Uncertainty quantification for neural state-space models

Marco Forgione, Dario Piga

IDSIA Dalle Molle Institute for Artificial Intelligence USI-SUPSI, Lugano, Switzerland

7th Edition of the Workshop on Nonlinear System Identification Benchmarks
Eindhoven, 19-21 April 2023

Motivations

Neural state-space models are widely used for dynamical systems. We have also applied them to the nonlinear benchmarks!

M. Forgione and D.Piga. Continuous-time system identification with neural networks: model structures and fitting criteria. *European Journal of Control, 2021*

B. Mavkov, M. Forgione and D.Piga. Integrated Neural Networks for Nonlinear Continuous-Time System Identification. *IEEE Control Systems Letters, 4(4), pp 851-856, 2020.*

G. Beintema, R. Toth, and M. Schoukens. Nonlinear state-space identification using deep encoder networks. *In Proc. of the 13rd Conference on Learning for Dynamics and Control, (PMLR), 2021.*

Little focus so far on uncertainty quantification.

- What's our confidence in the model predictions?
- Can we detect model usage outside its validity range?

M. Forgione and D. Piga. Neural state-space models: Empirical evaluation of uncertainty quantification. *Accepted for presentation at the 22nd IFAC World Congress, 2023*

`https://github.com/forgi86/sysid-neural-unc`

Motivations

Neural state-space models are widely used for dynamical systems. We have also applied them to the nonlinear benchmarks!

M. Forgione and D.Piga. Continuous-time system identification with neural networks: model structures and fitting criteria. *European Journal of Control, 2021*

B. Mavkov, M. Forgione and D.Piga. Integrated Neural Networks for Nonlinear Continuous-Time System Identification. *IEEE Control Systems Letters, 4(4), pp 851-856, 2020.*

G. Beintema, R. Toth, and M. Schoukens. Nonlinear state-space identification using deep encoder networks. *In Proc. of the 13rd Conference on Learning for Dynamics and Control, (PMLR), 2021.*

Little focus so far on uncertainty quantification.

- What's our confidence in the model predictions?
- Can we detect model usage outside its validity range?

M. Forgione and D. Piga. Neural state-space models: Empirical evaluation of uncertainty quantification. *Accepted for presentation at the 22nd IFAC World Congress, 2023*

https://github.com/forgi86/sysid-neural-unc

Motivations

Neural state-space models are widely used for dynamical systems. We have also applied them to the nonlinear benchmarks!

M. Forgione and D.Piga. Continuous-time system identification with neural networks: model structures and fitting criteria. *European Journal of Control, 2021*

B. Mavkov, M. Forgione and D.Piga. Integrated Neural Networks for Nonlinear Continuous-Time System Identification. *IEEE Control Systems Letters, 4(4), pp 851-856, 2020.*

G. Beintema, R. Toth, and M. Schoukens. Nonlinear state-space identification using deep encoder networks. *In Proc. of the 13rd Conference on Learning for Dynamics and Control, (PMLR), 2021.*

Little focus so far on uncertainty quantification.

- What's our confidence in the model predictions?
- Can we detect model usage outside its validity range?

M. Forgione and D. Piga. Neural state-space models: Empirical evaluation of uncertainty quantification. *Accepted for presentation at the 22nd IFAC World Congress, 2023*

`https://github.com/forgi86/sysid-neural-unc`

Settings

The true system is assumed to have a state-space representation:

$$x^+ = f(x, u)$$
$$y^\circ = g(x),$$

with noisy discrete-time measurements available: $y = y^\circ + e$.

Training dataset $\mathcal{D}$: a single input/output sequence with:

- input samples $\boldsymbol{u} = \{u_1,\ u_1, \ldots,\ u_N\}$
- output samples $\boldsymbol{y} = \{y_1,\ y_1, \ldots,\ y_N\}$

Objective: estimate a model of the system.

Note: output error model considered here (no process noise).

Settings

The true system is assumed to have a state-space representation:

$$x^+ = f(x, u)$$
$$y^\circ = g(x),$$

with noisy discrete-time measurements available: $y = y^\circ + e$.

Training dataset $\mathcal{D}$: a single input/output sequence with:

- input samples $\boldsymbol{u} = \{u_1,\ u_1, \ldots,\ u_N\}$
- output samples $\boldsymbol{y} = \{y_1,\ y_1, \ldots,\ y_N\}$

Objective: estimate a model of the system.

Note: output error model considered here (no process noise).

Settings

The true system is assumed to have a state-space representation:

$$x^+ = f(x, u)$$
$$y^\circ = g(x),$$

with noisy discrete-time measurements available: $y = y^\circ + e$.

Training dataset $\mathcal{D}$: a single input/output sequence with:

- input samples $\boldsymbol{u} = \{u_1,\ u_1, \ldots,\ u_N\}$
- output samples $\boldsymbol{y} = \{y_1,\ y_1, \ldots,\ y_N\}$

Objective: estimate a model of the system.

Note: output error model considered here (no process noise).

Settings

The true system is assumed to have a state-space representation:

$$x^+ = f(x, u)$$
$$y^\circ = g(x),$$

with noisy discrete-time measurements available: $y = y^\circ + e$.

Training dataset $\mathcal{D}$: a single input/output sequence with:

- input samples $\boldsymbol{u} = \{u_1,\ u_1, \ldots,\ u_N\}$
- output samples $\boldsymbol{y} = \{y_1,\ y_1, \ldots,\ y_N\}$

Objective: estimate a model of the system.

Note: output error model considered here (no process noise).

Neural state-space models

A very generic neural model structure:

$$x^+ = \mathcal{N}_f(x, u;\ \theta)$$
$$y = \mathcal{N}_g(x; \theta)$$

where $\mathcal{N}_f$, $\mathcal{N}_g$ are feed-forward neural networks, e.g.:

$$\mathcal{N}(z) = W_2\sigma\left(W_1 z + b_1\right) + b_2$$
$$\theta = \text{vec}(W_1, b_1, W_2, b_2)$$
$\sigma(\cdot)$ static non-linearity

Can be specialized, according to available system knowledge:

- State fully observed $\Rightarrow$

$$x^+ = \mathcal{N}_f(x, u;\ \theta)$$
$$y = x$$

- ...

