# Advanced Fault Detection in Condition Monitoring: Combining Model-Based and Data-Driven Approaches

**Part 1**


***Silvio Simani***

***URL:*** **http://www.silviosimani.it/talks.html**

***E-mail:*** **silvio.simani@unife.it**

# Lecture Main Topics

- General introduction
  - State-of-the-art review
  - Fault diagnosis nomenclature
- Main methods for fault diagnosis
  - **Parameter estimation methods**
  - Observer and filter approaches
  - Parity relations
  - **Neural networks and fuzzy systems**
- Application examples
- Concluding remarks

# Programme Details

- Introduction: Course Introduction
- Issues in Model-Based Fault Diagnosis
- Fault Detection and Isolation (FDI) Methods based on Analytical Redundancy
- Model-based Fault Detection Methods
- The Robustness Problem in Fault Detection
- Fault Identification Methods
- Modelling of Faulty Systems

# Programme Details (Cont'd)

- Residual Generation Techniques
- The Residual Generation Problem

- Fault Diagnosis Technique Integration
- Fuzzy Logic for Residual Generation
- Neural Networks in Fault Diagnosis

# Programme Details (Cont'd)

- Observers for Robust Residual Generation
- Residual Robustness to Disturbances
- Application Examples

# Annual Meetings on FDI

- IFAC SAFEPROCESS Symposium
- *Symposium on Fault Detection Supervision & Safety for Technical Processes*
  - 1st held in Baden–Baden, Germany in 1991
  - 2nd in Espo, Finland in 1994
  - 3rd at Hull, UK in 1997
  - 4th held in Budapest, Hungary in 2000
  - 5th at Washington DC, USA, July 2003
  - 6th in Beijing, P.R. China, August 2006
  - 7th in Barcelona, Spain, July 2009
  - 8th in Mexico City, Mexico, August 2012
  - 9th in Paris, France, 2015
  - 10th in Warsaw, Poland, 2018
  - 11th in Cyrpus, Greece, 2022
  - 12th in Ferrara, Italy, 2024
  - 13th in Delft, The Netherlands, scheduled in 2027

# Nomenclature

1. *States and Signals*

**Fault**
An unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable, usual or standard condition.

**Failure**
A permanent interruption of a system's ability to perform a required function under specified operating conditions.

**Malfunction**
An intermittent irregularity in the fulfilment of a system's desired function.

**Error**
A deviation between a measured or computed value of an output variable and its true or theoretically correct one.

# Nomenclature (cont'd)

1. *States and Signals*

**Disturbance**
An unknown and uncontrolled input acting on a system.

**Residual**
A fault indicator, based on a deviation between measurements and model-equation-based computations.

**Symptom**
A change of an observable quantity from normal behaviour.

# Nomenclature (Cont'd)

2. *Functions*

**Fault detection**
Determination of faults present in a system and the time of detection.

**Fault isolation**
Determination of the kind, location and time of detection of a fault. Follows fault detection.

**Fault identification**
Determination of the size and time-variant behaviour of a fault. Follows fault isolation.

**Fault diagnosis**
Determination of the kind, size, location and time of detection of a fault. Follows fault detection. Includes fault detection and identification.

**Monitoring**
A continuous real-time task of determining the conditions of a physical system, by recording information, recognising and indication anomalies in the behaviour.

**Supervision**
Monitoring a physical and taking appropriate actions to maintain the operation in the case of fault.

# Nomenclature (Cont'd)

3. *Models*

**Quantitative model**
Use of static and dynamic relations among system variables and parameters in order to describe a system's behaviour in quantitative mathematical terms.

**Qualitative model**
Use of static and dynamic relations among system variables in order to describe a system's behaviour in qualitative terms such as causalities and IF–THEN rules.

**Diagnostic model**
A set of static or dynamic relations which link specific input variables, *the symptoms*, to specific output variables, the faults.

**Analytical redundancy**
Use of more (not necessarily identical) ways to determine a variable, where one way uses a mathematical process model in analytical form.

# Nomenclature (Cont'd)

**Reliability**
Ability of a system to perform a required function under stated conditions, within a given scope, during a given period of time.

**Safety**
Ability of a system not to cause danger to persons or equipment or the environment.

**Availability**
Probability that a system or equipment will operate satisfactorily and effectively at any point of time.

# Nomenclature (Cont'd)

5. *Time dependency of faults*

**Abrupt fault**
Fault modelled as stepwise function. It represents bias in the monitored signal.

**Incipient fault**
Fault modelled by using ramp signals. It represents drift of the monitored signal.

**Intermittent fault**
Combination of impulses with different amplitudes.

*NOTE: Incipient fault (slowly developing fault)= hard to detect !!!*

# Nomenclature (Cont'd)

6. *Fault terminology*

**Additive fault**

Influences a variable by an addition of the fault itself. They may represent, *e.g.*, offsets of sensors.

**Multiplicative fault**

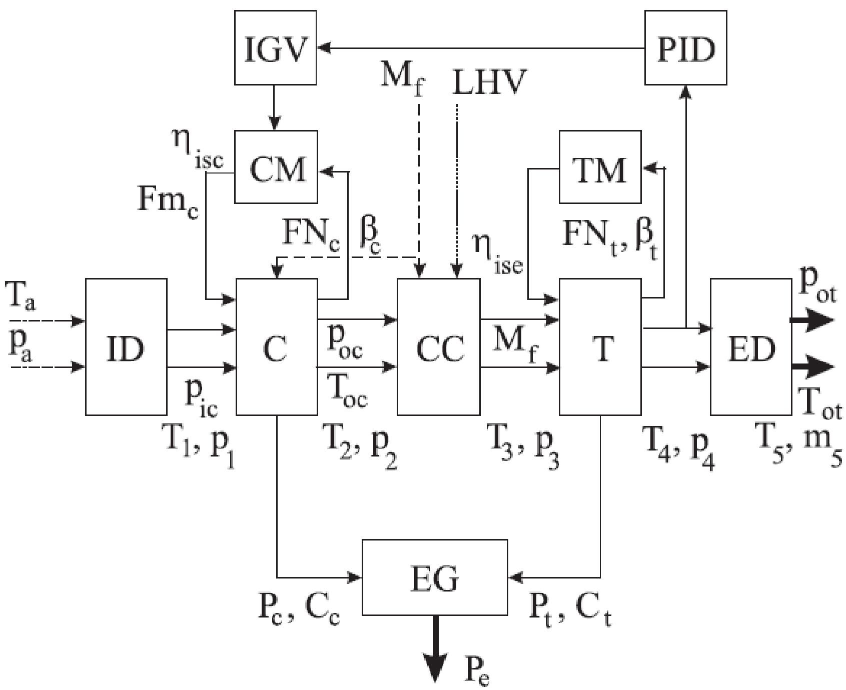
Are represented by the product of a variable with the fault itself. They can appear as parameter changes within a process.

# Application Examples

- Simulated case studies
- Identification/FDI applications
- Real processes
- Research works
- Undergraduate theses topics

# Simulated Application Examples

## Simulated Gas Turbine




| | |
|---|---|
| C | Compressor |
| CC | Combustor (Combustion Chamber) |
| CM | Compressor Map |
| ED | Exhaust Duct |
| EG | Electric Generator |
| ID | Intake Duct |
| IGV | Inlet Guide Vanes |
| PID | Proportional Integral Derivative Controller |
| T | Turbine |
| TM | Turbine Map |

Figure 5.2: Block diagram of the single–shaft gas turbine.

