DTU

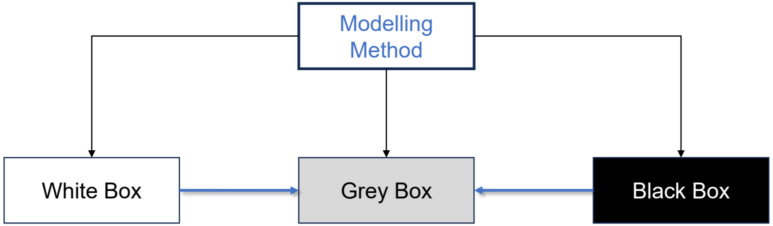
Industrial IoT for Digitization of Electronis Assets

Digital Twins, Models, and Parameters Estimation

Agenda

- White, Grey and Black Box Models
- LTI system and properties
- Autoregressive Model in System Identification
- Autoregressive with eXogenous
- Parameters Estimation
- Examples in Python
- Validation and Residual Analysis
- Order Selection
- AI & ML in System Identification

Model Types



White Box Models

- The structure of the model is known and is developed from fundamental laws of physics, thermodynamics, and heat transfer;
- The model parameters are well knownand used as inputs to the model.

Grey Box Models

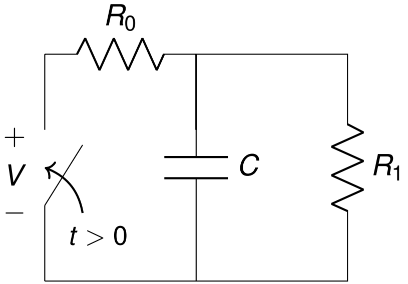
- Combination of black box and white box models.
- When data-driven models (black box) are partly supported by explicit models (equations) with a physical meaning.

Black Box Models

- Purely data-driven/statistical/empirical models
- Uses mathematical equations from statistics to map influential inputs to the outputs.
- The source of data: on-board monitored data from sensors, data simulated from simulation tools, surveyor standard data from public benchmark datasets.

Example of White Box Model

We know the physics, so we can write the equations:

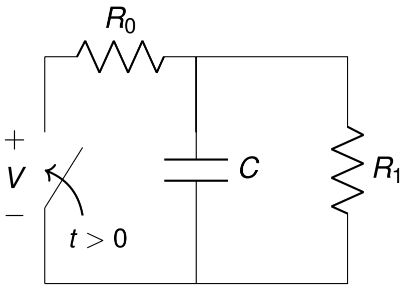


$$I = \frac{1}{R_1} V_C + C \frac{d}{dt} V_C$$
$$V = R_0 I + V_C$$

Willems, J. C., & Polderman, J. W. (1997). Introduction to mathematical systems theory: a behavioral approach (Vol. 26). Springer Science & Business Media.

Example of White Box Model

We know the physics, so we can write the equations:



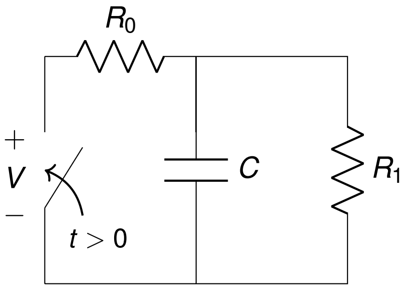
$$I = \frac{1}{R_1} V_C + C \frac{d}{dt} V_C$$
$$V = R_0 I + V_C$$

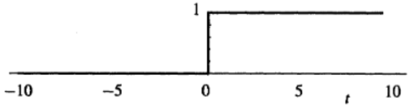
$$V + CR_1 \frac{d}{dt} V = (R_0 + R_1) I + CR_0 R_1 \frac{d}{dt} I$$

Willems, J. C., & Polderman, J. W. (1997). Introduction to mathematical systems theory: a behavioral approach (Vol. 26). Springer Science & Business Media.