Neural state-space models

A very generic neural model structure:

$$x^+ = \mathcal{N}_f(x, u;\ \theta)$$
$$y = \mathcal{N}_g(x; \theta)$$

where $\mathcal{N}_f$, $\mathcal{N}_g$ are feed-forward neural networks, e.g.:

$$\mathcal{N}(z) = W_2\sigma\left(W_1 z + b_1\right) + b_2$$
$$\theta = \text{vec}(W_1, b_1, W_2, b_2)$$
$$\sigma(\cdot) \text{ static non-linearity}$$

Can be specialized, according to available system knowledge:

- State fully observed $\Rightarrow$

$$x^+ = \mathcal{N}_f(x, u;\ \theta)$$
$$y = x$$

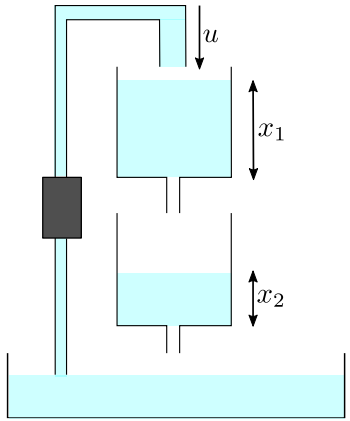
- ...

Neural state-space models

Physics-inspired model structures

Two-tank system. Input: flow u in upper tank, output: lower tank level x_2.

- The system has two states: x_1 and x_2
- State x_1 does not depend on x_2
- State x_2 does not depend directly on u
- The state x_2 is measured



These observations are embedded in the physics-inspired neural model:

$$\dot{x}_1 = \mathcal{N}_1(x_1, u;\ \theta)$$
$$\dot{x}_2 = \mathcal{N}_2(x_1, x_2;\ \theta)$$
$$y = x_2$$

M. Forgione and D.Piga. Continuous-time system identification with neural networks: model structures and fitting criteria. *European Journal of Control, 2021*

Bayesian Framework

We can frame model learning problem in a Bayesian framework:

$$
\begin{aligned}
x^+ &= \mathcal{N}_f(x, u; \theta) \\
\hat{y} &= \mathcal{N}_g(x; \theta) \\
y &= \hat{y} + e \\
\theta &\sim \mathcal{N}\left(0, \sigma_\theta^2\right), \quad e \sim \mathcal{N}\left(0, \sigma_e^2\right)
\end{aligned}
$$

Then, there exist the posterior of θ given observed data $\mathcal{D} = \{\boldsymbol{u}, \boldsymbol{y}\}$ ...

$$
p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})},
$$

... and the posterior on a new sequence $\boldsymbol{y}^*$, given input sequence $\boldsymbol{u}^*$:

$$
p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D}) = \int_{\theta \in \Theta} p(\boldsymbol{y}^*|\boldsymbol{u}^*, \theta) p(\theta|\mathcal{D}) \, d\theta
$$

Bayesian Framework

We can frame model learning problem in a Bayesian framework:

$$
\begin{aligned}
x^+ &= \mathcal{N}_f(x, u; \theta) \\
\hat{y} &= \mathcal{N}_g(x; \theta) \\
y &= \hat{y} + e \\
\theta &\sim \mathcal{N}\left(0, \sigma_\theta^2\right), \quad e \sim \mathcal{N}\left(0, \sigma_e^2\right)
\end{aligned}
$$

Then, there exist the posterior of θ given observed data $\mathcal{D} = \{\boldsymbol{u}, \boldsymbol{y}\}$ ...

$$
p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})},
$$

... and the posterior on a new sequence $\boldsymbol{y}^*$, given input sequence $\boldsymbol{u}^*$:

$$
p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D}) = \int_{\theta \in \Theta} p(\boldsymbol{y}^*|\boldsymbol{u}^*, \theta) p(\theta|\mathcal{D}) \, d\theta
$$

Bayesian Framework

We can frame model learning problem in a Bayesian framework:

$$
\begin{aligned}
x^+ &= \mathcal{N}_f(x, u; \theta) \\
\hat{y} &= \mathcal{N}_g(x; \theta) \\
y &= \hat{y} + e \\
\theta &\sim \mathcal{N}\left(0, \sigma_\theta^2\right), \quad e \sim \mathcal{N}\left(0, \sigma_e^2\right)
\end{aligned}
$$

Then, there exist the posterior of θ given observed data $\mathcal{D} = \{\boldsymbol{u}, \boldsymbol{y}\}$ ...

$$
p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})},
$$

... and the posterior on a new sequence $\boldsymbol{y}^*$, given input sequence $\boldsymbol{u}^*$:

$$
p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D}) = \int_{\theta \in \Theta} p(\boldsymbol{y}^*|\boldsymbol{u}^*, \theta) p(\theta|\mathcal{D}) \, d\theta
$$

Challenges

- What are good Bayesian models structures (architecture + priors)?
  - What families of dynamical systems are likely under given assumptions?
- How to obtain the posterior parameter distribution $p(\theta|\mathcal{D})$ and of the posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D})$?
  - $p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$ is accessible up to the denominator constant.
  - $p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D}) = \int_{\theta\in\Theta} p(\boldsymbol{y}^*|\boldsymbol{u}^*, \theta)p(\theta|\mathcal{D})\, d\theta$ is a hard multi-dimensional integral anyway...

In this preliminary contribution we do not provide an answer to 1 and use the simplest approach to tackle 2: Laplace approximation + linearization.

Challenges

- What are good Bayesian models structures (architecture + priors)?
  - What families of dynamical systems are likely under given assumptions?
- How to obtain the posterior parameter distribution $p(\theta|\mathcal{D})$ and of the posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D})$?
  - $p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$ is accessible up to the denominator constant.
  - $p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D}) = \int_{\theta \in \Theta} p(\boldsymbol{y}^*|\boldsymbol{u}^*, \theta) p(\theta|\mathcal{D})\, d\theta$ is a hard multi-dimensional integral anyway...

