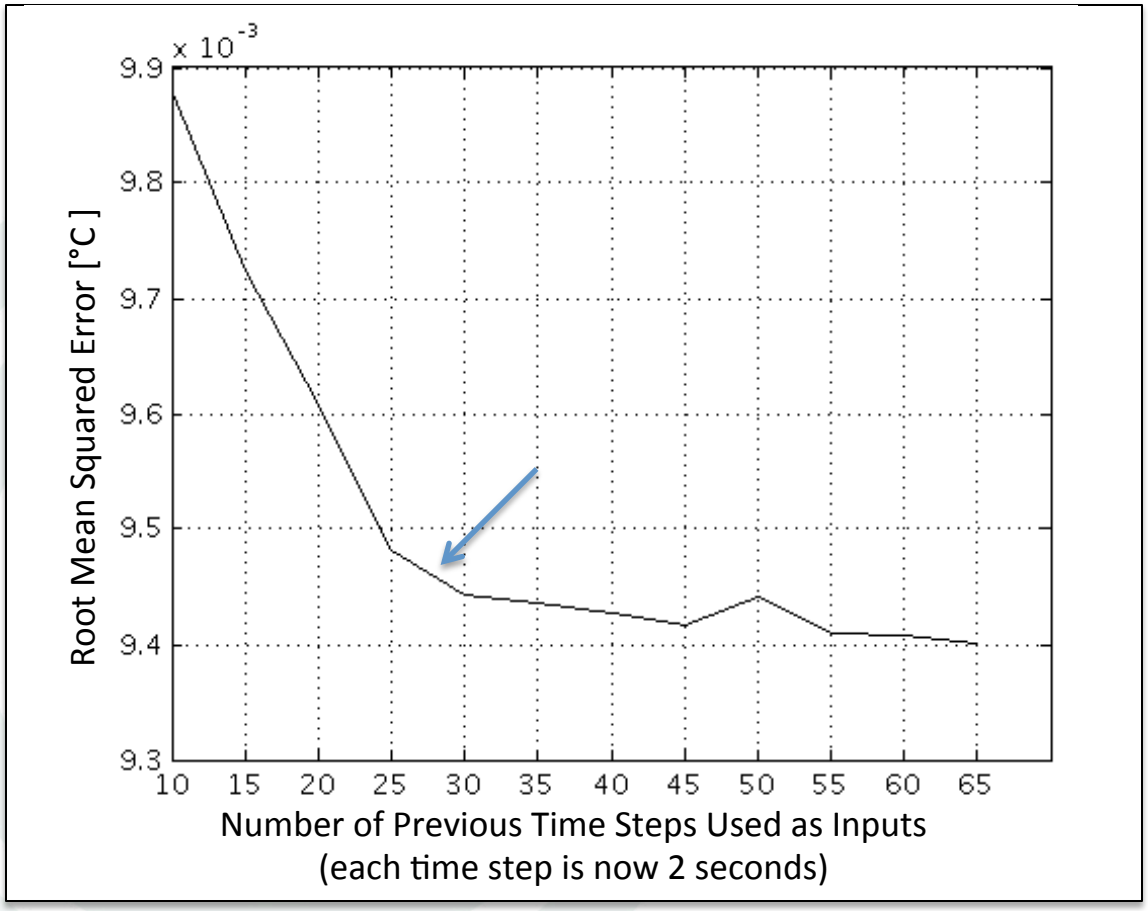




Electrical & Computer ENGINEERING

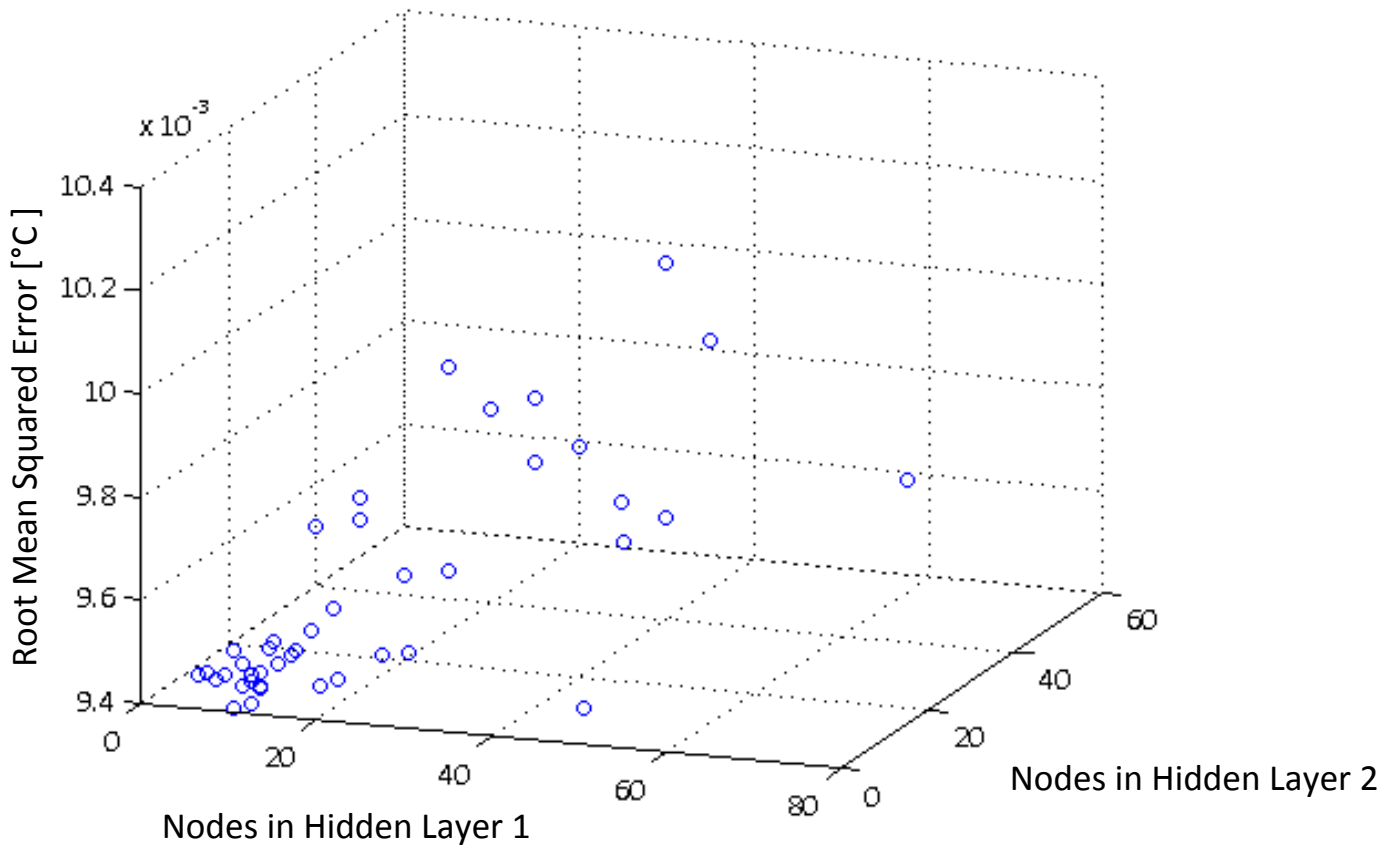
Model Design: Embedding Dimension

- 12 candidates
- 15-fold validation

