Adaptive Deep Neural Networks for Multi-output Nonlinear and Nonstationary Regression

Professor Sheng Chen

School of Electronics and Computer Science

University of Southampton

Southampton SO17 1BJ, United Kingdom

Joint work with Dr Tong Liu, Department of Computer Science, University of Sheffield, U.K.

Background

- Artificial neural networks have evolved from '**shallow**' one-hidden-layer architecture, such as RBF, to '**deep**' architecture
  - **Deep learning** has achieved **breakthrough** progress in many walks of life
  - Deep neural networks have been applied to modeling of multi-output industrial processes

- Deep learning's success coincides with **digital big data** era
  - With massive historical data, training of deep neural network models becomes practical
  - Enabling the exploitation of deep learning capability to capture complex underlying nonlinear dynamic behaviours from data

- Many real-life processes are not only nonlinear but also highly **nonstationary**
  - During online operation, system's nonlinear dynamics can change significantly
  - Deep neural network model must **adapt fast** to such change

Motivations

- **Sampling period** of many industrial processes is **small**, and **adaptation** must be **sufficiently fast** to be completed within a sampling period
  - **Impossible to adapt structure** of deep neural network model, such as SAE, within sampling period
  - Instead, adaptation is taken place **only on weights of output regression layer**
  - **Insufficient** for tracking significant and fast changes in system

- We have proposed an adaptive **gradient radial basis function** network
  - Adapting structure of multi-output GRBF (MGRBF) is not only optimal but also imposes **litter** online computation complexity
  - Completely feasible to complete adaptation within a sample period
  - MGRBF is a **shallow** neural network

- **Combining** deep learning capability of **deep neural network**, such as SAE, with excellent adaptability of **MGRBF**? $\Rightarrow$ Motivate this research

System Model

- Multi-output **nonlinear** and **nonstationary** system

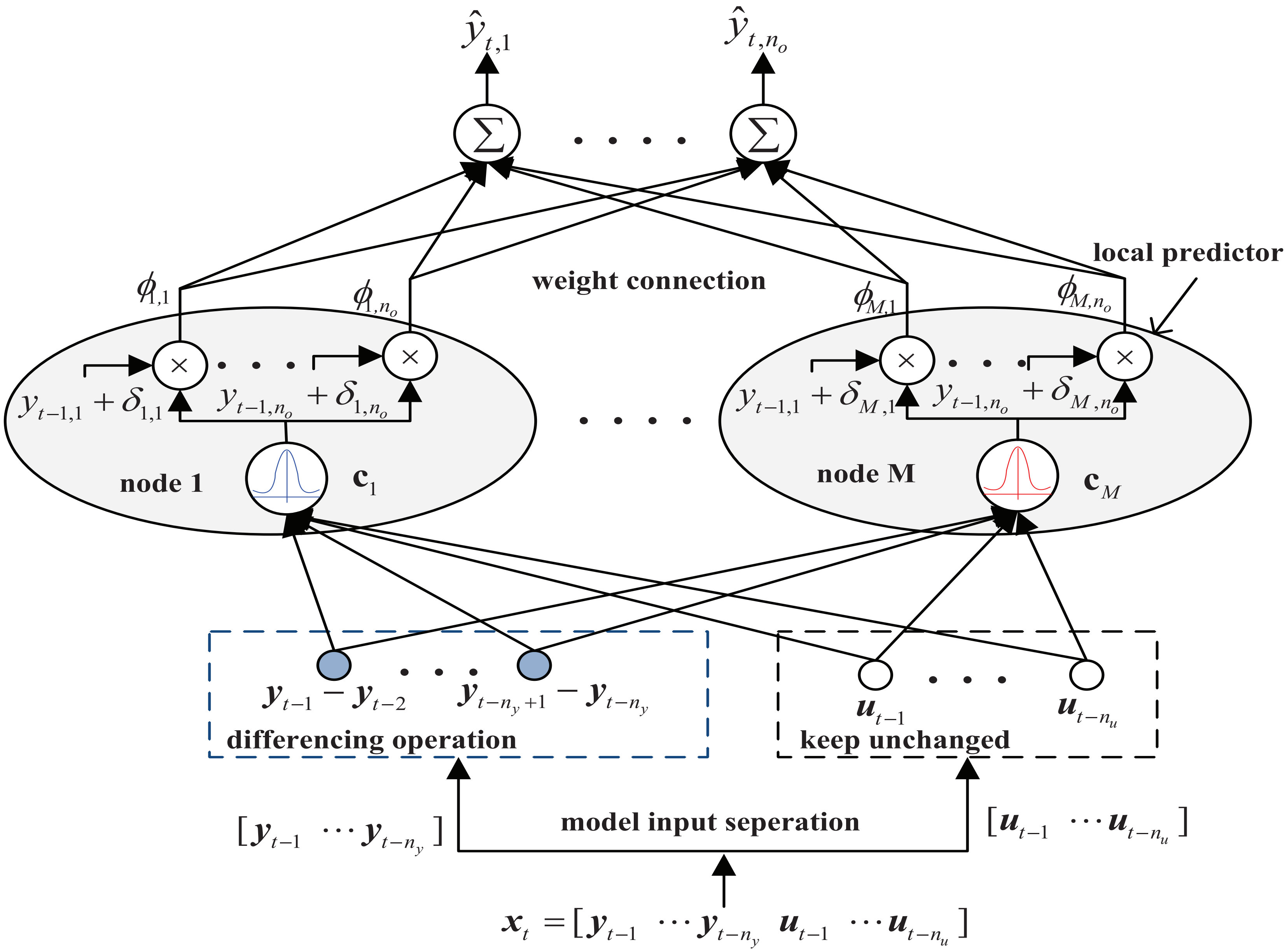
$$\boldsymbol{y}_t = \boldsymbol{f}_{\text{sys}}(\boldsymbol{x}_t; t) + \boldsymbol{\xi}_t$$