In this preliminary contribution we do not provide an answer to 1 and use the simplest approach to tackle 2: Laplace approximation + linearization.

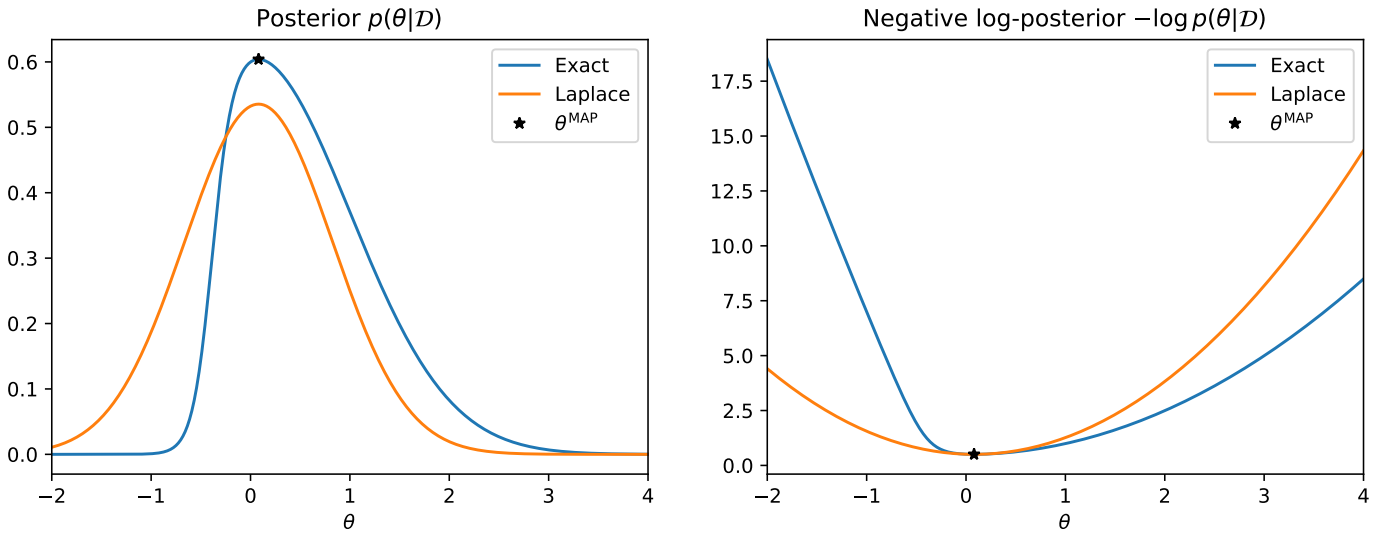
Challenges

- What are good Bayesian models structures (architecture + priors)?
  - What families of dynamical systems are likely under given assumptions?
- How to obtain the posterior parameter distribution $p(\theta|\mathcal{D})$ and of the posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D})$?
  - $p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$ is accessible up to the denominator constant.
  - $p(\boldsymbol{y}^*|\boldsymbol{u}^*, \mathcal{D}) = \int_{\theta\in\Theta} p(\boldsymbol{y}^*|\boldsymbol{u}^*, \theta)p(\theta|\mathcal{D})\, d\theta$ is a hard multi-dimensional integral anyway...

In this preliminary contribution we do not provide an answer to 1 and use the simplest approach to tackle 2: Laplace approximation + linearization.

Laplace approximation

Quadratic approximation of the (negative) log-posterior around a *mode*, leading to a Gaussian parameter posterior approximation.



Requires computation of:

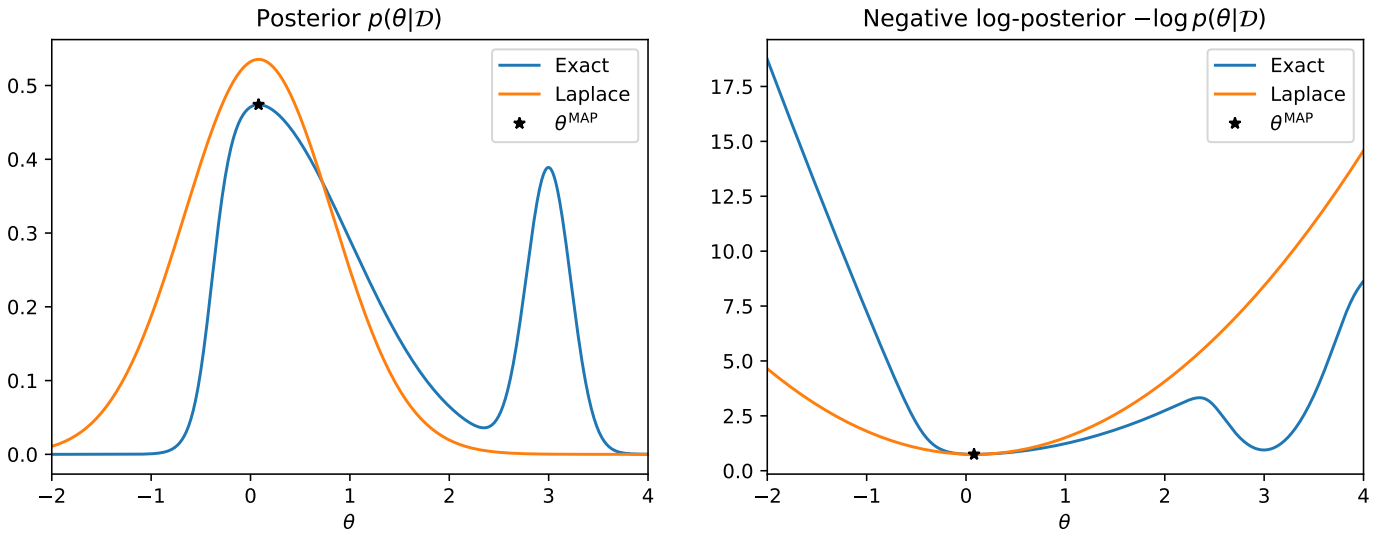
- A point estimate $\Rightarrow$ maximimum θ^{MAP} of $\log p(\theta|\mathcal{D})$
- Curvature arount the point $\Rightarrow$ Hessian H of $\log p(\theta|\mathcal{D})$ in θ^{MAP}

that define the posterior mean and inverse covariance, respectively.