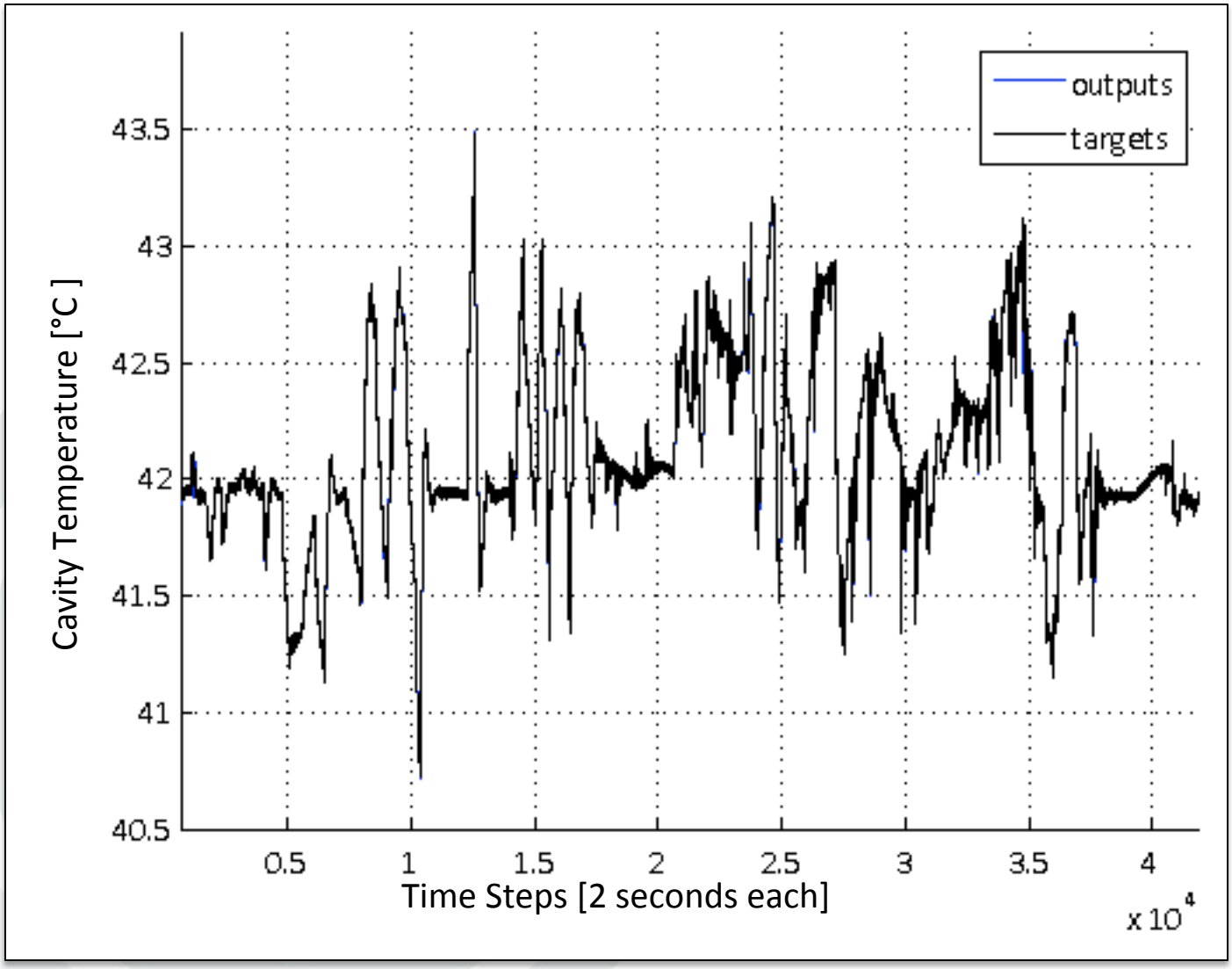


Number of Hidden Layers and Nodes

- 55 candidates for two-layer network
- 30 other candidates (not shown): recurrent, single-layer