  - **Output** $\boldsymbol{y}_t \in \mathbb{R}^{n_o}$ with lag n_y, **Input** $\boldsymbol{u}_t \in \mathbb{R}^{n_i}$ with lag n_u, **Noise** $\boldsymbol{\xi}_t$
  - Unknown nonlinear and nonstationary system map $\boldsymbol{f}_{\text{sys}}(\cdot; t)$
  - System 'input' **embedding** vector $\boldsymbol{x}_t \in \mathbb{R}^{n_o n_y + n_i n_u}$

$$\boldsymbol{x}_t = \left[\boldsymbol{y}_{t-1}^{\text{T}} \cdots \boldsymbol{y}_{t-n_y}^{\text{T}} \ \boldsymbol{u}_{t-1}^{\text{T}} \cdots \boldsymbol{u}_{t-n_u}^{\text{T}}\right]^{\text{T}}$$

- This is **one-step** ahead predictor model.
  - Extension to **multi-step** ahead predictor straightforward
- The task is to construct predictor: $\widehat{\boldsymbol{y}}_t = \widehat{\boldsymbol{f}}_{\text{model}}\big(\boldsymbol{x}_t; \boldsymbol{\Theta}_t\big)$
  - with model structure $\widehat{\boldsymbol{f}}_{\text{model}}$ and parameter matrix $\boldsymbol{\Theta}_t$ available at t

Multi-output GRBF Network



MGRBF – How It Works

- **Differencing output** variable to reduce nonstationarity: **MGRBF** input

$$\boldsymbol{x}'_t = \left[\boldsymbol{y}^{\mathrm{T}}_{t-1} - \boldsymbol{y}^{\mathrm{T}}_{t-2} \cdots \boldsymbol{y}^{\mathrm{T}}_{t-n_y} - \boldsymbol{y}^{\mathrm{T}}_{t-n_y-1}\ \boldsymbol{u}^{\mathrm{T}}_{t-1} \cdots \right]^{\mathrm{T}} \in \mathbb{R}^{n_o(n_y-1)+n_i n_u}$$

- **Hidden node as local predictor** of $\boldsymbol{y}_t$: **MGRBF** j-th node

$$\varphi_{j,i}(\boldsymbol{x}'_t) = \left(y_{t-1,i} + \delta_{j,i}\right) \cdot e^{-\frac{\|\boldsymbol{x}'_t - \boldsymbol{c}_j\|^2}{2\sigma^2}},\ 1 \le j \le M, 1 \le i \le n_o$$

- In training, if $\boldsymbol{x}'_{t_j}$ is selected as j-th center $\boldsymbol{c}_j$, local predictor **scalar** is set to $\delta_{j,i} = y_{t_j,i} - y_{t_j-1,i}$
  - In training, $\varphi_{j,i}(\boldsymbol{x}'_t)$ is **perfect** predictor of $y_{t,i}$
  - In prediction, if $\boldsymbol{x}'_t$ is close to jth center, $\varphi_{j,i}(\boldsymbol{x}'_t)$ is **very good** predictor of $y_{t,i}$

- **Hidden nodes encode** system **states** observed

MGRBF – Training/Adaptation

- Given training data $\left\{\boldsymbol{x}_t, \boldsymbol{d}_t = \boldsymbol{y}_t - \boldsymbol{y}_{t-1}; \boldsymbol{y}_t\right\}_{t=1}^{N}$, **efficient two-stage training**
  - **OLS** selects **subset** model $\left\{\boldsymbol{c}_{t_j}, \boldsymbol{\delta}_{t_j}\right\}_{j=1}^{M}$, hidden nodes' centers and scalars
  - Regularized LS estimates connection weight matrix