# Simulated Application Examples (Cont'd)

## Simulated Gas Turbine

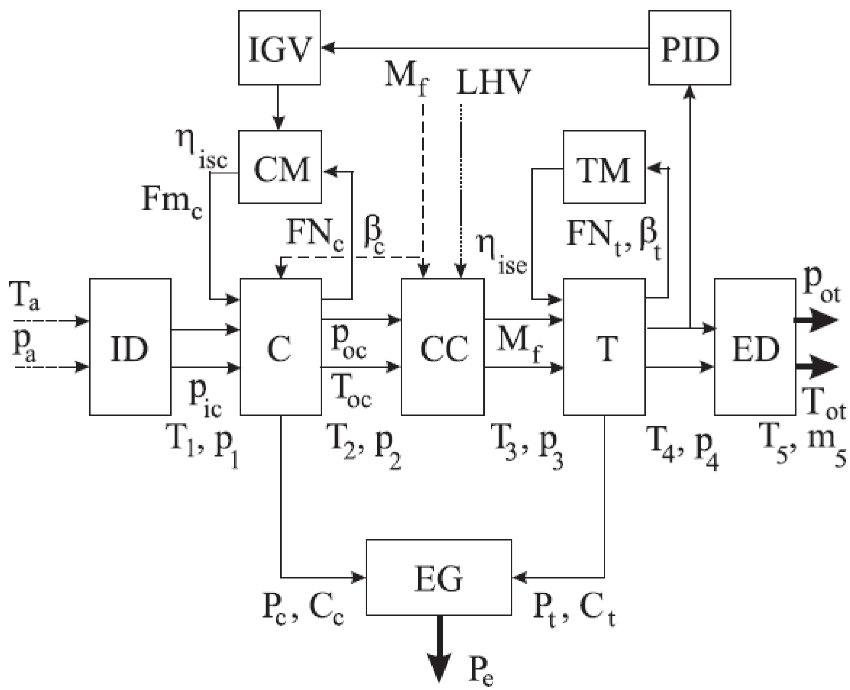
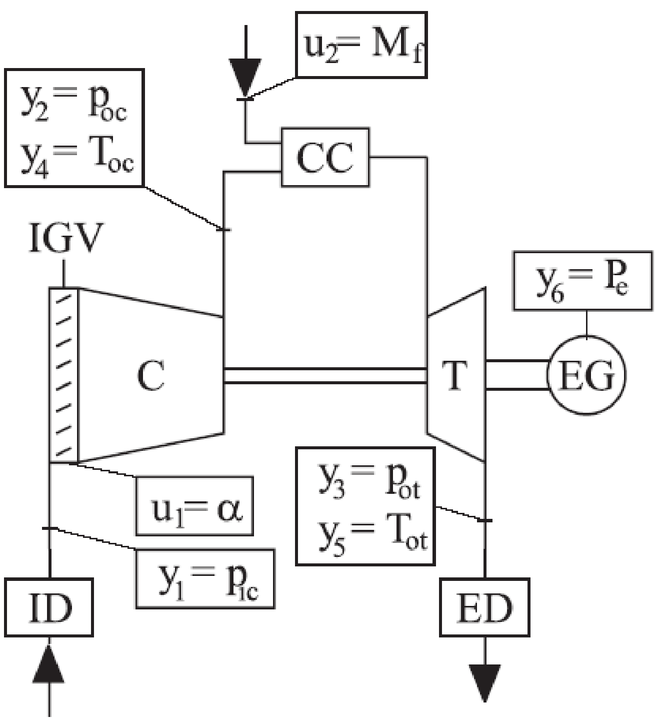



Figure 5.2: Block diagram of the single–shaft gas turbine.

| | |
|---|---|
| $M_f$ | Fuel mass flow rate |
| $LHV$ | Lower Heating Value |
| $\eta_{isc}$ | Isentropic compressor efficiency |
| $Fm_c$ | Compressor mass flow function |
| $FN_c$ | Compressor rotational speed function |
| $\beta_c$ | Compressor pressure ratio |
| $\eta_{ise}$ | Isentropic expansion efficiency |
| $FN_t$ | Turbine rotational speed function |
| $\beta_t$ | Turbine pressure ratio |
| $T_i$ | $i$–th section (module) temperature ($i = 1, \cdots, 5$) |
| $p_i$ | $i$–th section (module) pressure ($i = 1, \cdots, 5$) |
| $m_5$ | 5–th module mass flow rate |
| $T_a$ | Ambient temperature |
| $p_a$ | Ambient pressure |
| $P_c$ | Compressor power |
| $P_t$ | Turbine power |
| $C_c$ | Compressor torque |
| $C_t$ | Turbine torque |
| $P_e$ | Electrical power |

# Simulated Application Examples (Cont'd)

## Simulated Gas Turbine




$u_1(t)$, Inlet Guide Vane (IGV) angular position ($\alpha$);

$u_2(t)$, fuel mass flow rate ($M_f$).

$y_1(t)$, pressure at the compressor inlet ($p_{ic}$);

$y_2(t)$, pressure at the compressor outlet ($p_{oc}$);

$y_3(t)$, pressure at the turbine outlet ($p_{ot}$);

$y_4(t)$, temperature at the compressor outlet ($T_{oc}$);

$y_5(t)$, temperature at the turbine outlet ($T_{ot}$);

$y_6(t)$, electrical power at the generator terminal ($P_e$).