Performance Over Training and Validation Set



0.0094 RMSE validation set

0.0092 RMSE training set

Testing Regions

Testing Regions

- **Region 1: pseudo-random impulses**

Testing Regions

- Region 1: pseudo-random impulses
- Region 2: steady-state with regulation

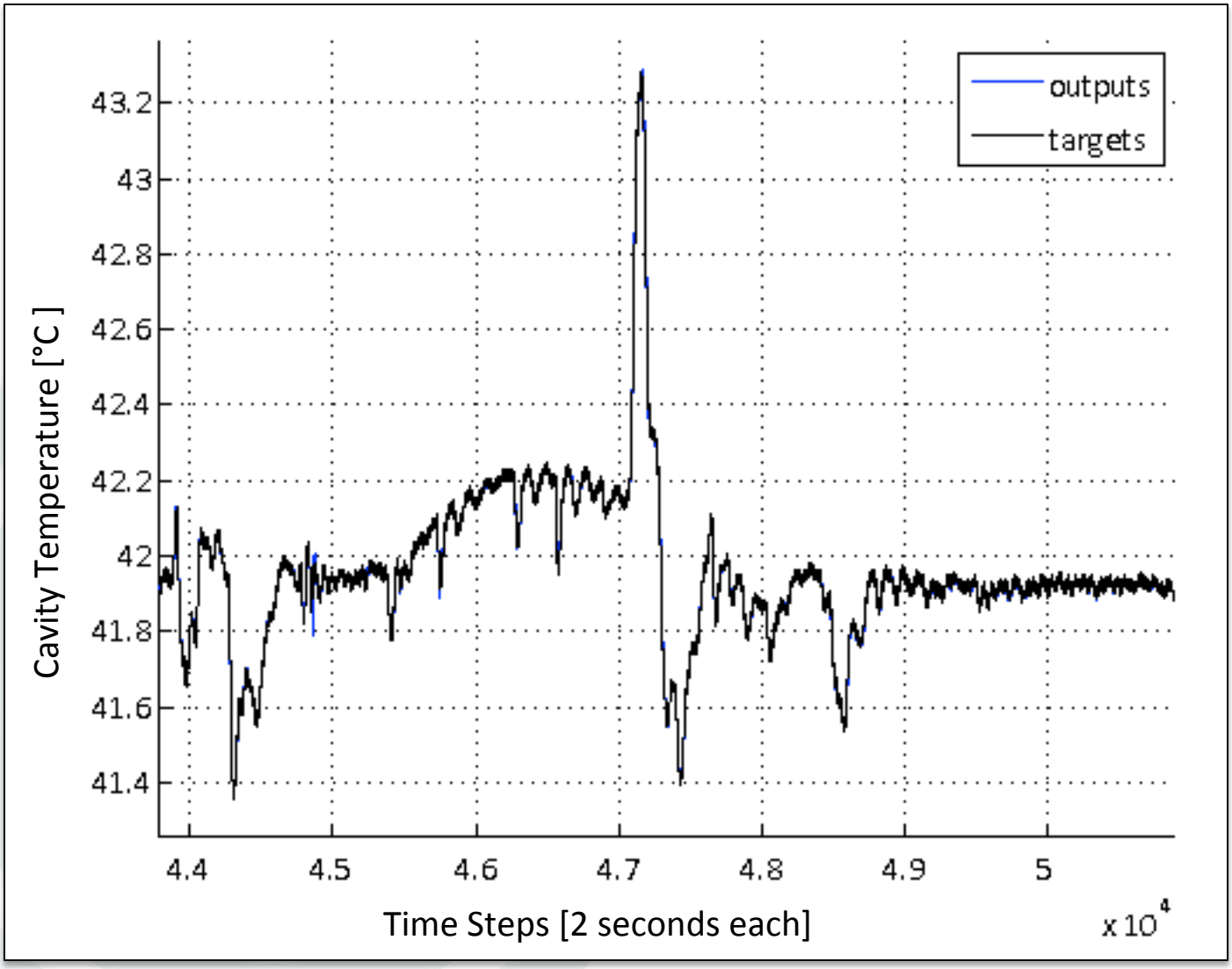
Testing Regions

- Region 1: pseudo-random impulses
- Region 2: steady-state with regulation
- Region 3: cavity conditioning, slow change in RF power

Testing Regions

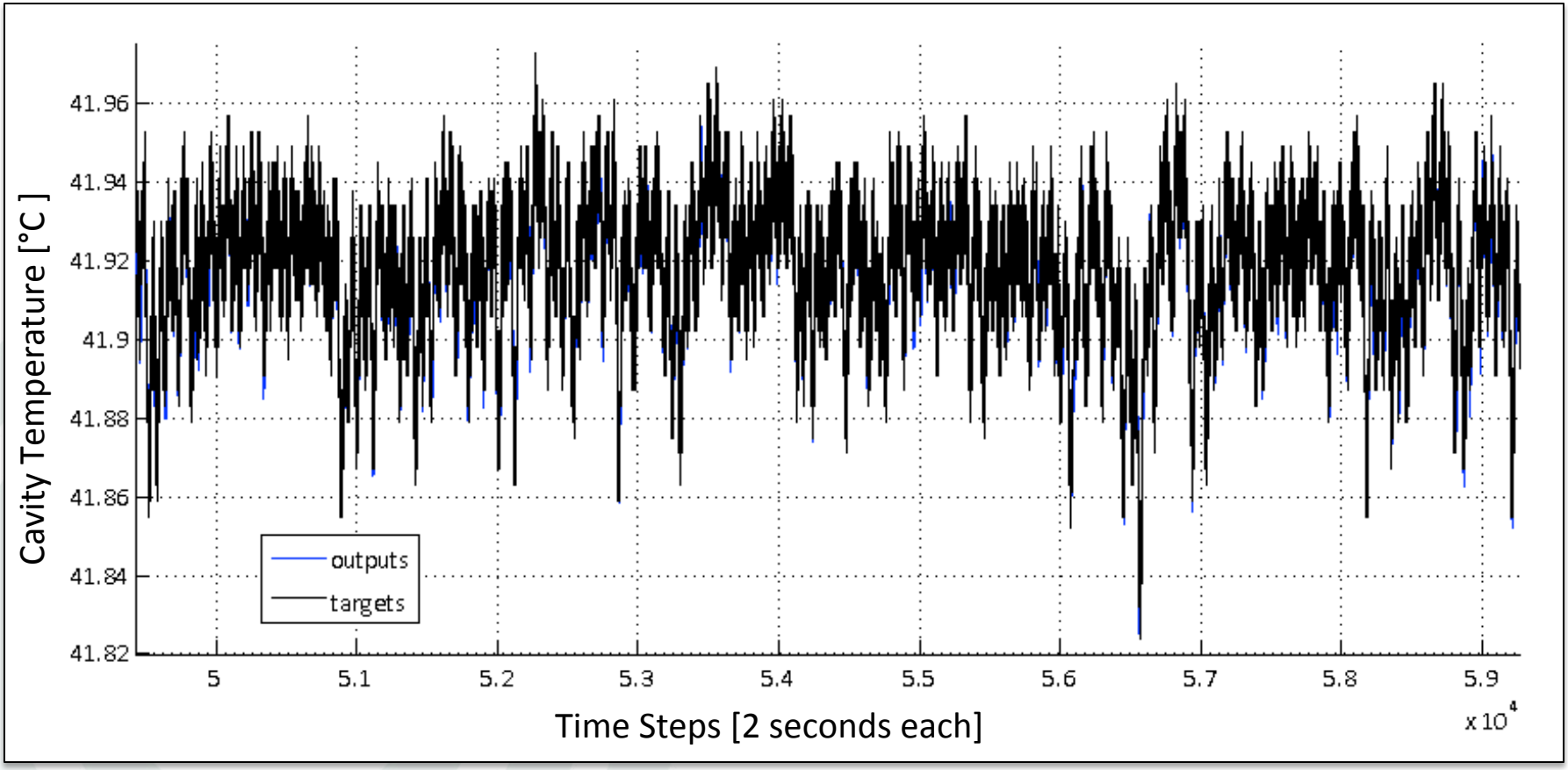
- Region 1: pseudo-random impulses
- Region 2: steady-state with regulation
- Region 3: cavity conditioning, slow change in RF power
- Region 4: cavity conditioning, faster change in RF power