- During online operation, when current modeling $\widehat{\boldsymbol{y}}_t$ is insufficient:

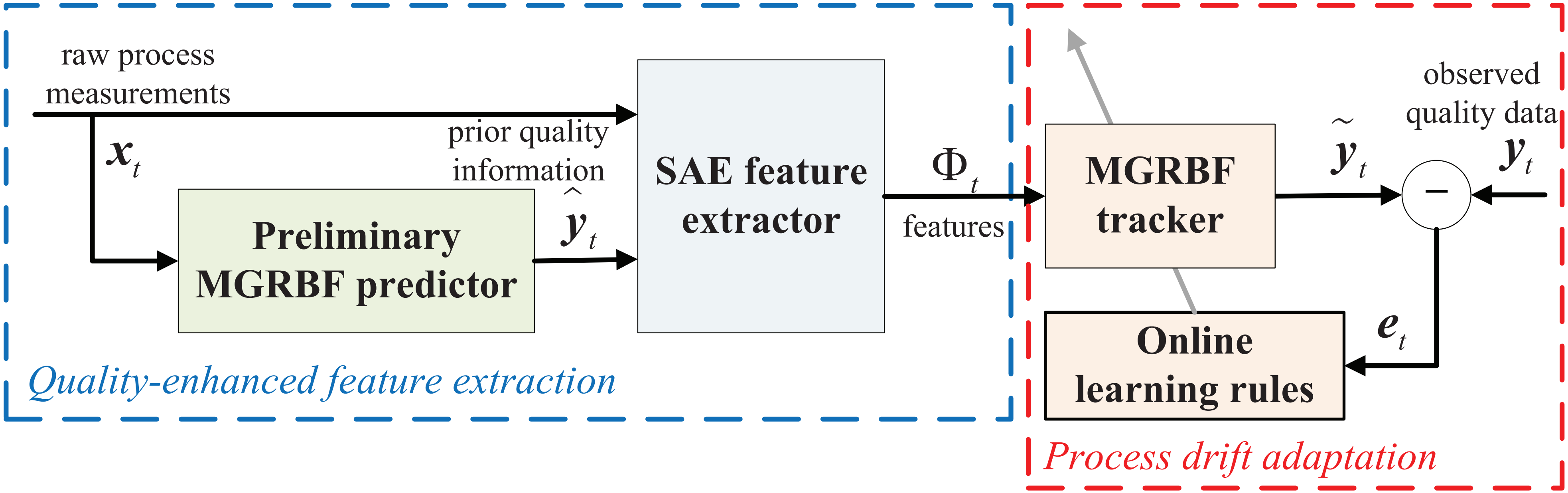
$$\left\|\boldsymbol{y}_t - \widehat{\boldsymbol{y}}_t\right|^2 / \|\boldsymbol{y}_t^2 \geq \text{threshold}$$

  - Worst (contributing smallest to output) node **replaced** with a new node:

$$\text{node center } \boldsymbol{c}_r \leftarrow \boldsymbol{x}_t' \quad \text{node scalar} \boldsymbol{\delta}_r \leftarrow \boldsymbol{y}_t - \boldsymbol{y}_{t-1}$$

- Adaptive MGRBF achieves **balanced** trade-off of stability and plasticity
  - ability to retain acquired knowledge (**stability**) and ability to forget out-of-the-date knowledge so as to learn new one as quickly as possible (**plasticity**)

Proposed Deep Neural Network: Structure



- **MGRBF preliminary predictor** module, provide preliminary output prediction
- **Output-enhanced stacked autoencoder** module, provide deep output-relevant features
- **MGRBF adaptive predictor** module, provide final output prediction

Proposed Deep Neural Network: Rationale

- **SAE** is a **deep neural network** finding its way to **regression** application
  - Layers of stacked autoencoders extract deep features from input
  - Given information of output $\boldsymbol{y}_t$, SAE can extract much better-quality features

- Impossible to provide y_t as input to SAE - We do next best thing, provide a perdition of $\boldsymbol{y}_t$ as input to SAE by **MGRBF preliminary predictor**

- Instead of usual linear output regression layer on top of SAE to provide prediction of $\boldsymbol{y}_t$, we replace it by a much stronger **MGRBF adaptive predictor**

- **Training** of proposed deep neural network
  - **OLS** based two-stage for MGRBF preliminary predictor
  - **Standard optimization** procedure for SAE
  - **OLS** based two-stage for MGRBF adaptive predictor

Proposed Deep Neural Network: Operation

- Proposed DNN: SAE enhanced by MGRBF preliminary predictor maps process input space onto deep **feature space**, and MGRBF adaptive predictor then maps feature space onto process **output space**