It ignores multiple modes (common in neural networks).

Laplace approximation

Quadratic approximation of the (negative) log-posterior around a *mode*, leading to a Gaussian parameter posterior approximation.



Requires computation of:

- A point estimate $\Rightarrow$ maximimum θ^{MAP} of $\log p(\theta|\mathcal{D})$
- Curvature arount the point $\Rightarrow$ Hessian H of $\log p(\theta|\mathcal{D})$ in θ^{MAP}

that define the posterior mean and inverse covariance, respectively.
It ignores multiple modes (common in neural networks).

Laplace method + Linearization

Tested for small ($\approx$ 400 params) neural state-space models:

- Posterior parameter distribution approximated with Laplace's method. $p(\theta|\mathcal{D})$ Gaussian with covariance:

$$P_\theta^{-1} = H = \frac{I}{\sigma_\theta^2} + \frac{1}{\sigma_e^2} \sum_{k=0}^{N-1} \frac{\partial y_k}{\partial \theta} \frac{\partial y_k}{\partial \theta}^\top .$$

  - Gauss-Newton (GN) approximation for the log-likelihood Hessian H
  - Could also be seen as the Fisher matrix in a frequentist setting

- Posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{u}, \mathcal{D})$ based on linearization
  Predictions also Gaussian, centered about nominal $\hat{\boldsymbol{y}}^*$, covariance:

$$\Sigma_{\boldsymbol{y}^*} = J^* P_\theta J^{*\top}$$

Technical challenge: fast computation of the Hessian extending methods recently developed in another work.

M. Forgione, A. Muni, D. Piga, and M. Gallieri. On the adaptation of recurrent neural networks for system identification. *Accepted for publication in Automatica, 2023*

Laplace method + Linearization

Tested for small ($\approx$ 400 params) neural state-space models:

- Posterior parameter distribution approximated with Laplace's method. $p(\theta|\mathcal{D})$ Gaussian with covariance:

$$P_\theta^{-1} = H = \frac{I}{\sigma_\theta^2} + \frac{1}{\sigma_e^2} \sum_{k=0}^{N-1} \frac{\partial y_k}{\partial \theta} \frac{\partial y_k}{\partial \theta}^\top .$$

  - Gauss-Newton (GN) approximation for the log-likelihood Hessian H
  - Could also be seen as the Fisher matrix in a frequentist setting
- Posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{u}, \mathcal{D})$ based on linearization
  Predictions also Gaussian, centered about nominal $\hat{\boldsymbol{y}}^*$, covariance:

$$\Sigma_{\boldsymbol{y}^*} = J^* P_\theta J^{*\top}$$

Technical challenge: fast computation of the Hessian extending methods recently developed in another work.

M. Forgione, A. Muni, D. Piga, and M. Gallieri. On the adaptation of recurrent neural networks for system identification. *Accepted for publication in Automatica, 2023*

Laplace method + Linearization

Tested for small ($\approx$ 400 params) neural state-space models:

- Posterior parameter distribution approximated with Laplace's method. $p(\theta|\mathcal{D})$ Gaussian with covariance:

$$P_\theta^{-1} = H = \frac{I}{\sigma_\theta^2} + \frac{1}{\sigma_e^2} \sum_{k=0}^{N-1} \frac{\partial y_k}{\partial \theta} \frac{\partial y_k}{\partial \theta}^\top .$$

  - Gauss-Newton (GN) approximation for the log-likelihood Hessian H
  - Could also be seen as the Fisher matrix in a frequentist setting
- Posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{u}, \mathcal{D})$ based on linearization
  Predictions also Gaussian, centered about nominal $\hat{\boldsymbol{y}}^*$, covariance:

$$\Sigma_{\boldsymbol{y}^*} = J^* P_\theta J^{*\top}$$

Technical challenge: fast computation of the Hessian extending methods recently developed in another work.

M. Forgione, A. Muni, D. Piga, and M. Gallieri. On the adaptation of recurrent neural networks for system identification. *Accepted for publication in Automatica, 2023*

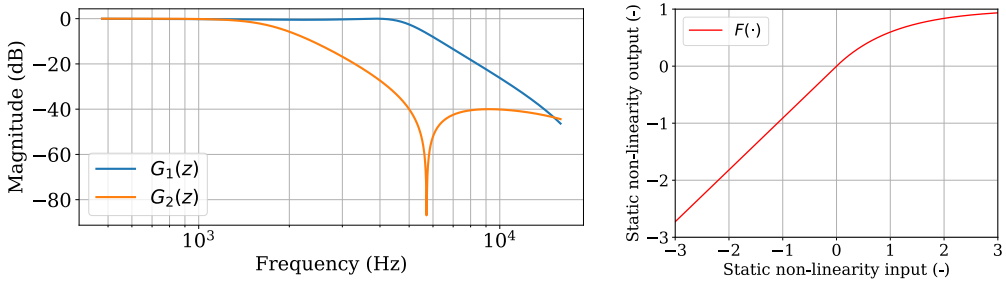
Results: synthetic WH benchmark

Methodology tested on a simulated Wiener-Hammerstein system

model:

$$x^+ = \mathcal{N}_f(x, u; \theta)$$
$$\hat{y} = \mathcal{N}_g(x; \theta)$$

system: $G_1(z) \Rightarrow F(\cdot) \Rightarrow G_2(z)$



$\mathcal{N}_f, \mathcal{N}_g$: 1 hidden layer, 15 neurons + linear term, as in previous literature

Training on a multisine with std 0.4, bandwidth [0 2] kHz, 10K samples.