Performance in Testing Region 1



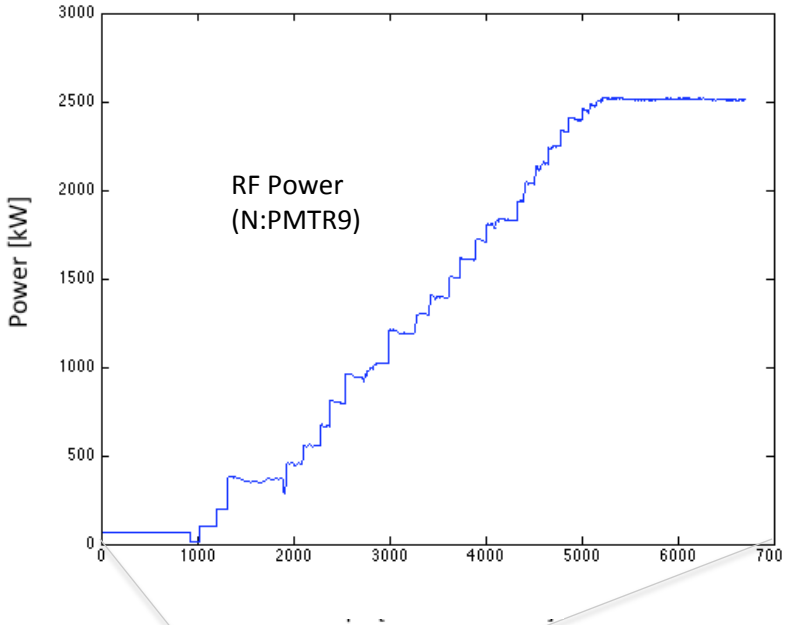
0.0100 RMSE

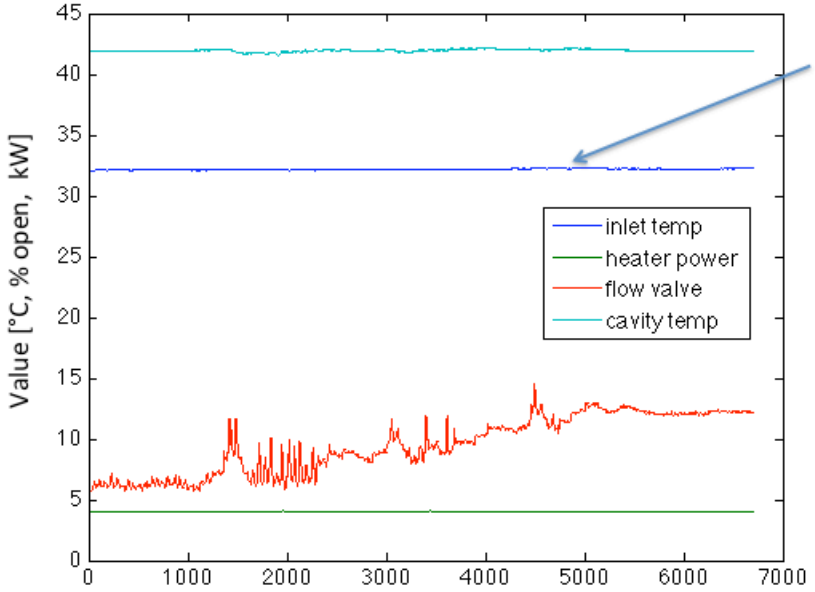
Performance in Testing Region 2



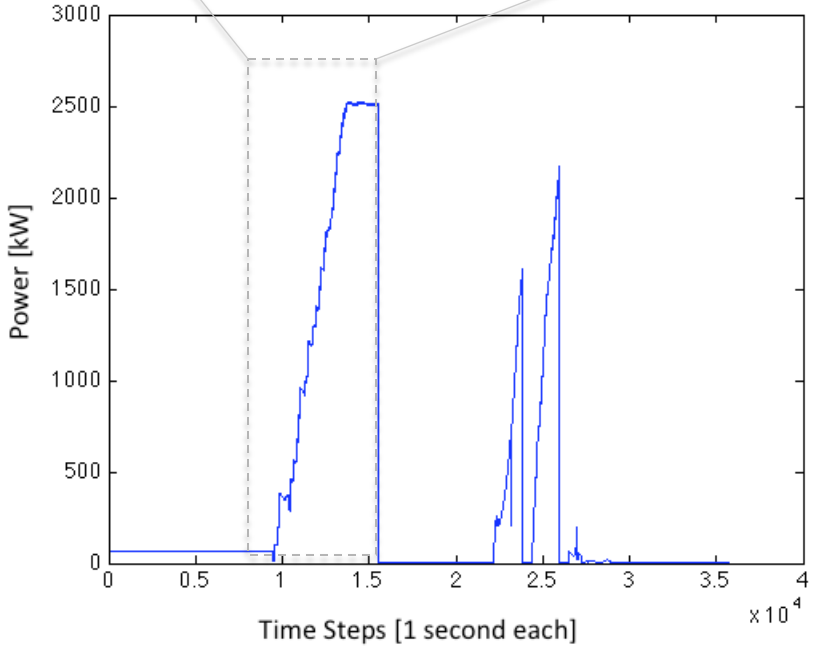
0.0096 RMSE