- During online operation, MGRBF preliminary predictor and SAE are **fixed** (impossible to adapt whole SAE structure online anyway)

- MGRBF adaptive predictor is **adapted** online to track process's changing dynamics
  - When underlying system dynamics change significant, feature space changes accordingly
  - MGRBF adaptive predictor capable of fast adapting to changing process dynamics
  - while imposing very low online computational complexity, capable of meeting **real-time** constraint of small sampling period

- Proposed deep neural network integrates **deep learning capability** of **SAE** with **excellent adaptability** of **MGRBF**

Experiment Setup

- **Proposed** DNN is compared with following **benchmarks**
  - Partial least square (PLS): **fixed** during online operation
  - Multi-output long short-term memory (LSTM): **fixed** during online operation
  - Adaptive multi-output SAE (SAE_{RLS}): during online operation, only **weights** of output regression layer are **adapted** by RLS
  - Fast tunable multi-output RBF (TRBF): during online operation, RBF **hidden layer** is **adaptive**
  - Multi-output selective ensemble regression with growing and pruning (GAP-SER): during online operation, **grow and prune local model set**
  - Adaptive multi-output GRBF (AGRBF): during online operation, GRBF **hidden layer** is **adaptive**

- Performance measures: **determinant of test error covariance** $\log(\det(\text{Cov}(\boldsymbol{E})))$ and **coefficient of determination** (R^2)

- Online computational complexity: measured by **averaged computation time per sample** (ACTpS) in [ms]

Penicillin Fermentation Process

- Penicillin concentration, biomass concentration and substrate concentration are three process outputs, while 10 other process variables are process inputs

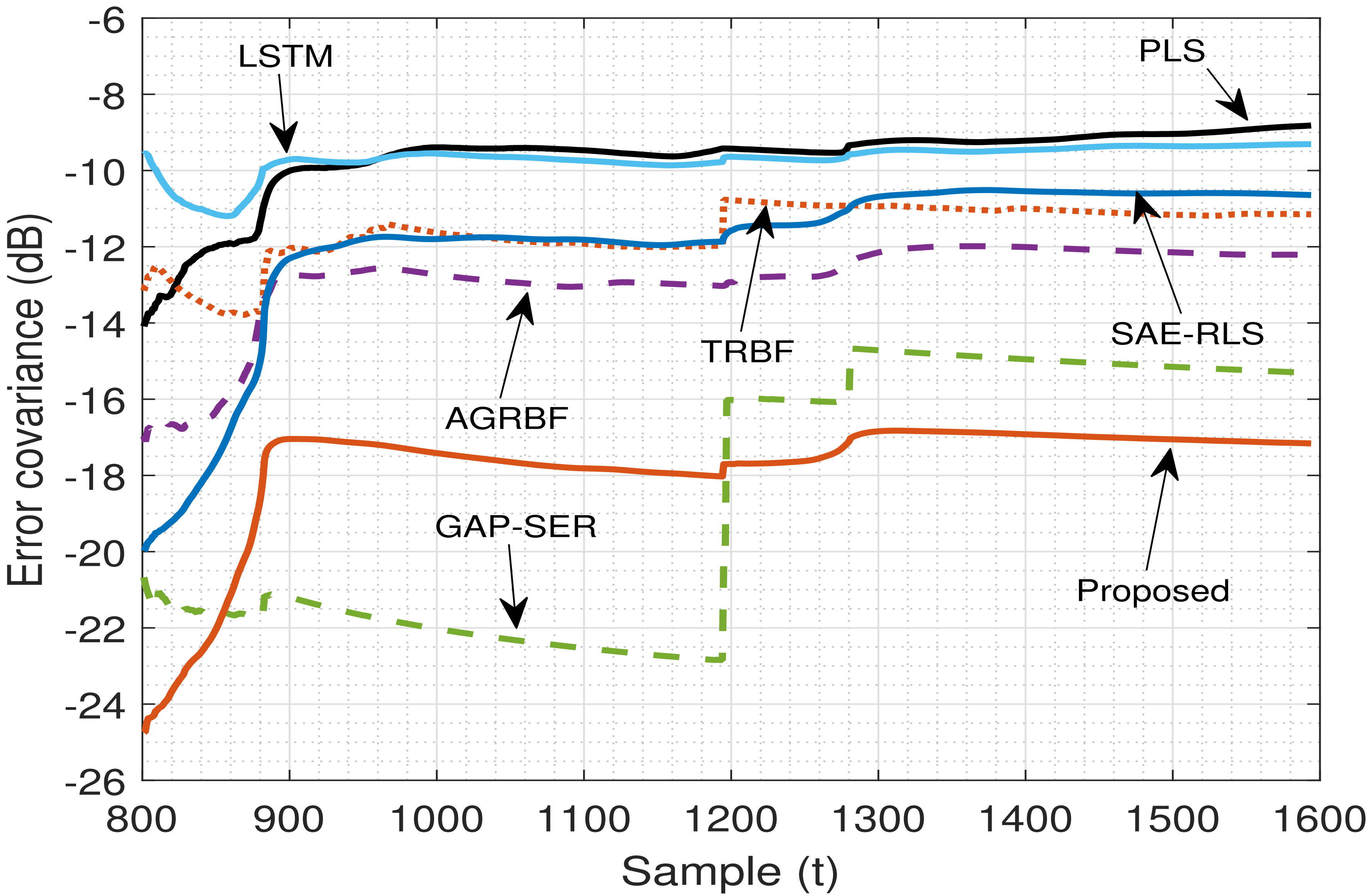
Method	$\log(\det(\text{Cov}(\boldsymbol{E})))$ (dB)	averaged R^2	ACTpS (ms)
PLS	-8.8180	0.9292	NA
TRBF	-11.1485	0.9943	0.0780
AGRBF	**-12.2161**	**0.9983**	**0.0296**
GAP-SER	**-15.3111**	**0.9936**	4.3732
LSTM	-9.3079±0.2651	0.9696±0.0169	NA
SAE_{RLS}	-10.6432±1.4741	0.9359±0.1174	**0.0036**
Proposed	**-17.1598±0.8739**	**0.9998±0.0002**	**0.0221**