# Simulated Application Examples (Cont'd)

## Simulated Gas Turbine (SIMULINK®)

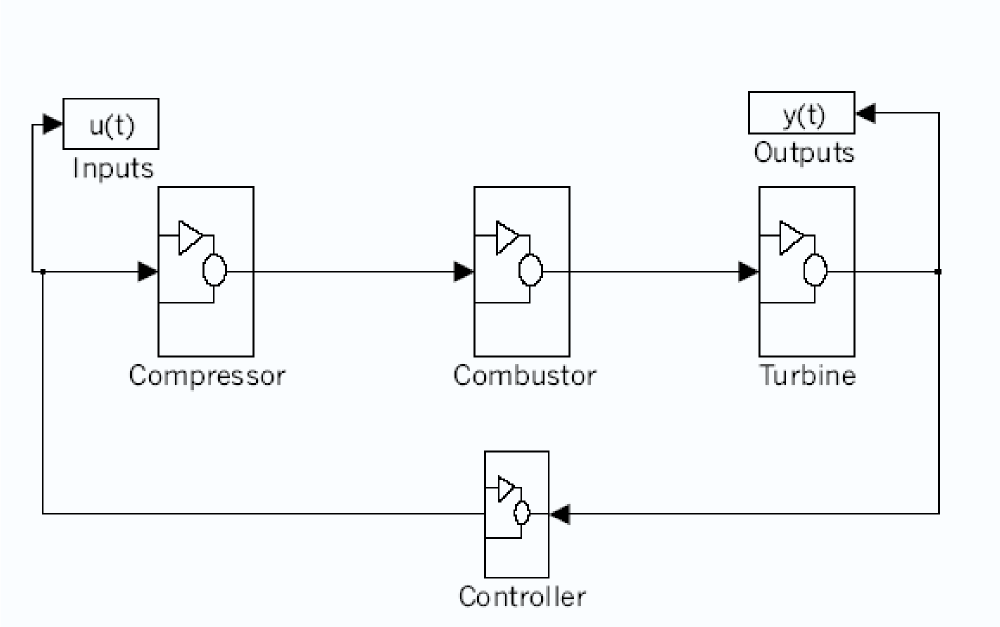


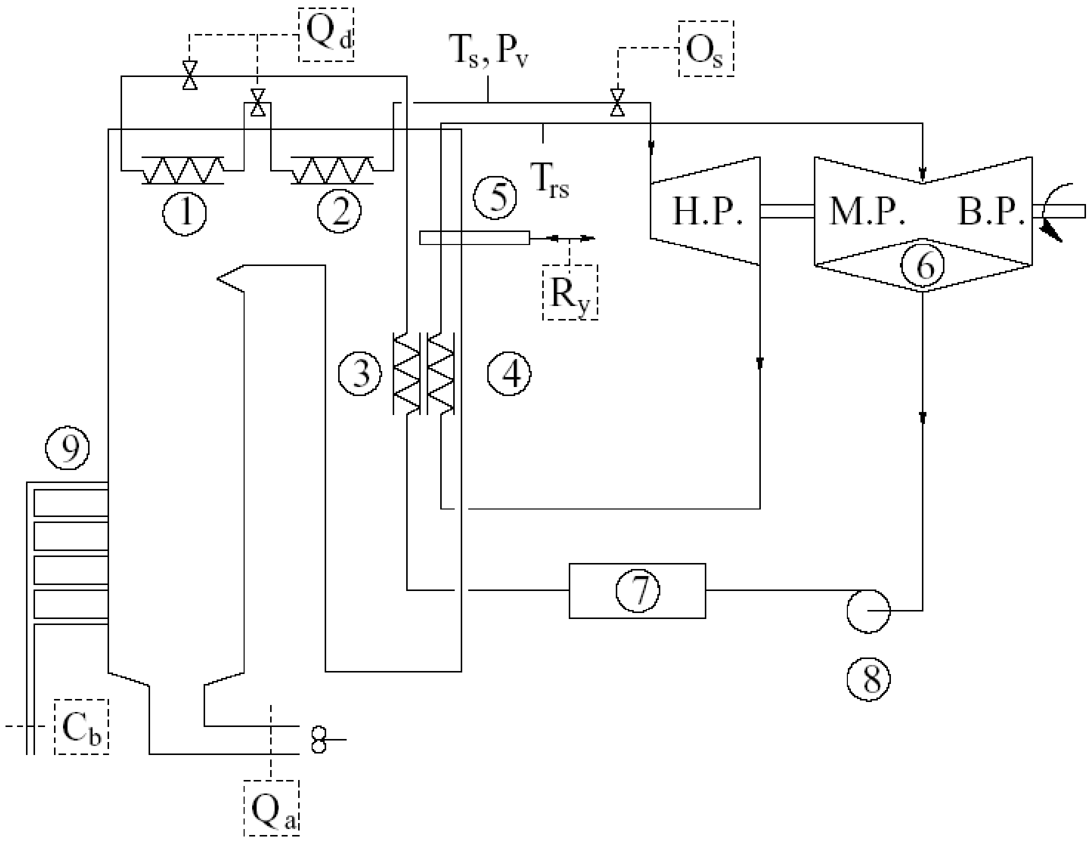

Figure 5.7: SIMULINK block diagram of the process.

Gas turbine main cycle parameters (ISO design conditions).

| | |
|---|---|
| Air mass flow rate [kg/s] | 24.4 |
| Cycle pressure ratio $(P_{oc}/P_{ic})$ | 9.1 |
| Electrical power $(P_e)$ [kW] | 5220 |
| Exhaust temperature $(T_{ot})$[K] | 796 |
| Fuel mass flow rate $(M_f)$ [kg/s] | 0.388 |
| IGV angle range $(\Delta\alpha)$ [deg] | 17 |

# Simulated Application Examples (Cont'd)

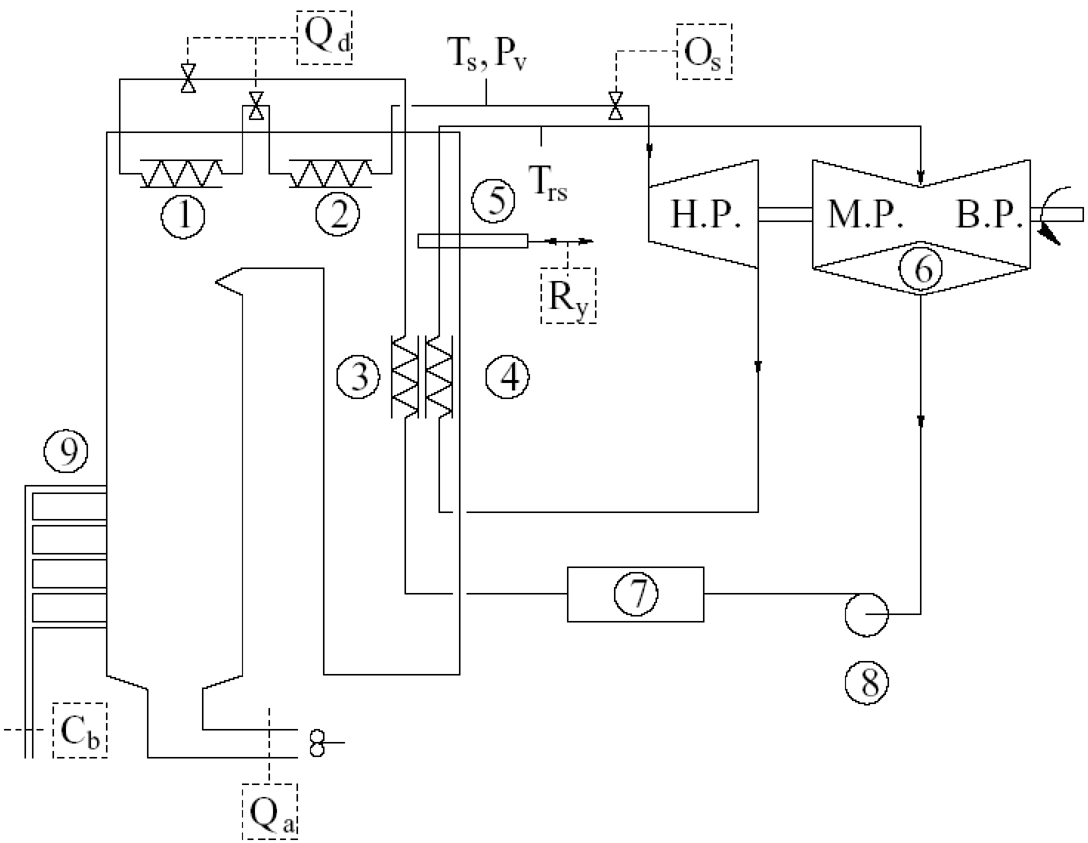
## Simulated Power Plant: *Pont sur Sambre*




1. super heater (radiation);
2. super heater (convection);
3. super heater;
4. reheater;
5. dampers;
6. condenser;
7. drum;
8. water pump;
9. burner.