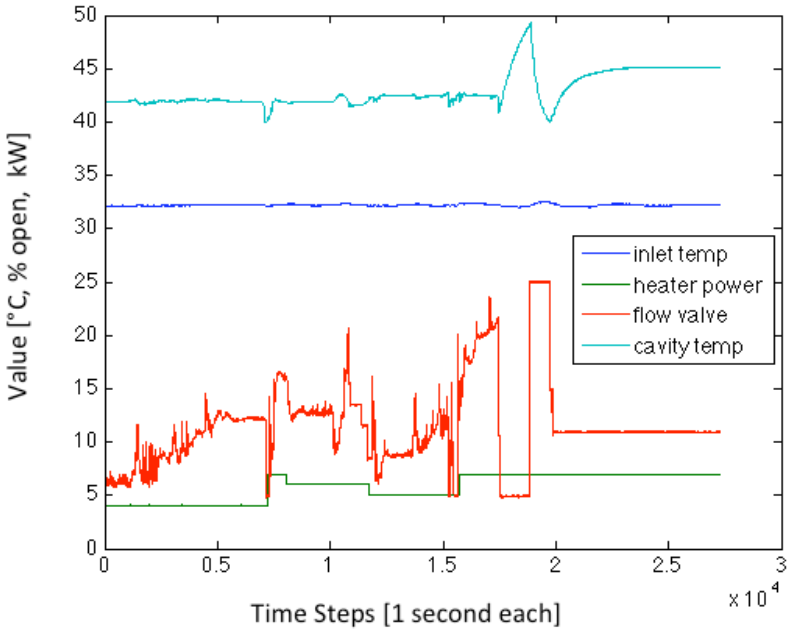
Testing Regions 3 and 4



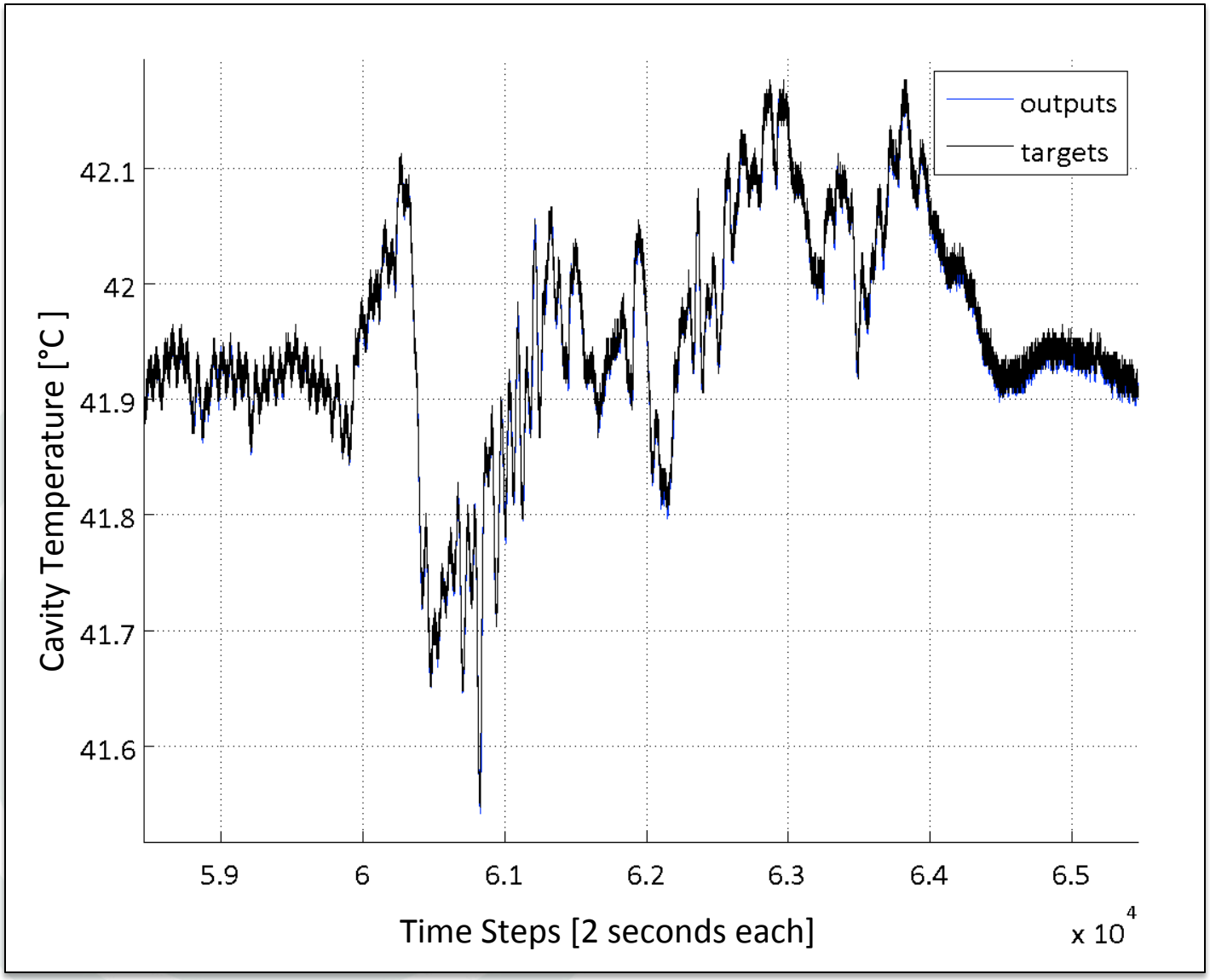


Note: these LCW temp disturbances go up to ~ 1°C, even though they look miniscule on this scale