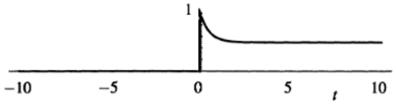
Example of White Box Model

At time $t < 0$, the circuit is shorted ($V = 0$) and at $t = 0$ a 1 V battery is attached.





(a) Voltage step function.



(b) The current response.

Willems, J. C., & Polderman, J. W. (1997). Introduction to mathematical systems theory: a behavioral approach (Vol. 26). Springer Science & Business Media.

Example of Grey Box Model

Stochastic Models

$$dx_t = f(x_t, u_t, t, \theta) + \sigma(u_t, t, \theta) d\omega$$
$$y_k = h(x_k, u_k, t_k, \theta) + e_k$$

Suitable for both linear, non-linear models. Used to represent systems influenced by both deterministic and stochastic components, accounting for random fluctuations in the system and uncertainties.

Example of Grey Box Model

Stochastic Models

$$dx_t = f(x_t, u_t, t, \theta) + \sigma(u_t, t, \theta)d\omega$$
$$y_k = h(x_k, u_k, t_k, \theta) + e_k$$

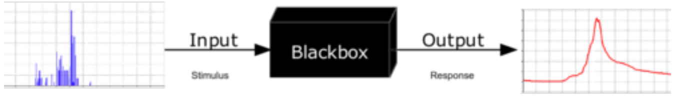
Suitable for both linear, non-linear models. Used to represent systems influenced by both deterministic and stochastic components, accounting for random fluctuations in the system and uncertainties.

PINNs:Physics-informed neural networks

A class of machine learning techniques that combine data-driven neural networks with physical equations to solve complex problems in various fields, such as physics, and engineering, minimizing the discrepancy between the predictions made by the neural network and physical equations.

Example of Black Box Models

A black box model is a system or algorithm that makes predictions or decisions based solely on input and output data, without an interpretable framework that can explain the connection between the inputs and the outputs.



AutoRegressive Model (AR)

First Order Model

Let's consider the simplest form of an AR model:

$$y(t) = a_1 y(t-1) + \epsilon$$

Where $\boldsymbol{y_{t-1}}$ is the *lag* of $\boldsymbol{y}$ at time $t-1$ and ϵ an error term.

n_a order model

$$y(t) = a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) + \epsilon_t$$

Definition of Dynamical System

A dynamical system Σ is defined as a triple

$$\Sigma = (T, W, \mathscr{B}),$$

- T a subset of $\mathbb{Z}_+$ (in *discrete-time systems*).
- W a set called the *signal space*.
- $\mathscr{B}$ a subset of W^T called the *behavior*.

Willems, J. C., & Polderman, J. W. (1997). Introduction to mathematical systems theory: a behavioral approach (Vol. 26). Springer Science & Business Media.

Linear Time-Invariant (LTI) Dynamic Systems

Definition:

A *Linear Time-Invariant* (LTI) system is a dynamical system where the principles of superposition (linearity) and shift invariance (time invariance) hold.

Linear Time-Invariant (LTI) Dynamic Systems

Definition:

A *Linear Time-Invariant* (LTI) system is a dynamical system where the principles of superposition (linearity) and shift invariance (time invariance) hold.

Properties:

- **Linearity:** Given $y_1(t)$ and $y_2(t)$ the outputs corresponding to inputs $x_1(t)$ and $x_2(t)$, then for any constants a and b, the system's response to $ax_1(t) + bx_2(t)$ is $ay_1(t) + by_2(t)$.
- **Time Invariance:** If the output of the system to an input $x(t)$ is $y(t)$, then the output to an input $x(t - t_0)$, for any time shift t_0, is $y(t - t_0)$.