- SAE_{RLS}, LSTM, and proposed DNN depend on **initialization**, average and standard deviation over 10 independent runs are given

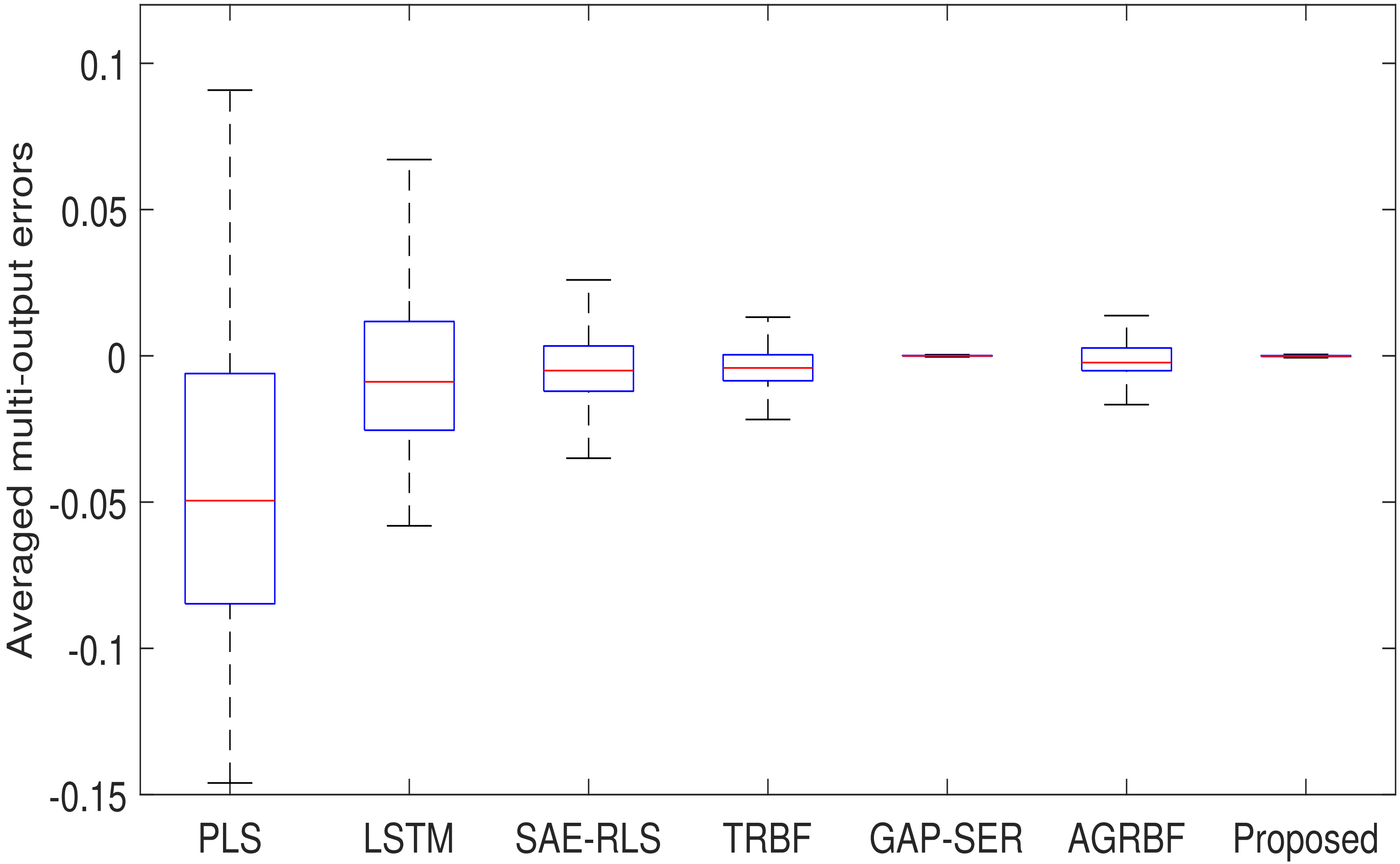
- SAE_{RLS} has smallest ACTpS, as it **only** adapts output weights