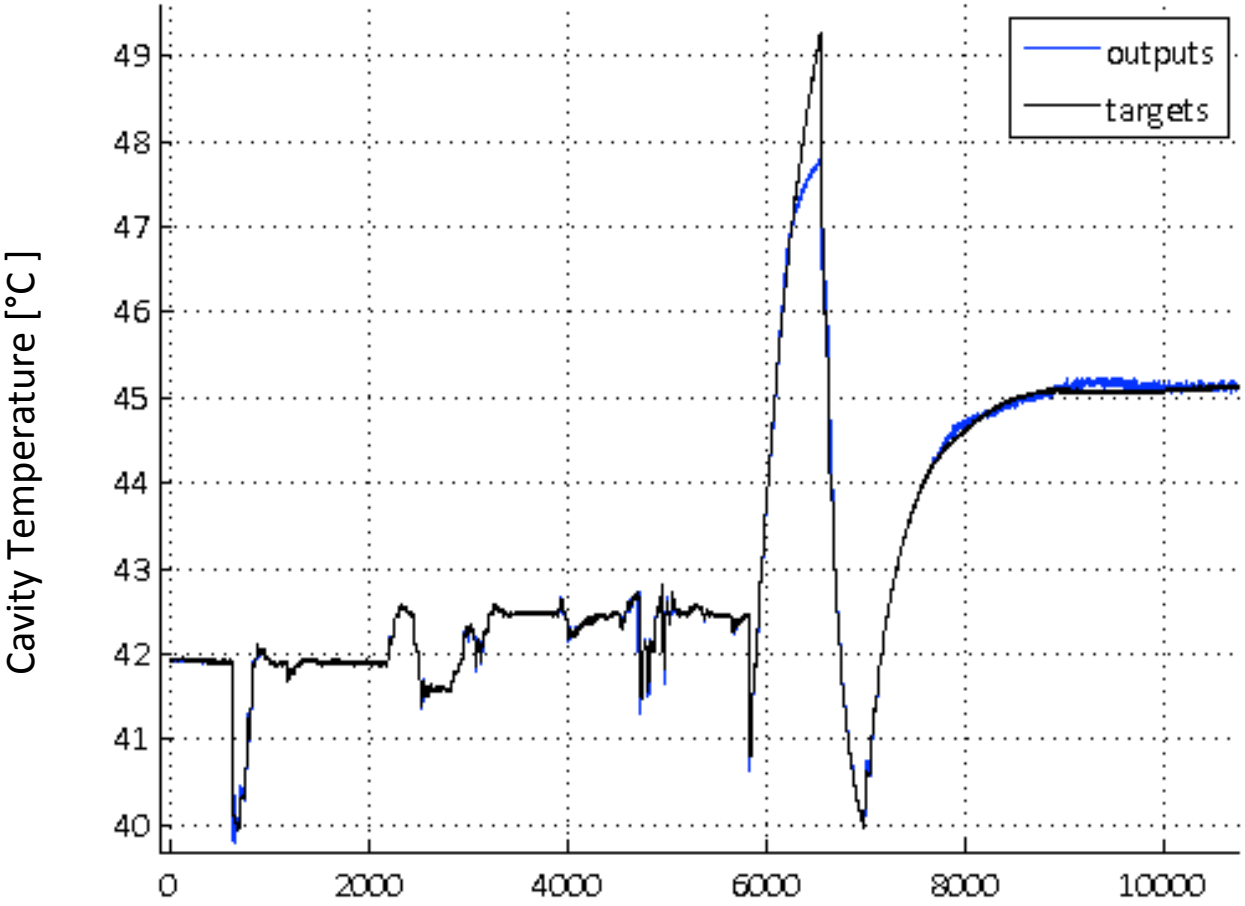


Performance on Testing Region 3



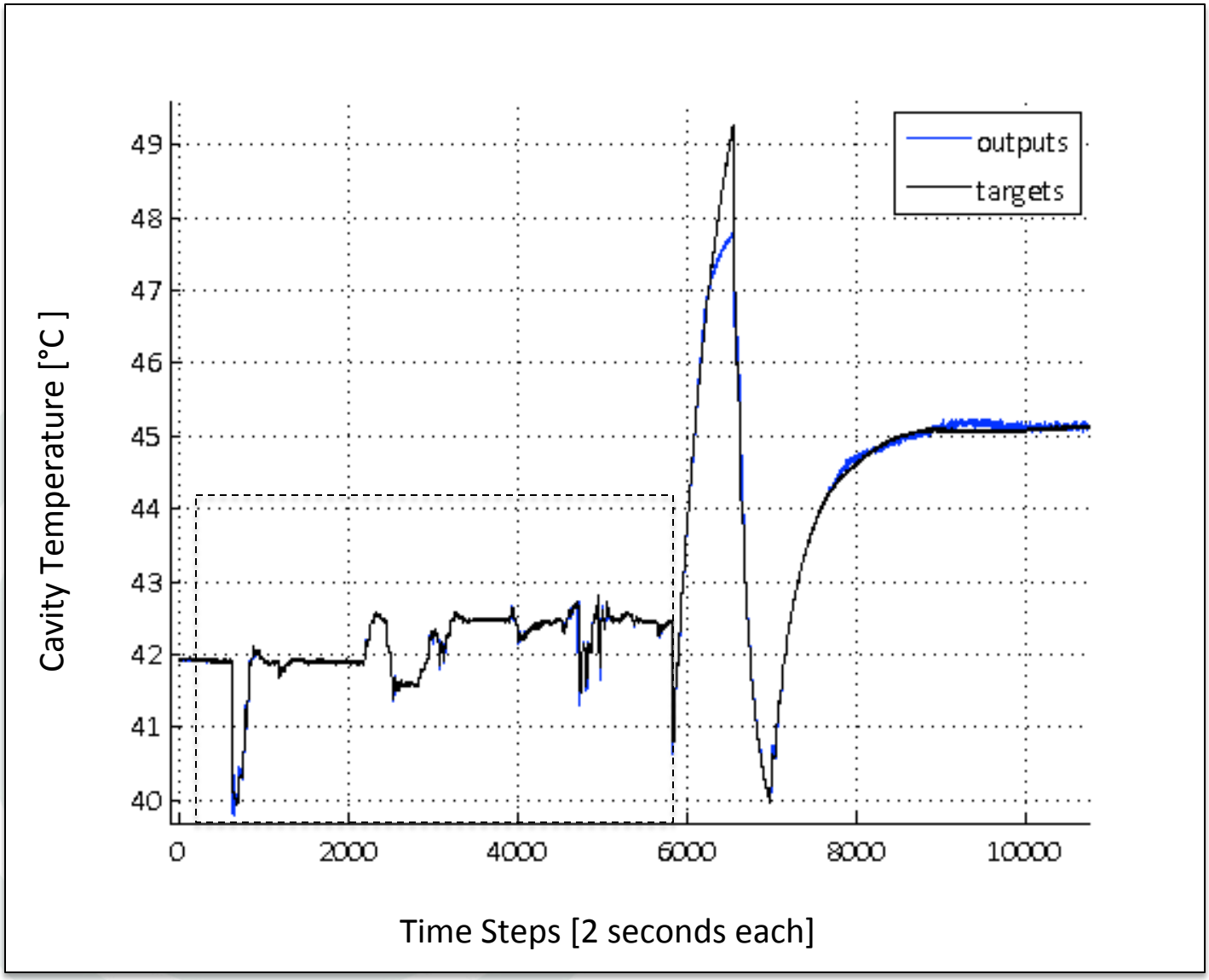
0.0156 RMSE

Performance on Testing Region 4



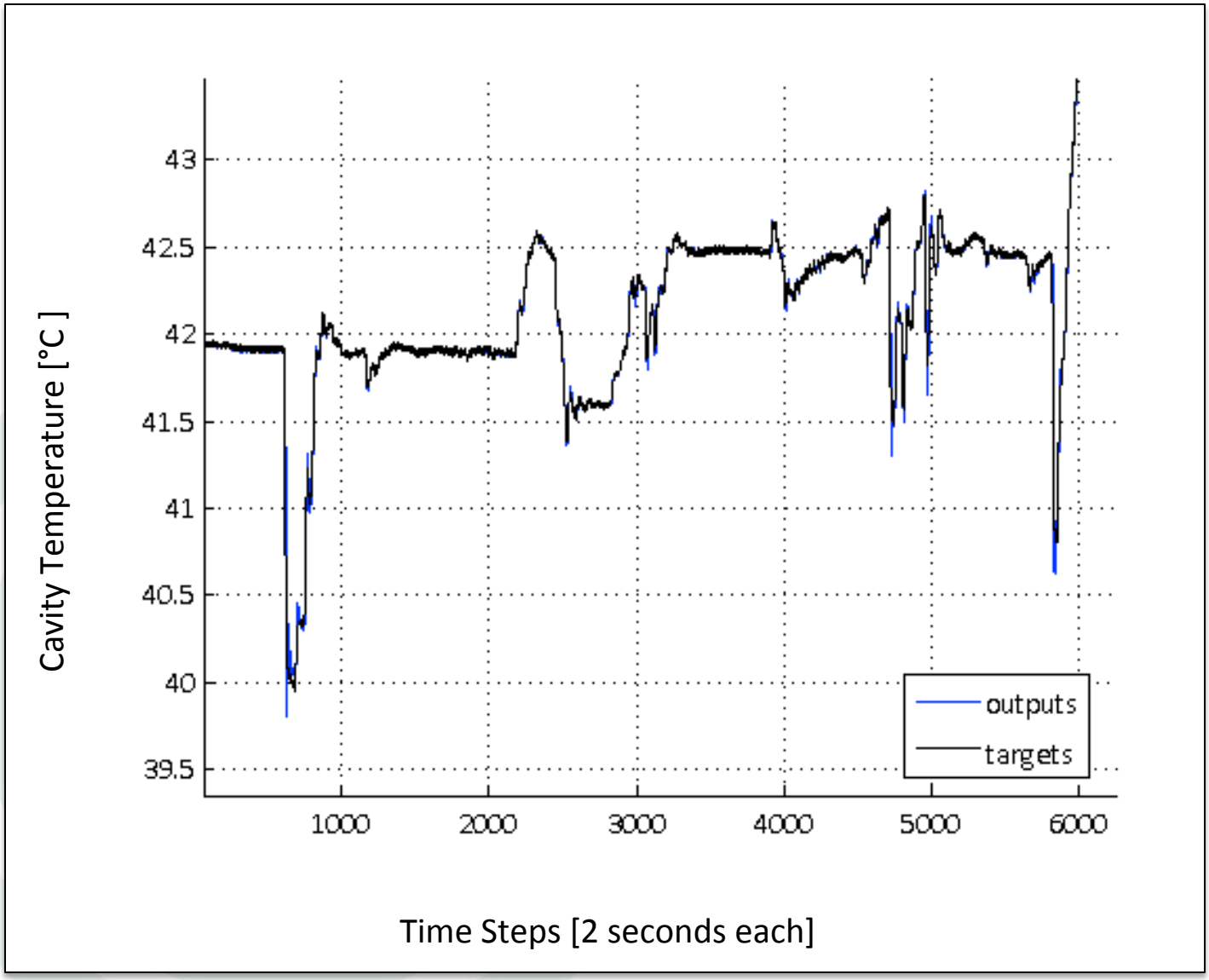
0.0549 RMSE

Performance on Testing Region 4

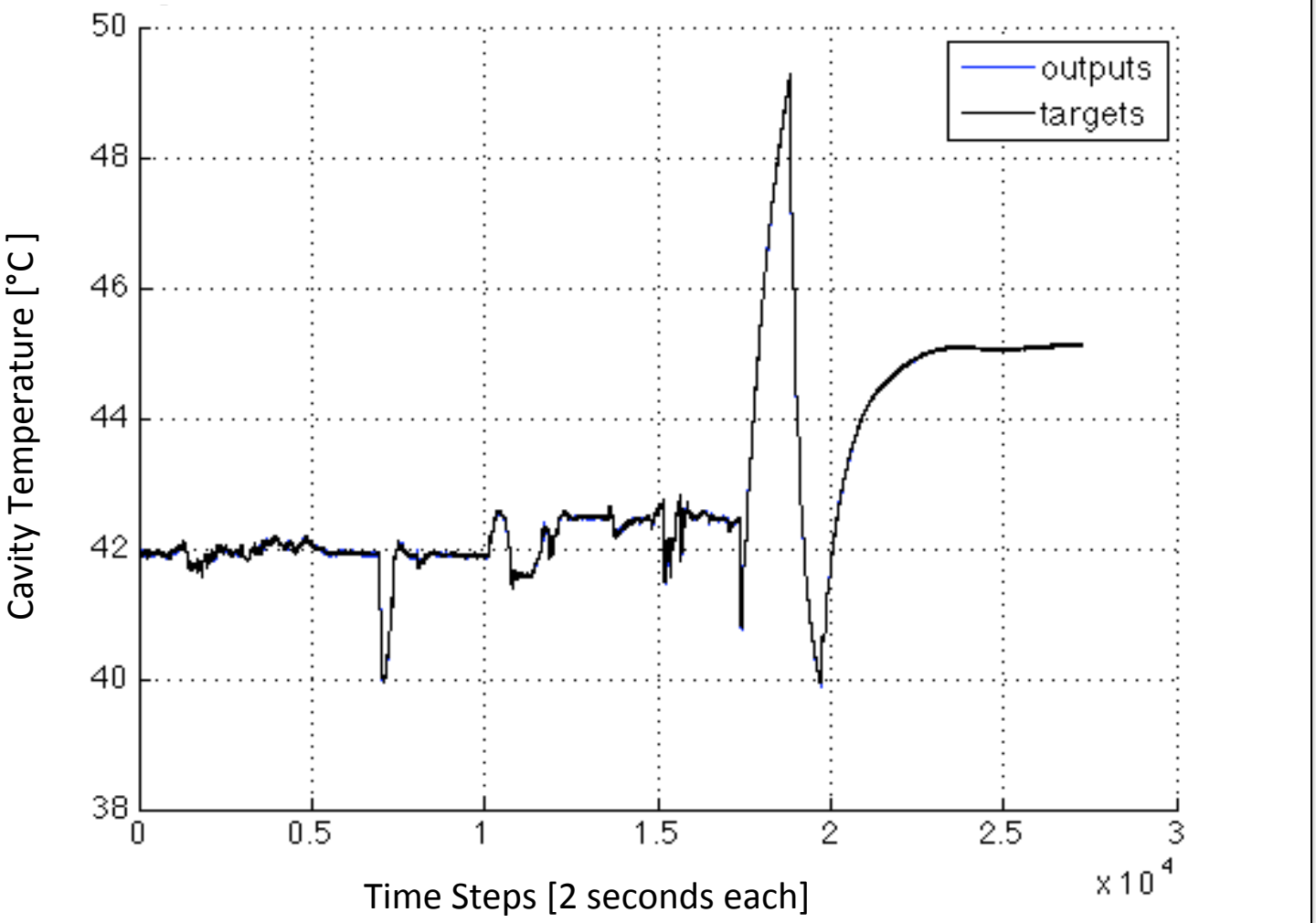


0.0549 RMSE

Performance on Testing Region 4



Performance on Testing Region 4 with Adaptation



0.0099 RMSE

What about large changes and different operating regimes?

What about large changes and different operating regimes?

- **Several options:**
  - Online adaptation
    - Slow and potentially unstable for large, rapid changes
    - Loss of previous learning

What about large changes and different operating regimes?

- **Several options:**
  - Online adaptation
    - Slow and potentially unstable for large, rapid changes
    - Loss of previous learning
  - Use RF power as an input and train across the operating regimes