AutoRegressive eXogenous Model (ARX)

Given an **LTI** system, with $\boldsymbol{y_t}$ the output signal *with exogenous* input $\boldsymbol{u_t}$ with delay k, an ARX model can be defined as follows:

$$y(t) = c + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) + \\ + b_1 u(t-n_k) + \cdots + b_{n_b} u(t-n_k-n_b-1) + \epsilon(t)$$

- $\boldsymbol{n_a}$: numbers of lags of the output signal y.
- $\boldsymbol{n_b}$: numbers of lags of input signal$u$.
- $\boldsymbol{n_k}$: delay order between the y and u.

$$\boxed{y(t) = \varphi'\theta + \epsilon(t)}$$

AutoRegressive eXogenous Model (ARX)

Or equivalently:

$$\boxed{y(t) = \varphi'\theta + \epsilon(t)}$$

$$\theta = [a_1, \ldots, a_{n_a}\ b_1, \ldots, b_{n_b}]' \quad \textit{unknown paramters}$$

$$\varphi(t) = [y(t-1), \ldots, y(t-n_a)\ u(t-n_k) \ldots, u_{t-n_k-n_b+1}]' \quad \textit{regressors}$$

$$\epsilon(t) \sim \mathcal{N}(0, \sigma^2) \quad \textit{error}$$

AutoRegressive eXogenous Model (ARX)

Or equivalently:

$$\boxed{y(t) = \varphi'\theta + \epsilon(t)}$$

$$\begin{aligned}\theta &= [a_1, \ldots, a_{n_a}\ b_1, \ldots, b_{n_b}]' \ \textit{unknown paramters}\\ \varphi(t) &= [y(t-1), \ldots, y(t-n_a)\ u(t-n_k) \ldots, u_{t-n_k-n_b+1}]' \ \textit{regressors}\\ \epsilon(t) &\sim \mathcal{N}(0, \sigma^2) \ \textit{error}\end{aligned}$$

Therefore, for a SISO (Single Input Single Output) time series, y_{t+1} can be written as:

$$\boxed{\hat{y}(t+1|\theta) = \varphi(t)'\theta}$$

Remark

We denote $\hat{\boldsymbol{y}}(\boldsymbol{t}+\mathbf{1}|\boldsymbol{\theta}) = \boldsymbol{\varphi}(\boldsymbol{t})'\boldsymbol{\theta}$ the estimated output prediction to emphasize that the estimate of $\hat{y}(t+1|\theta)$ depends from the past data on the paramters vector θ with $\epsilon = 0$ (best case scenarion - **no residual error**).

Remark

We denote $\hat{\boldsymbol{y}}(\boldsymbol{t}+\mathbf{1}|\boldsymbol{\theta}) = \varphi(\boldsymbol{t})'\boldsymbol{\theta}$ the estimated output prediction to emphasize that the estimate of $\hat{y}(t+1|\theta)$ depends from the past data on the paramters vector θ with $\epsilon = 0$ (best case scenarion - **no residual error**).

Remark

We denote $\hat{\boldsymbol{y}}(\boldsymbol{t}+\mathbf{1}|\boldsymbol{\theta}) = \varphi(\boldsymbol{t})'\boldsymbol{\theta}$ the estimated output prediction to emphasize that the estimate of $\hat{y}(t+1|\theta)$ depends from the past data on the paramters vector θ with $\epsilon = 0$ (best case scenarion - **no residual error**).

- $\varphi' \rightarrow$ regressor vector (past data)
- $\epsilon(\boldsymbol{t}) = 0 \rightarrow$ (known distribution)

What is still missing?

Remark

We denote $\hat{\boldsymbol{y}}(\boldsymbol{t}+\mathbf{1}|\boldsymbol{\theta}) = \boldsymbol{\varphi}(\boldsymbol{t})'\boldsymbol{\theta}$ the estimated output prediction to emphasize that the estimate of $\hat{y}(t+1|\theta)$ depends from the past data on the paramters vector θ with $\epsilon = 0$ (best case scenarion - **no residual error**).

- $\varphi' \rightarrow$ regressor vector (past data)
- $\boldsymbol{\epsilon}(\boldsymbol{t}) = 0 \rightarrow$ (known distribution)

What is still missing?