# Simulated Application Examples (Cont'd)

## Simulated Power Plant: *Pont sur Sambre*




| | | |
|---|---|---|
| $u_1(t)$ | $C_b$ | gas flow |
| $u_2(t)$ | $O_s$ | turbine valves opening |
| $u_3(t)$ | $Q_d$ | super heater spray flow |
| $u_4(t)$ | $R_y$ | gas dampers |
| $u_4(t)$ | $Q_a$ | air flow |
| $y_1(t)$ | $P_v$ | steam pressure |
| $y_2(t)$ | $T_s$ | main steam temperature |
| $y_i(t)$ | $T_{rs}$ | reheat steam temperature |

# Simulated Application Examples (Cont'd)

## Simulated Gas Turbine

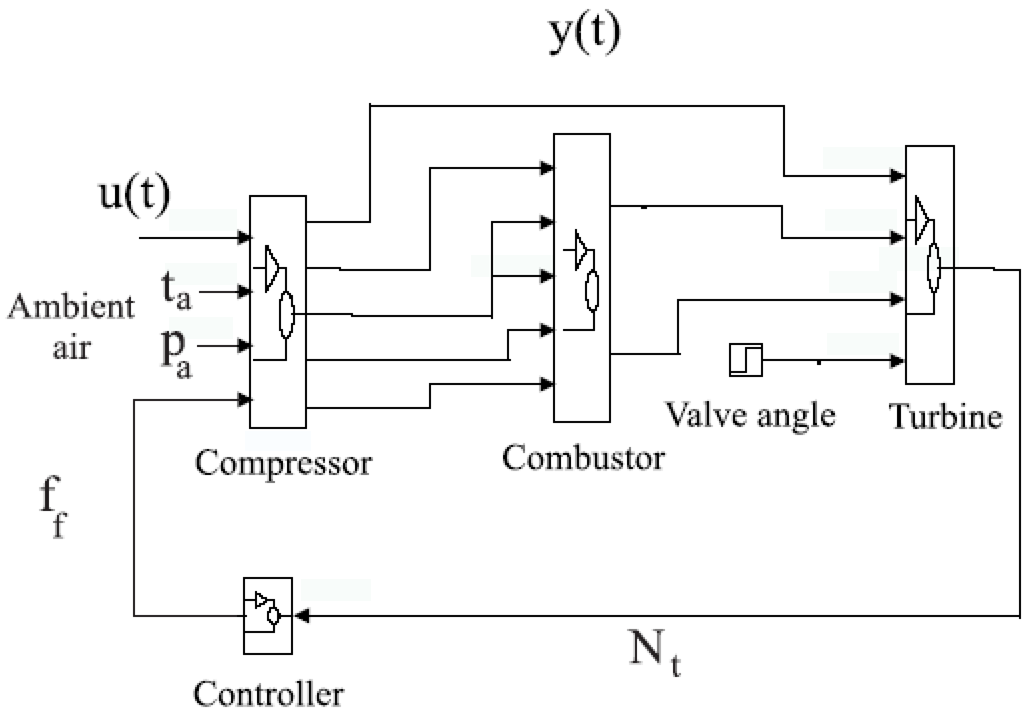



Figure 5.44: The monitored system.

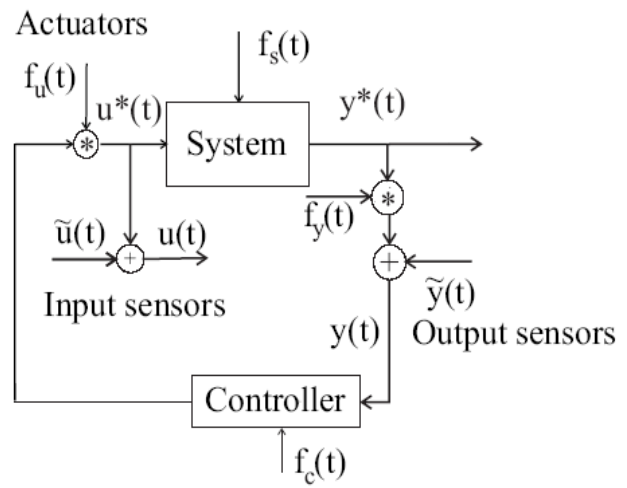



Figure 5.46: Turbine closed–loop scheme.

# Simulated Application Examples (Cont'd)

## Small Aircraft Model

Piper Malibu

Table 1: Nomenclature

| | | | |
|---|---|---|---|
| $V$ | True Air Speed (TAS) | $H$ | altitude |
| $\alpha$ | angle of attack | $\delta_e$ | elevator deflection angle |
| $\beta$ | angle of sideslip | $\delta_a$ | aileron deflection angle |
| $P$ | roll rate | $\delta_r$ | rudder deflection angle |
| $Q$ | pitch rate | $\delta_{th}$ | throttle aperture percentage |
| $R$ | yaw rate | $\gamma$ | flight path angle |
| $\phi$ | bank angle | $g$ | acceleration of gravity |
| $\theta$ | elevation angle | $m$ | airplane mass |
| $\psi$ | heading angle | $I_x$, $I_y$, $I_z$ | principal-axis inertia moments |
| $n$ | engine shaft angular rate | $d_t$ | distance of c.g. from the Thrust line |

# Simulated Application Examples (Cont'd)

## Small Aircraft Model


Table 1: Nomenclature

| | | | |
|---|---|---|---|
| $V$ | True Air Speed (TAS) | $H$ | altitude |
| $\alpha$ | angle of attack | $\delta_e$ | elevator deflection angle |
| $\beta$ | angle of sideslip | $\delta_a$ | aileron deflection angle |
| $P$ | roll rate | $\delta_r$ | rudder deflection angle |
| $Q$ | pitch rate | $\delta_{th}$ | throttle aperture percentage |
| $R$ | yaw rate | $\gamma$ | flight path angle |
| $\phi$ | bank angle | $g$ | acceleration of gravity |
| $\theta$ | elevation angle | $m$ | airplane mass |
| $\psi$ | heading angle | $I_x$, $I_y$, $I_z$ | principal-axis inertia moments |
| $n$ | engine shaft angular rate | $d_t$ | distance of c.g. from the Thrust line |

# Simulated Application Examples (Cont'd)

6.1 SATELLITE OVERVIEW

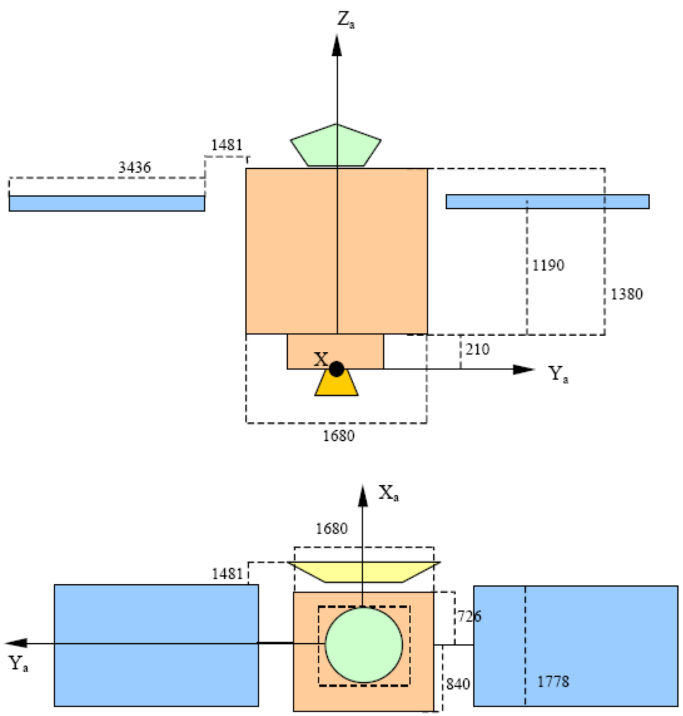



Figure 6-1: Satellite dimensions

6.4 SOLAR ARRAYS DYNAMICS

A GLOBALSTAR solar array has been selected. Its dimensions are recalled on the Figure 6-2.