Testing on:

1. A multisine with std 0.4, bandwidth [0 2] kHz
2. A multisine with std 0.4, bandwidth [1 2] kHz
3. A multisine with std 0.8, bandwidth [0 2] kHz
4. A multisine with std 0.4, bandwidth [0 10] kHz

We expect lower performance and higher uncertainty on signals 3 and 4.

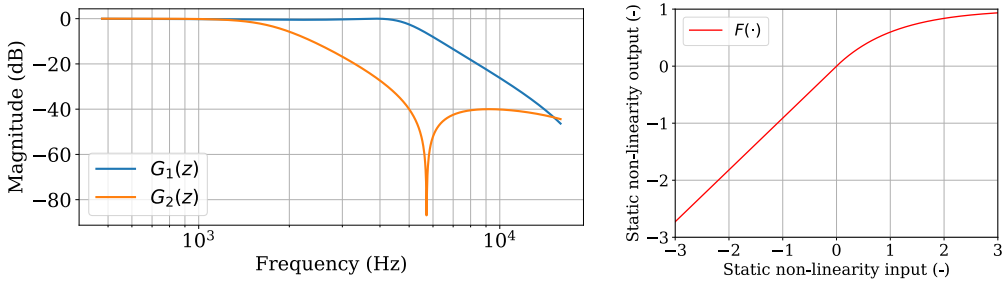
Results: synthetic WH benchmark

Methodology tested on a simulated Wiener-Hammerstein system

model:

$$x^+ = \mathcal{N}_f(x, u; \theta)$$
$$\hat{y} = \mathcal{N}_g(x; \theta)$$

system: $G_1(z) \Rightarrow F(\cdot) \Rightarrow G_2(z)$



$\mathcal{N}_f, \mathcal{N}_g$: 1 hidden layer, 15 neurons + linear term, as in previous literature
Training on a multisine with std 0.4, bandwidth [0 2] kHz, 10K samples.

Testing on:

1. A multisine with std 0.4, bandwidth [0 2] kHz
2. A multisine with std 0.4, bandwidth [1 2] kHz
3. A multisine with std 0.8, bandwidth [0 2] kHz
4. A multisine with std 0.4, bandwidth [0 10] kHz

We expect lower performance and higher uncertainty on signals 3 and 4.

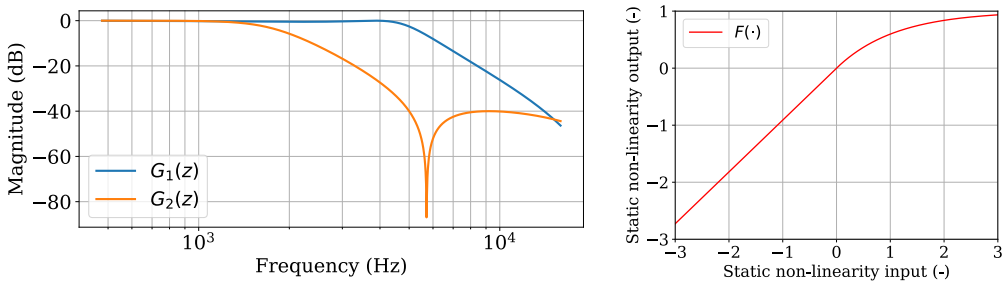
Results: synthetic WH benchmark

Methodology tested on a simulated Wiener-Hammerstein system

model:

$$x^+ = \mathcal{N}_f(x, u; \theta)$$
$$\hat{y} = \mathcal{N}_g(x; \theta)$$

system: $G_1(z) \Rightarrow F(\cdot) \Rightarrow G_2(z)$



$\mathcal{N}_f, \mathcal{N}_g$: 1 hidden layer, 15 neurons + linear term, as in previous literature
Training on a multisine with std 0.4, bandwidth [0 2] kHz, 10K samples.
Testing on:

1. A multisine with std 0.4, bandwidth [0 2] kHz
2. A multisine with std 0.4, bandwidth [1 2] kHz
3. A multisine with std 0.8, bandwidth [0 2] kHz
4. A multisine with std 0.4, bandwidth [0 10] kHz

We expect lower performance and higher uncertainty on signals 3 and 4.

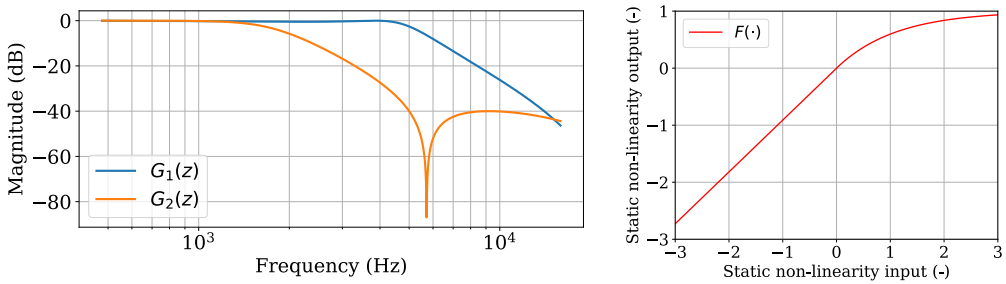
Results: synthetic WH benchmark

Methodology tested on a simulated Wiener-Hammerstein system

model:

$$x^+ = \mathcal{N}_f(x, u; \theta)$$
$$\hat{y} = \mathcal{N}_g(x; \theta)$$

system: $G_1(z) \Rightarrow F(\cdot) \Rightarrow G_2(z)$



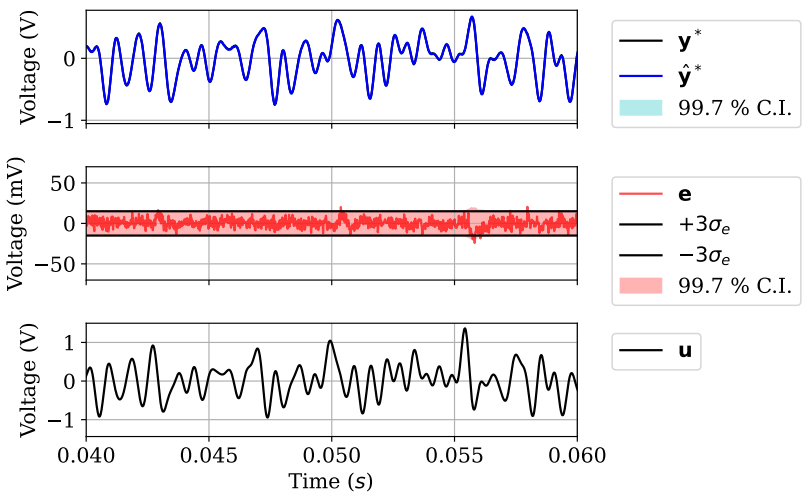
$\mathcal{N}_f, \mathcal{N}_g$: 1 hidden layer, 15 neurons + linear term, as in previous literature
Training on a multisine with std 0.4, bandwidth [0 2] kHz, 10K samples.
Testing on:

1. A multisine with std 0.4, bandwidth [0 2] kHz
2. A multisine with std 0.4, bandwidth [1 2] kHz
3. A multisine with std 0.8, bandwidth [0 2] kHz
4. A multisine with std 0.4, bandwidth [0 10] kHz

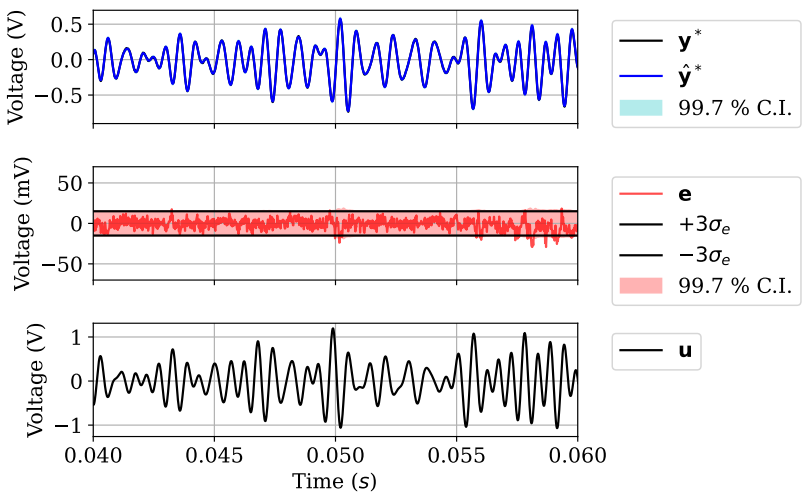
We expect lower performance and higher uncertainty on signals 3 and 4.

Results: synthetic WH benchmark

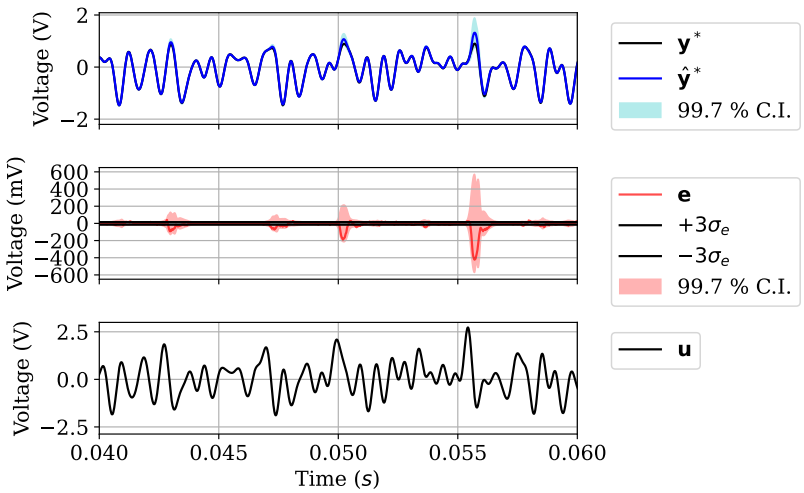
Test signal 1



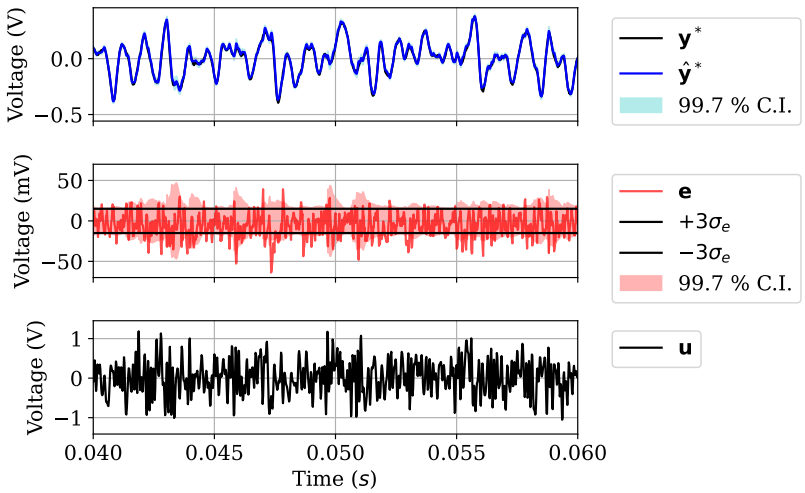
Test signal 2



Test signal 3



Test signal 4

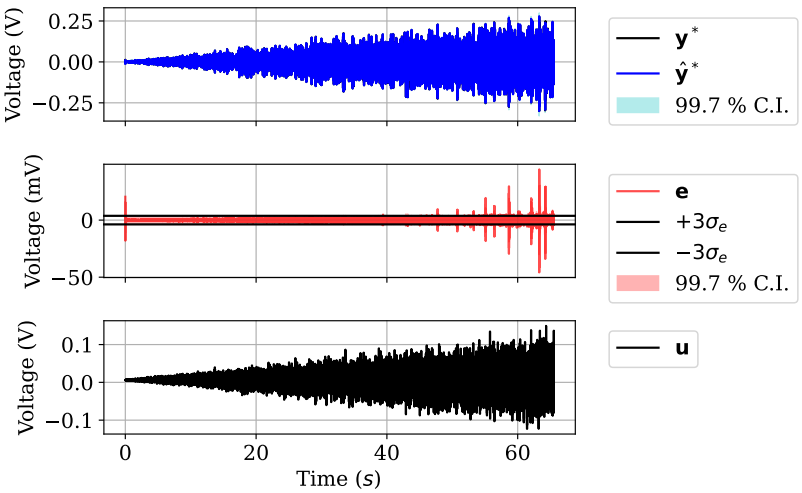


Credible intervals get indeed wider for test signals 3 and 4!