Remark

We denote $\hat{\boldsymbol{y}}(\boldsymbol{t}+\mathbf{1}|\boldsymbol{\theta}) = \varphi(\boldsymbol{t})'\boldsymbol{\theta}$ the estimated output prediction to emphasize that the estimate of $\hat{y}(t+1|\theta)$ depends from the past data on the paramters vector θ with $\epsilon = 0$ (best case scenarion - **no residual error**).

- $\varphi' \rightarrow$ regressor vector (past data)
- $\epsilon(\boldsymbol{t}) = 0 \rightarrow$ (known distribution)
- $\boldsymbol{\theta}$ is the coefficients vector (to be estimated)

Parameters Estimation

We don't know θ yet, but we have collected a set Z^N of measured data.

$$Z^N = \{u(-n), y(-n) \dots u(N-1), y(N-1)\}, \; n = max\{n_a, n_b + n_k - 1\}$$

Therefore, we use the least squared error to find the optimal paramters vector θ^* that minimize the prediction error between $\hat{y}(t+1|\theta)$ and $y(t)$, namely:

$$\theta^* = \arg\min_{\theta}\{\mathcal{L}(\theta, Z^N)\}$$

$$\mathcal{L}(\theta, Z^N) = \frac{1}{N}\sum_{k=0}^{N-1}(y(k) - \hat{y}(t|\theta))^2 = \frac{1}{N}\sum_{k=0}^{N-1}(y(k) - \varphi'(t)\theta)^2$$

Paramters Estimation

$\mathcal{L}(\theta, Z^N)$ is a quadratic function of θ. Therefore, θ^* can be found by zeroing The derivative of $\mathcal{L}$.

$$\frac{d}{d\theta}\mathcal{L}_N(\theta, Z^N) = \frac{2}{N}\sum_{k=0}^{N-1}\varphi(t)(y(k) - \varphi'(t)\theta) = 0$$

Thus, obtaining:

$$\sum_{k=0}^{N-1}\varphi(t)y(k) = \sum_{k=0}^{N-1}\varphi(t)\varphi'(k)$$

Finally, the θ^* can be derived:

Paramters Estimation

Thus, obtaining:

$$\sum_{k=0}^{N-1} \varphi(t)y(k) = \sum_{k=0}^{N-1} \varphi(t)\varphi'(k)$$

Finally, θ^* can be derived:

$$\boxed{\theta^* = \left[\sum_{k=0}^{N-1} \varphi(t)\varphi'(t)\right]^{-1} \left[\sum_{k=0}^{N-1} \varphi(t)y(t)\right]}$$

Generate SISO data

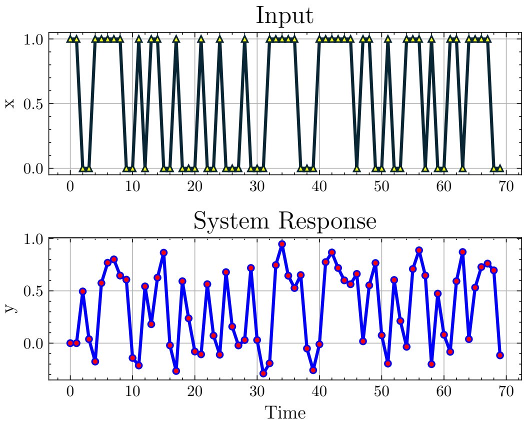
```python
def generate_siso_data(n, test_size=0.2, noise_level=0.1, a1=0.5, a2=-0.3, b1=0.7, b2=-0.2):
    """
    Generates synthetic SISO data for an ARX model with na=2 and nb=2.
    :param n: Number of data points to generate.
    :param test_size: Proportion of the dataset to include in the test split.
    :param noise_level: Standard deviation of the noise.
    :param a1, a2: Coefficients for the autoregressive part of the model.
    :param b1, b2: Coefficients for the exogenous input part of the model.
    :return: Tuple of (y_train, u_train, y_test, u_test)
    """
    u = np.random.randint(0, 2, size=n)
    y = np.zeros(n)

    for t in range(2, n):
        y[t] = a1 * y[t-1] + a2 * y[t-2] +
               b1 * u[t-1] + b2 * u[t-2] + np.random.normal(0, noise_level)

    # Splitting the data into training and testing sets
    y_train, y_test, u_train, u_test = train_test_split(y, u, test_size=test_size,
                                                        shuffle=False)
    return y_train, u_train, y_test, u_test
```