- Proposed DNN has **best test** performance with ACTpS smaller than AGRBF
  - Dimension of deep feature space is much smaller than that of input space

Test $\log(\det(\mathrm{Cov}(\boldsymbol{E})))$ learning curves



Box Plots

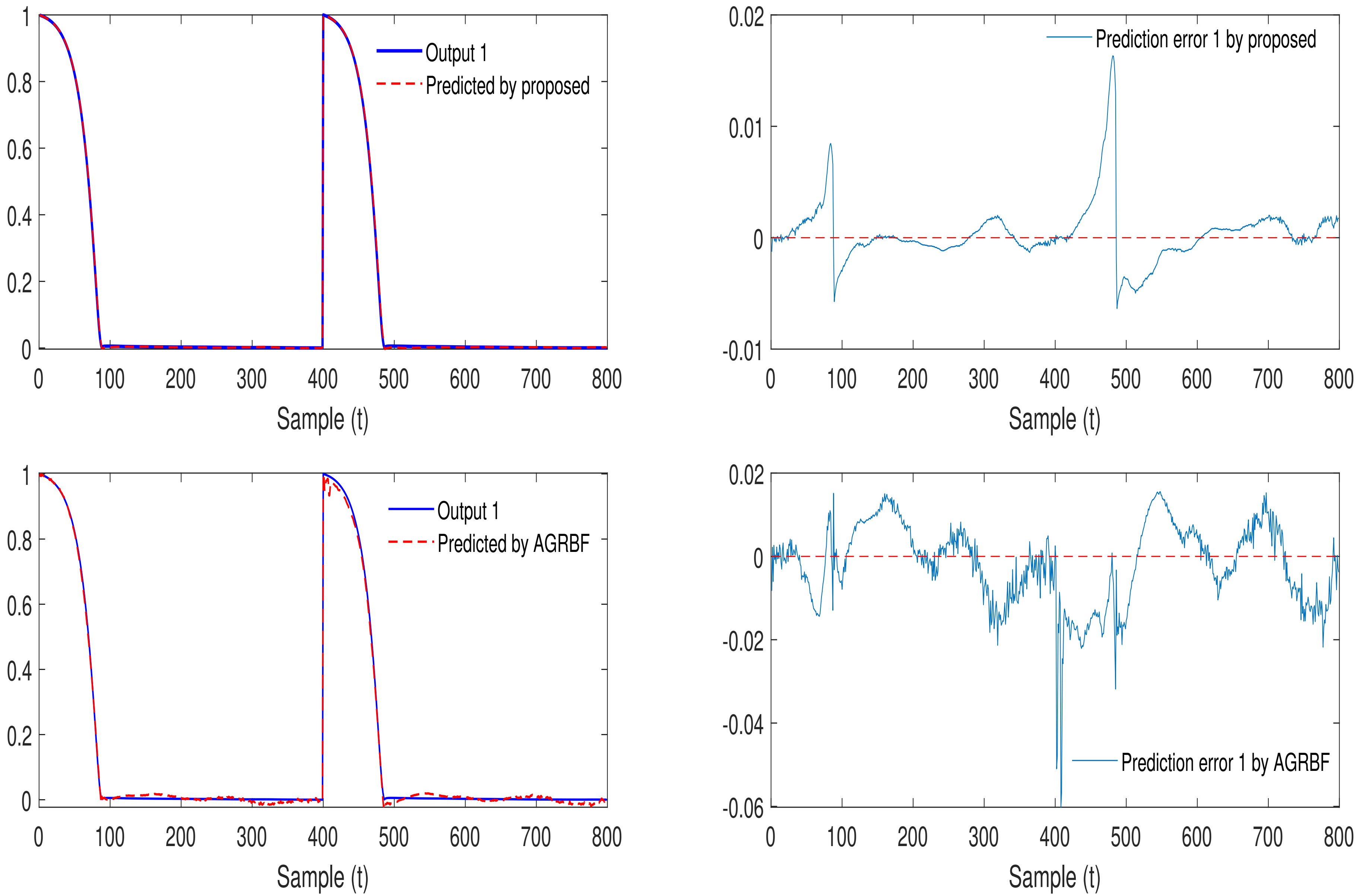

Averaged multi-output errors
0.1
0.05
0
-0.05
-0.1
-0.15
PLS
LSTM
SAE-RLS
TRBF
GAP-SER
AGRBF
Proposed