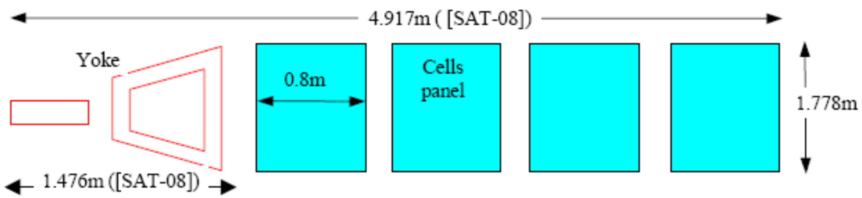



*Figure 6-2 : MARS-EXPRESS solar array*

Simple SA model

**Aerospace Satellite**

# Simulated Application Examples (Cont'd)

## Aerospace Satellite

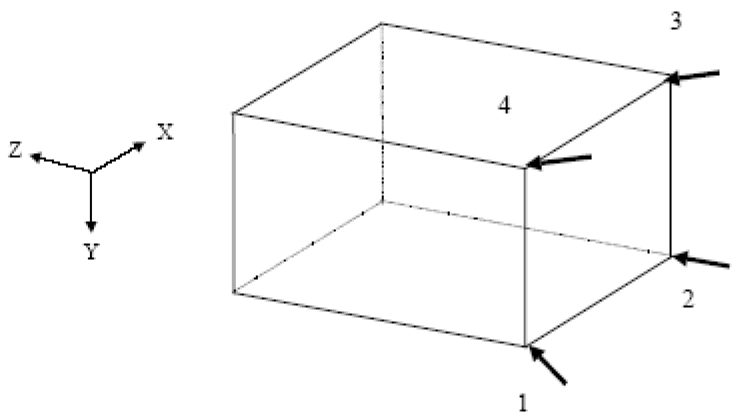



*Figure 6-3 : Thrusters implementation*

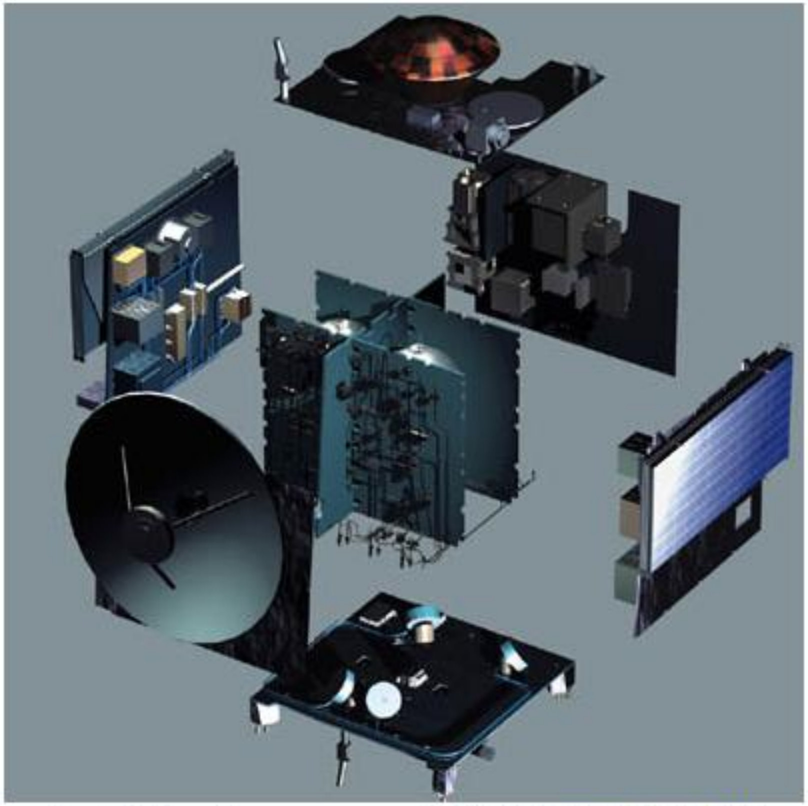
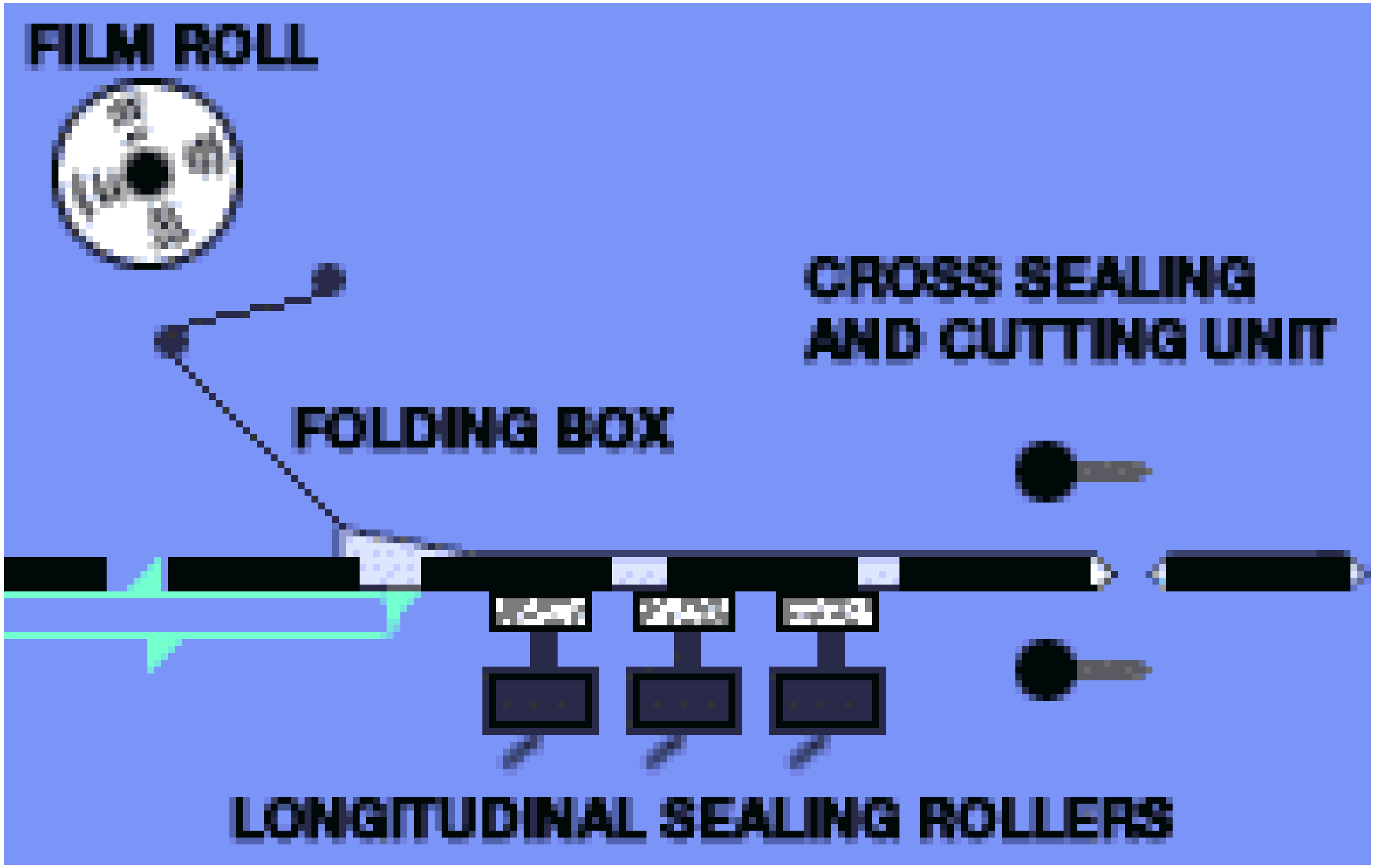