    - Would need a lot of data for many transition types → time intensive + likely not feasible to capture adequately
    - Likely would need to a more cumbersome NN structure

What about large changes and different operating regimes?

- **Several options:**
  - Online adaptation
    - Slow and potentially unstable for large, rapid changes
    - Loss of previous learning
  - Use RF power as an input and train across the operating regimes
    - Would need a lot of data for many transition types → time intensive + likely not feasible to capture adequately
    - Likely would need to a more cumbersome NN structure
  - Train several NNs with data gathered in different operating regimes
    - Can switch models in control routines
    - Can adapt online for small adjustments
    - Many small models instead of one large, potentially problematic model

Current and Next Steps

- **Fully characterize model performance under challenging conditions**
  - If the model won't perform well, a model-based controller certainly won't
  - Need to test in real-time while doing I-O though ACNET
  - Need to test different updating schemes
  - Need to test regime-switching schemes

- **Improve implementation of control script in Matlab**
  - Real-time execution of NN, I-O with ACNET, optimization procedure, data preprocessing
  - Several computationally intensive processes

  → tested ACNET and NN I-O previously with a very limited direct inverse controller, but this did not involve any updating or optimization routines

- **Improve execution speed for Model Predictive Controller optimization**
  - Instantaneous linearization
  - Different algorithm

- **Simulations of controller performance**
- **Test controller output only**
- **Test controller operation in training regime**