Test MSE for Individual Outputs

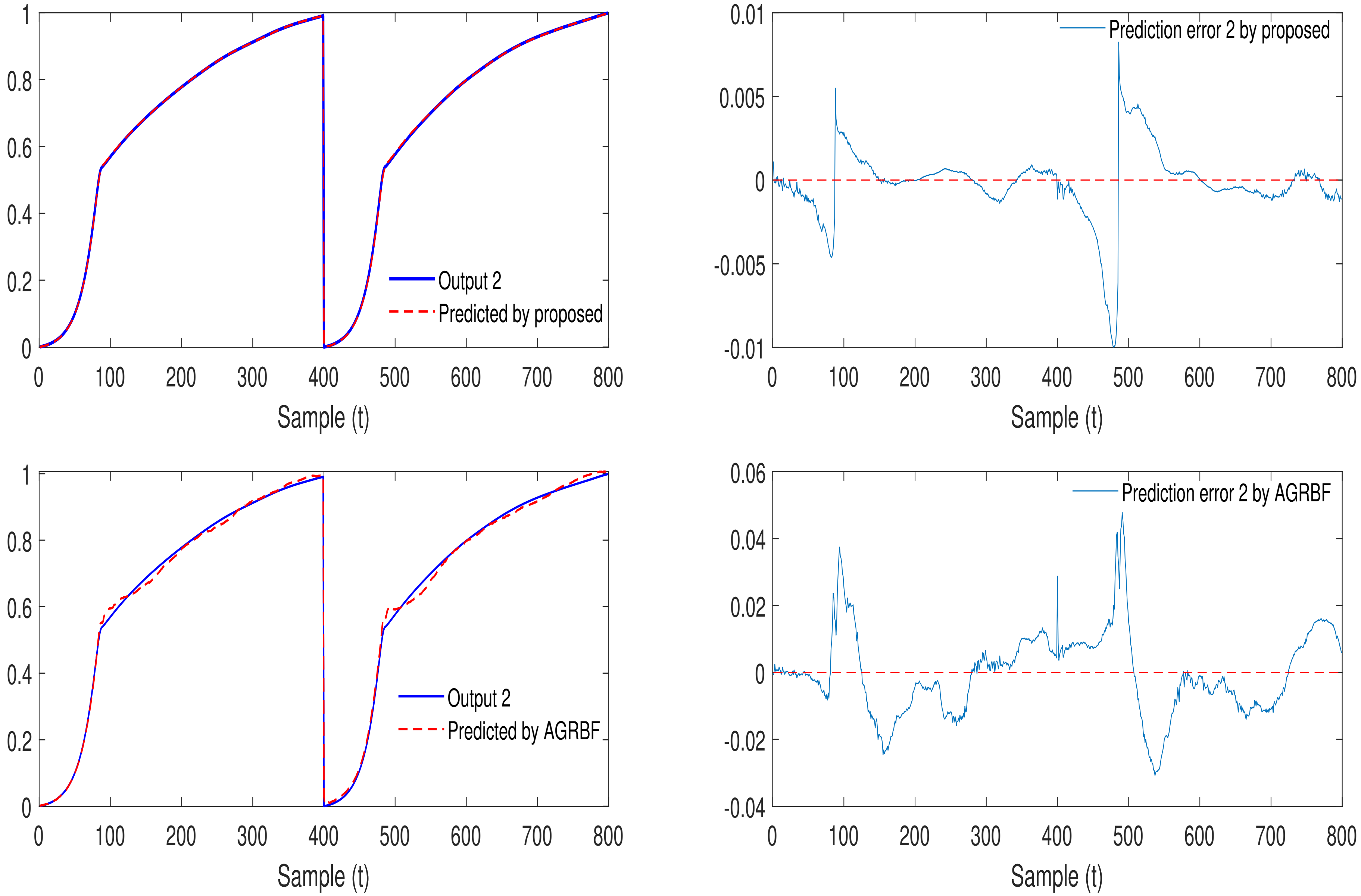
- Three best methods in terms of test MSE for individual outputs

Method	MSE (dB)		
	y_1	y_2	y_3
AGRBF	**-39.8615**	**-37.9934**	**-35.2746**
GAP-SER	-28.7491	**-81.2151**	-30.7240
Proposed	**-46.9541$\pm$3.5820**	**-49.9950$\pm$4.6632**	**-53.0888$\pm$7.7268**