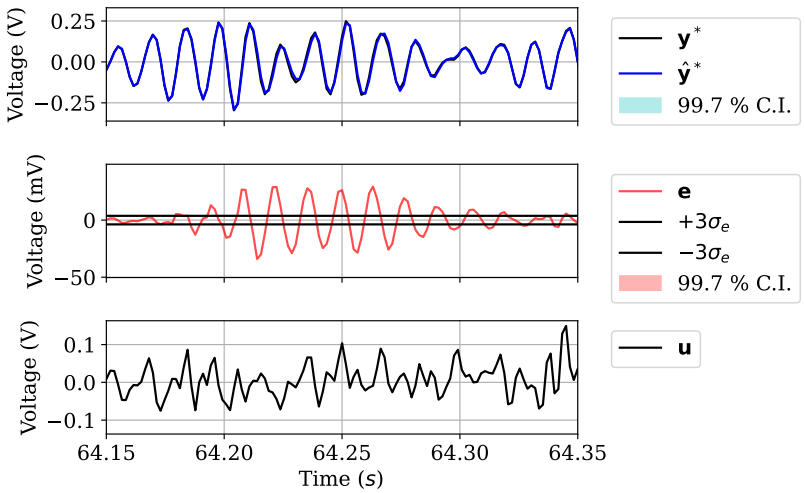
Results: Silverbox benchmark

Methodology also tested on the original Silverbox dataset

Test set (full)



Test set (zoom)



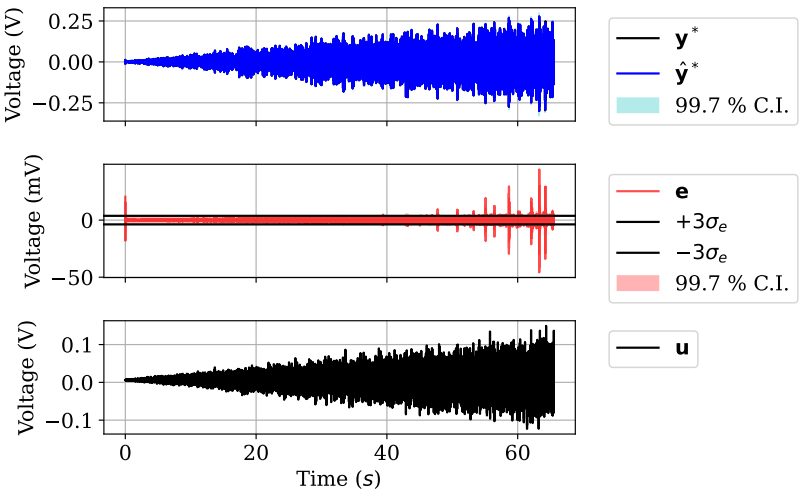
- Large errors towards the end of the ramp test signal (left plot)
- Credible intervals useful, though not calibrated (right plot)

Large values not seen in training, therefore high uncertainty towards the end of the ramp.

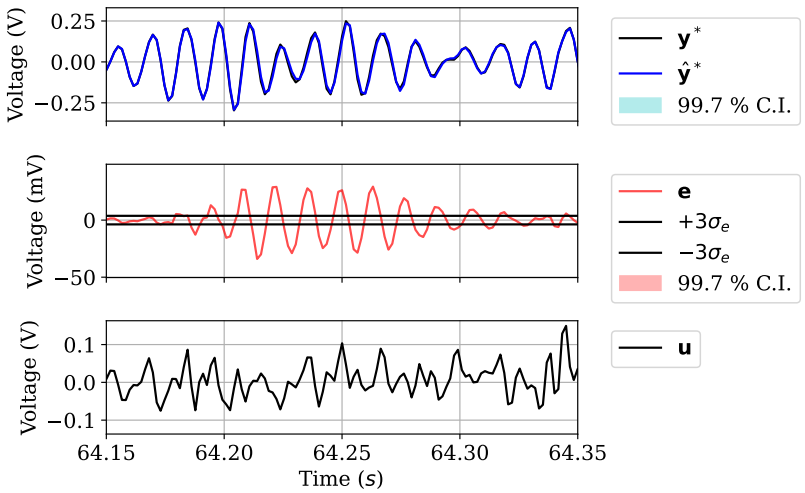
Results: Silverbox benchmark

Methodology also tested on the original Silverbox dataset

Test set (full)



Test set (zoom)



- Large errors towards the end of the ramp test signal (left plot)
- Credible intervals useful, though not calibrated (right plot)

Large values not seen in training, therefore high uncertainty towards the end of the ramp.

Conclusions

Preliminary uncertainty quantification for neural state-space models.

- In a Bayesian Framework
- Using the simplest inference techniques

Obtained credible intervals not well calibrated, but allow one to detect out-of-distribution regimes.

Large room for improvement, both in the theoretical framework and in the approximate inference engine. Benchmarks and metrics also to be devised!

Open opportunity: to exploit the uncertainty for Experiment Design!

Conclusions

Preliminary uncertainty quantification for neural state-space models.

- In a Bayesian Framework
- Using the simplest inference techniques

Obtained credible intervals not well calibrated, but allow one to detect out-of-distribution regimes.

Large room for improvement, both in the theoretical framework and in the approximate inference engine. Benchmarks and metrics also to be devised!