Fig. 1: Mars Express spacecraft (MEX) (www.esa.int)

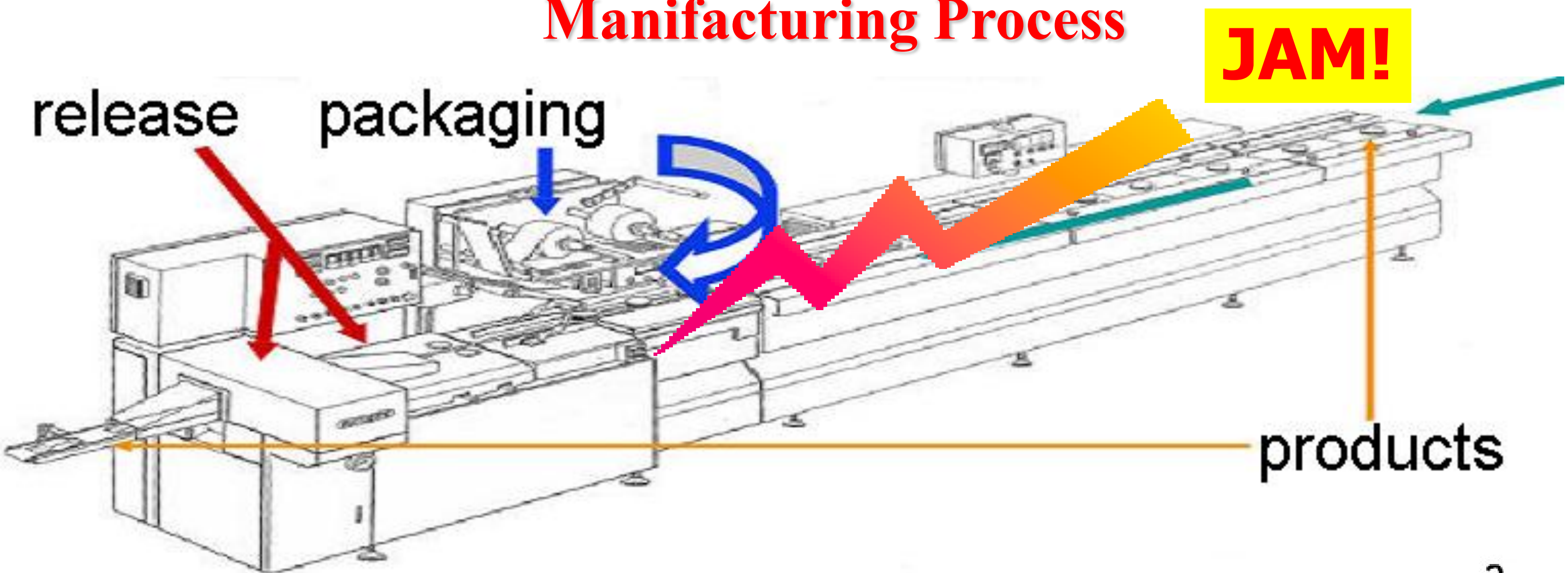
# Simulated Application Examples (Cont'd)

**Manifacturing Process**



# Simulated Application Examples (Cont'd)

**Manifacturing Process**




**(possibly unskilled)**
**human operator**

# Introduction

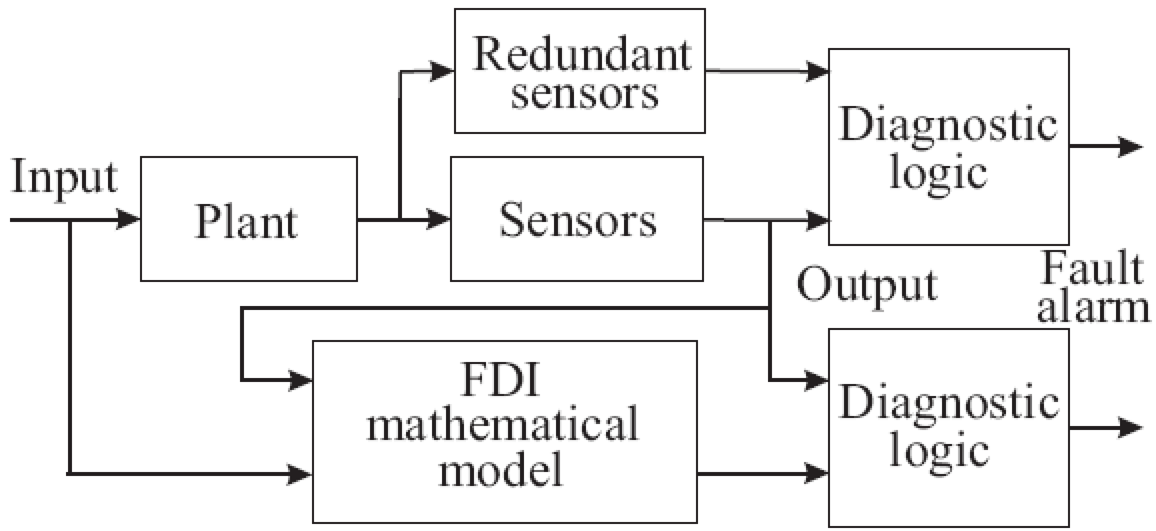



Figure 1.1: Comparison between hardware and analytical redundancy schemes.