Visualize the data

```
y_train, u_train, y_test, u_test = generate_siso_data(n=1000, noise_level=0.05)
```



Fit an ARX Model

```python
def fit_arx_model(y, u, na, nb):
    """
    :param y: Target time series.
    :param u: Exogenous time series.
    :param na: Order of the autoregressive part.
    :param nb: Order of the exogenous input part.
    :return: Coefficients of A, B and the intercept of the ARX model.
    """
    max_lag = max(na, nb)
    features = []
    target = y[max_lag:]

    for i in range(max_lag, len(y)):
        lag_y = y[i-na:i] if na > 0 else []
        lag_u = u[i-nb:i] if nb > 0 else []
        row = np.concatenate([lag_y, lag_u])
        features.append(row)

    U = np.array(features)
    model = LinearRegression().fit(U, target)
    return model.coef_[:na], model.coef_[na:], model.intercept_
```

GEKKO:

GEKKO is a Python library designed for a Python package for machine learning and (dynamic) optimization.

- Among many functionalities, also provides an ARX system identification.

```
from gekko import GEKKO

m = GEKKO()    #optimization object

# system identification
t = np.arange(0, len(u_train))
na = 2 # output coefficients
nb = 2 # input coefficients
yp,p,K = m.sysid(t, u_train, y_train, na, nb, pred='meas')
```

Where y_p is the estimated reponse, p is the vector of paramters and K is the gain.

SysIdentPy: A Python for System identification

SysIdentPy is a Python library designed for linear and non-linear system identification.

- FROLS $\rightarrow$ Forward Regression Orthogonal Least Squares.
- Particularly effective in nonlinear system.
- Itertively adding new terms to the model, creating a pool of candidate terms (linear, nonlinear) based on the input data.
- The orthogonalization process multicollinearity.

SysIdentPy: A Python for System identification

In this example we test this library on our SISO system to fit an ARX fit, manually selecting the order of the model.

```
from sysidentpy.model_structure_selection import FROLS
from sysidentpy.basis_function._basis_function import Polynomial

basis_function = Polynomial(degree=1)

model = FROLS(9
    order_selection=False,
    n_terms=5,
    extended_least_squares=False,
    ylag=2,
    xlag=2,
    estimator="least_squares",
    basis_function=basis_function,
)

model.fit(X=u_train.reshape(-1,1), y=y_train.reshape(-1,1))
```

Estimated Paramters Comparison

Do you remember the coeffiecients of our ARX model?

$$\boxed{y_t = \mathbf{0.5} * y_{t-1} + \mathbf{0.3} * y_{t-2} + \mathbf{0.7} * u_{t-1} + \mathbf{-0.2} * u_{t-2}}$$

Estimated Paramters Comparison

Do you remember the coeffiecients of our ARX model?

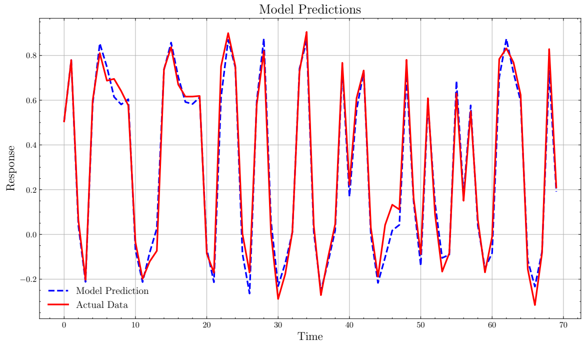
$$\boxed{y_t = \mathbf{0.5} * y_{t-1} + \mathbf{0.3} * y_{t-2} + \mathbf{0.7} * u_{t-1} + \mathbf{-0.2} * u_{t-2}}$$