Open opportunity: to exploit the uncertainty for Experiment Design!

Conclusions

Preliminary uncertainty quantification for neural state-space models.

- In a Bayesian Framework
- Using the simplest inference techniques

Obtained credible intervals not well calibrated, but allow one to detect out-of-distribution regimes.

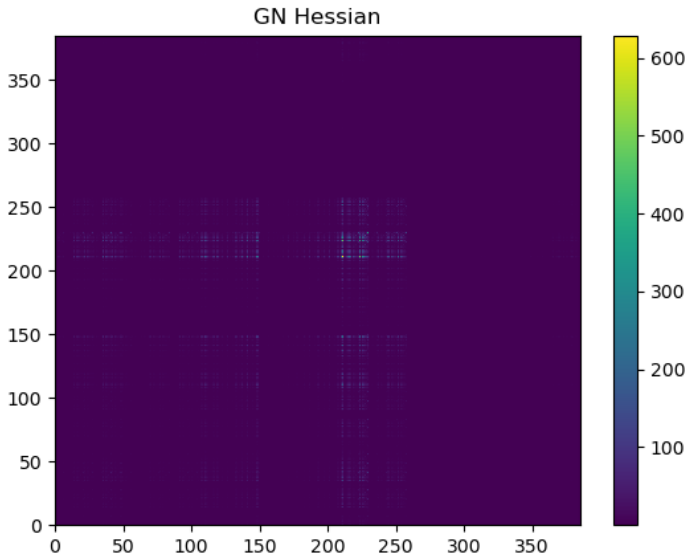
Large room for improvement, both in the theoretical framework and in the approximate inference engine. Benchmarks and metrics also to be devised!

Open opportunity: to exploit the uncertainty for Experiment Design!

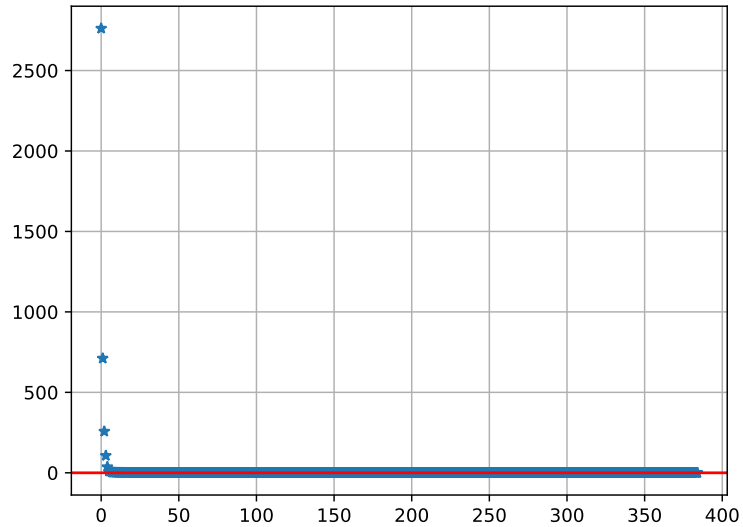
Bonus topic: Hessian Matrix Analysis

Wiener-Hammerstein: Hessian matrix at optimum:

GN Hessian Matrix abs



GN Hessian Matrix eigenvalues

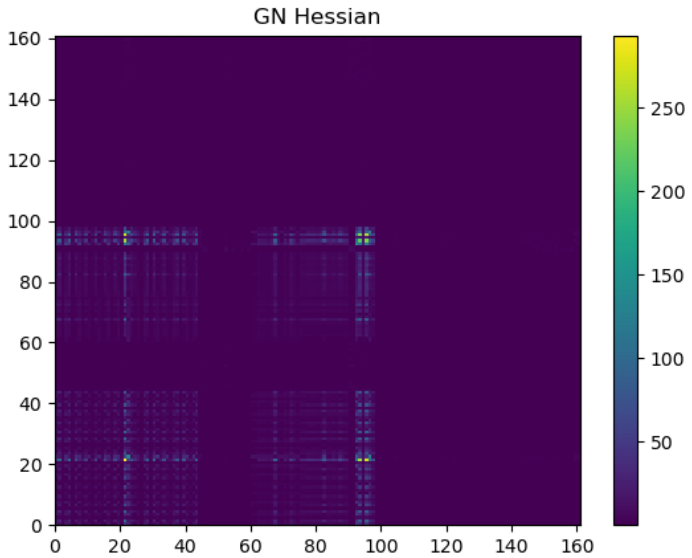


The Hessian is non-degenerate only in ~ 7 directions.

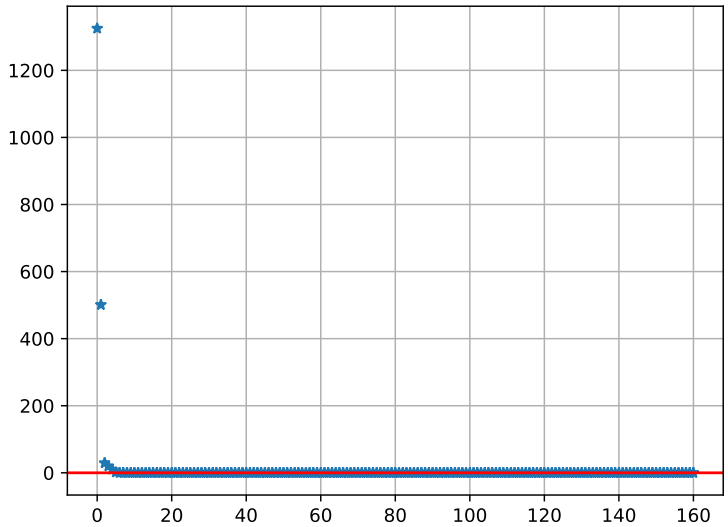
Bonus topic: Hessian Matrix Analysis

Silverbox: Hessian matrix at optimum:

GN Hessian Matrix abs



GN Hessian Matrix eigenvalues



The Hessian is non-degenerate only in $\sim$ 6 directions.

Thank you.
Questions?

marco.forgione@idsia.ch