# Introduction (Cont'd)

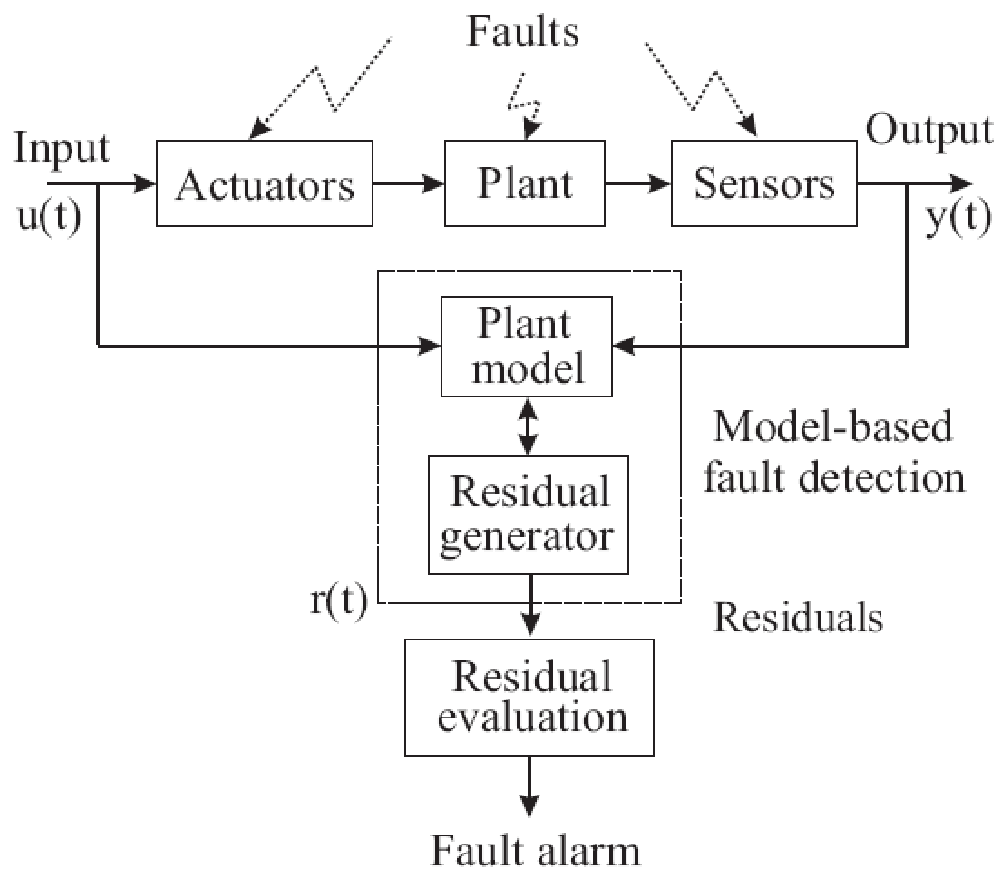



Figure 1.2: Scheme for the model–based fault detection.

# Residual Generation

- This block generates residual signals using available inputs and outputs from the monitored system
- This residual (or fault symptom) should indicate that a fault has occurred
- Normally zero or close to zero under no fault condition, whilst distinguishably different from zero when a fault occurs

# Residual Evaluation

- This block examines residuals for the likelihood of faults and a decision rule is then applied to determine if any faults have occurred
- It may perform a simple threshold test (geometrical methods) on the instantaneous values or moving averages of the residuals
- It may consist of statistical methods, e.g., generalised likelihood ratio testing or sequential probability ratio testing

# Introduction (Cont'd)

- Model-Based FDI Methods:

1. Output observers (OO, estimators, filters);
2. Parity equations;
3. Identification and parameter estimation.

# Introduction (Cont'd)

- Signal Model-Based Methods:

  1. Bandpass filters;
  2. Spectral analysis (FFT);
  3. Maximum-entropy estimation.

- Change Detection: Residual Analysis

  1. Mean and variance estimation;
  2. Likelihood-ratio test, Bayes decision;
  3. Run-sum test.

# Introduction (Cont’d)

- Model Uncertainty in Fault Detection & Isolation (FDI)
  - Model-reality mismatch
  - Sensitivity problem: incipient faults!
- Robustness in FDI
  - Disturbance, modelling errors, uncertainty
  - Unknown Input Observer (UIO) and Kalman filter: robust residual generation
- System Identification for FDI
  - Estimation of a reliable model
  - Modelling accuracy & disturbance estimation

# Introduction (Cont'd)

- Fault Identification Methods
  - Fault nature (type, shape) & size (amplitude)

    1. Geometrical distance and probabilistic methods;
    2. Artificial neural networks;
    3. Fuzzy clustering.

- Approximate Reasoning Methods:

    1. Probabilistic reasoning;
    2. Possibilistic reasoning with fuzzy logic;
    3. Reasoning with artificial neural networks.

# Introduction (Cont'd)

- FDI applications status & review

Table 1.1: FDI applications and number of contributions.

| Application | Number of contributions |
|---|---|
| Simulation of real processes | 55 |
| Large-scale pilot processes | 44 |
| Small-scale laboratory processes | 18 |
| Full-scale industrial processes | 48 |

Table 1.2: Fault type and number of contributions.

| Fault type | Number of contributions |
|---|---|
| Sensor faults | 69 |
| Actuator faults | 51 |
| Process faults | 83 |
| Control loop or controller faults | 8 |

# Introduction (Cont'd)

- FDI applications status & review

Table 1.3: FDI methods and number of contributions.

| Method type | Number of contributions |
|---|---|
| Observer | 53 |
| Parity space | 14 |
| Parameter estimation | 51 |
| Frequency spectral analysis | 7 |
| Neural networks | 9 |

Table 1.4: Residual evaluation methods and number of contributions.

| Evaluation method | Number of contributions |
|---|---|
| Neural networks | 19 |
| Fuzzy logic | 5 |
| Bayes classification | 4 |
| Hypothesis testing | 8 |

# Introduction (Cont'd)

- FDI applications status & review

Table 1.5: Reasoning strategies and number of contributions.

| Reasoning strategy | Number of contributions |
|---|---|
| Rule based | 10 |
| Sign directed graph | 3 |
| Fault symptom tree | 2 |
| Fuzzy logic | 6 |

Table 1.6: Applications of model–based fault detection.

| FDD | Number of contributions |
|---|---|
| Milling and grinding processes | 41 |
| Power plants and thermal processes | 46 |
| Fluid dynamic processes | 17 |
| Combustion engine and turbines | 36 |
| Automotive | 8 |
| Inverted pendulum | 33 |
| Miscellaneous | 42 |
| DC motors | 61 |
| Stirred tank reactor | 27 |
| Navigation system | 25 |
| Nuclear process | 10 |

# Model-based FDI Techniques

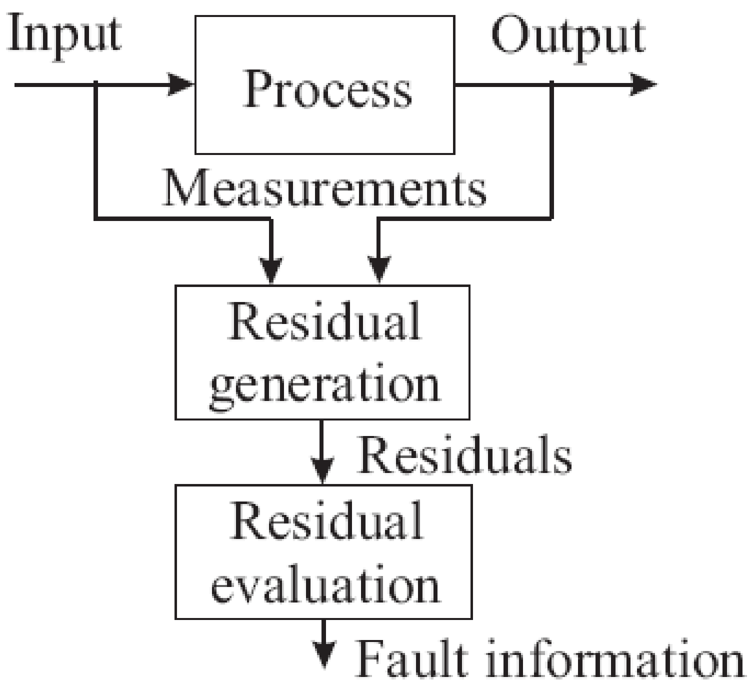



Figure 2.1: Structure of model-based FDI system.