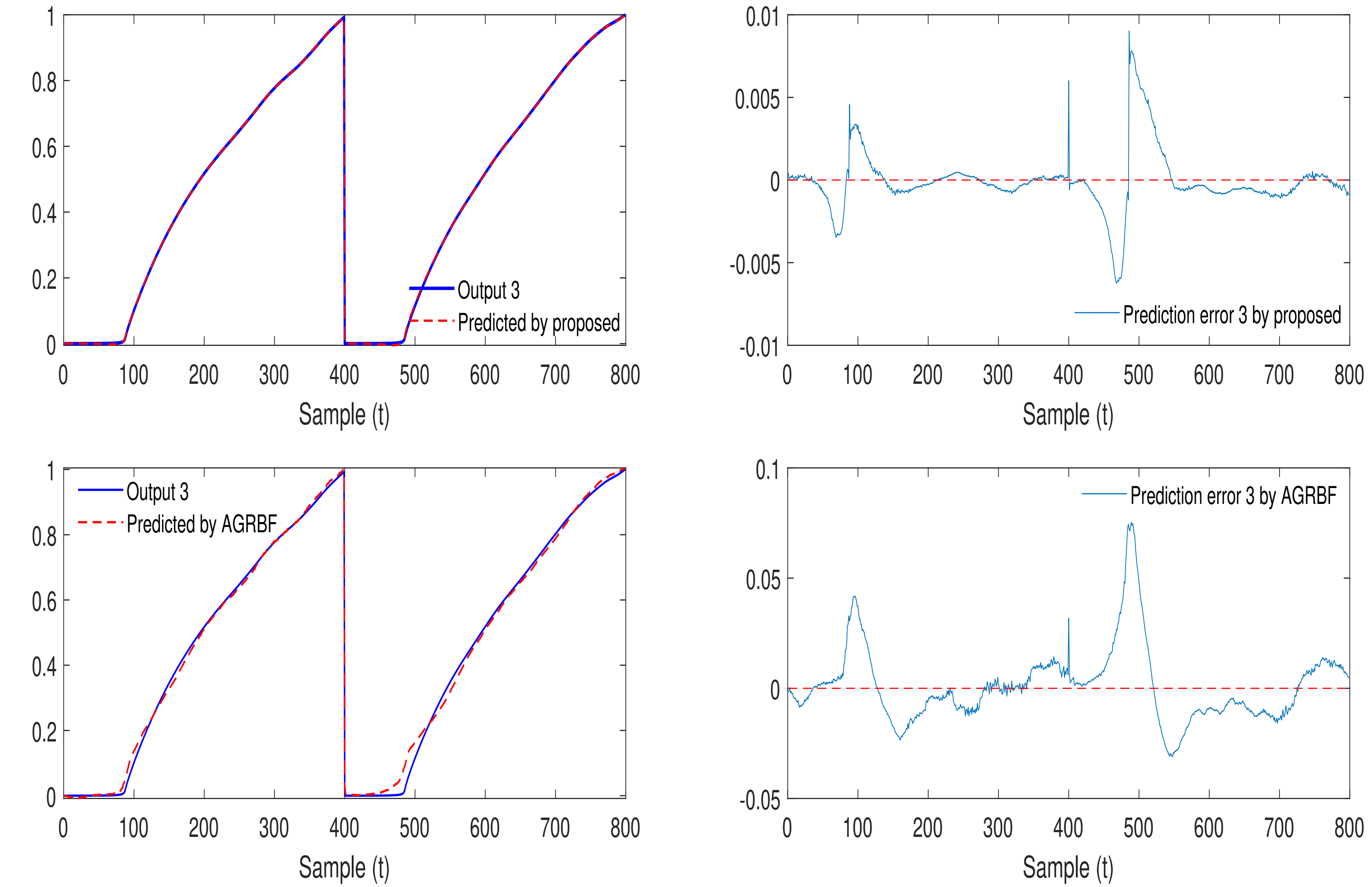
Output One Prediction Performance



Output Two Prediction Performance



Output Three Prediction Performance



Conclusions

- **Deep neural networks**, such as stacked autoencoder, has **deep nonlinear learning** capability, but it is **impossible to adapt** network structure online in real time
- **Shallow gradient RBF** network has **excellent adaptability**
- We have shown how to **integrate deep nonlinear learning** capability of SAE with **excellent adaptability** of adaptive multi-output GRBF
- Proposed deep neural network architecture is capable of adapting to changing underlying system dynamics in **real-time**
  - Particularly suitable for **online modeling** of **highly nonlinear and nonstationary** multi-output industrial processes