	$\hat{a}_1$	$\hat{a}_2$	$\hat{b}_1$	$\hat{b}_2$	**Intercept**
Our Model	0.491774	-0.300984	0.695373	-0.194563	0.002977
GEKKO	0.491774	-0.300984	0.695373	-0.194563	0.002977
SysIdentPy	0.491774	-0.300984	0.695373	-0.194562	0.002977

Table: Estimated parameters of an ARX model using 3 different libraries

Forecast using SysIdentPy

```
yhat = model.predict(X=u_test.reshape(-1,1), y=y_test.reshape(-1,1)).squeeze()
```



MAE (Mean Absolute Error)

The MAE is the average of the absolute errors between prediction $\hat{y}_i$ and actual observations y_i. It's calculated as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

RMSE (Root Mean Square Error)

The RMSE is the square root of the average of squared differences between prediction $\hat{y}_i$ and actual observation y_i. It's calculated as:

$$\text{RMSE} = \sqrt{\tfrac{1}{n}\textstyle\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

RRMSE (Relative Root Mean Square Error)

RRMSE is the RMSE normalized against the actual value measurement $\hat{y}_i$.

$$\text{RMSE} = \sqrt{\frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(\hat{y})2}}$$

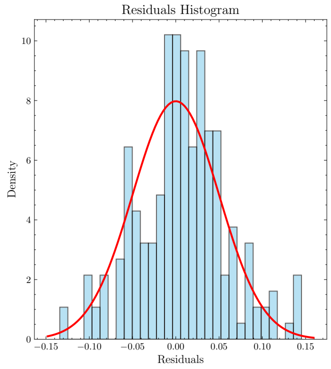
Do you rember the assumtion on $\epsilon(t)$?

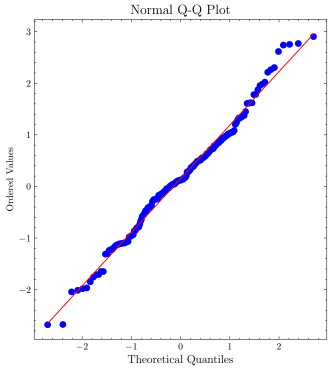
We previously assumed that the residual of the model, namely $(y - \hat{y})$ are distributed as $\epsilon(t) \sim \mathcal{N}(0, \sigma^2)$.

In our example we have introduce a distrurbance modeled as gaussian noise with $\sigma = 0.1$ If the model is good enough, we error will coincide with the gaussian noise we introduced.

Model Validation

Shapiro-Wilk test confirmed that the residuals are white noise and the Q-Q Plot below shows that the distribution of the residual is compatible with the noise introduced in our data $\sim \mathcal{N}(0, \sigma^2 = 0.1)$.





Model Selection: *AIC* & *BIC*

AIC and BIC are fundamental statistical tools used to select the best model from a group of potential models based on their performance with a given dataset. They are particularly crucial when there are multiple models, as they help balance fitting the data and model complexity, thus preventing overfitting or underfitting.

Model Selection: *AIC* & *BIC*

Akaike Information Criterion (AIC)

Definition:

$$\boxed{\text{AIC} = 2k - 2\ln(L)}$$

where:

- **k** is the number of parameters
- **L** is the maximum likelihood of the model.

Bayesian Information Criterion (BIC)

Definition:

$$\boxed{\text{BIC} = \ln(n)k - 2\ln(L)}$$

where:

- **n** is the number of observations
- **k** is the number of parameters
- **L** is the maximum likelihood of the model.

Model Selection: *AIC* & *BIC*

Akaike Information Criterion (AIC)

Definition:

- Balances model fit and complexity.
- More emphasis on goodness of fit than on simplicity.

Limitations:

- Relative measure; cannot be used to test a single model.
- Can be biased in small samples.

Bayesian Information Criterion (BIC)

Definition:

- Includes a penalty term for the number of observations
- making it more reliable for larger datasets.
- Favors simpler models compared to AIC.
- Assumes that the model errors are independently and identically distributed.