# Model-based FDI Techniques (Cont'd)

1. **Residual generation:** this block generates residual signals using available inputs and outputs from the monitored system. This residual (or fault symptom) should indicate that a fault has occurred. It should normally be zero or close to zero under no fault condition, whilst distinguishably different from zero when a fault occurs. This means that the residual is characteristically independent of process inputs and outputs, in ideal conditions. Referring to Figure 2.1, this block is called *residual generation*.

2. **Residual evaluation:** This block examines residuals for the likelihood of faults and a decision rule is then applied to determine if any faults have occurred. The *residual evaluation* block, shown in Figure 2.1, may perform a simple threshold test (geometrical methods) on the instantaneous values or moving averages of the residuals. On the other hand, it may consist of statistical methods, *e.g.*, generalised likelihood ratio testing or sequential probability ratio testing [Isermann, 1997, Willsky, 1976, Basseville, 1988, Patton et al., 2000].

# Model-based FDI Techniques (Cont'd)

- Modelling of Faulty Systems

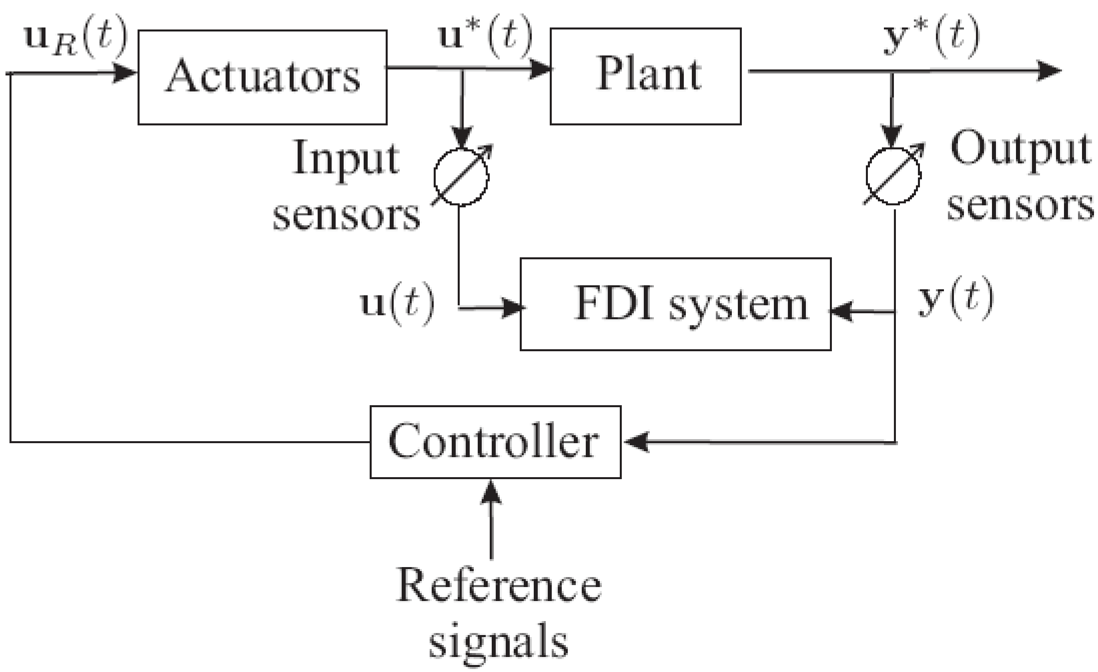



Figure 2.2: Fault diagnosis in a closed-loop system.

# Model-based FDI Techniques (Cont'd)

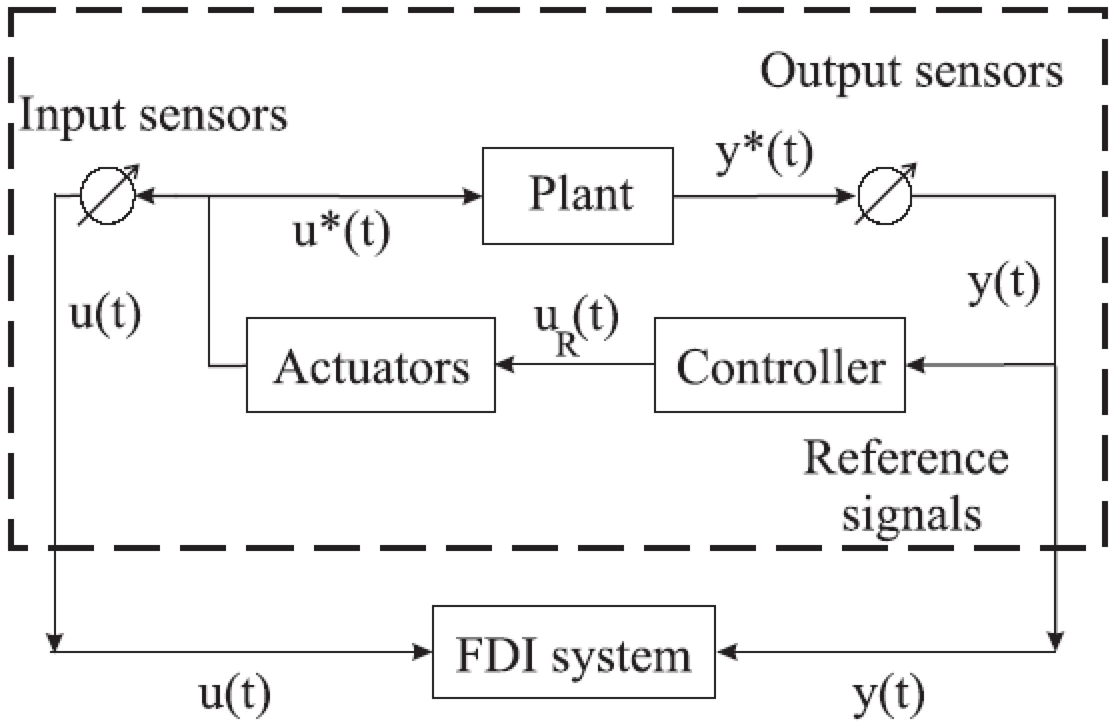



Figure 2.3: The rearranged fault diagnosis scheme.

# Model-based FDI Techniques (Cont'd)

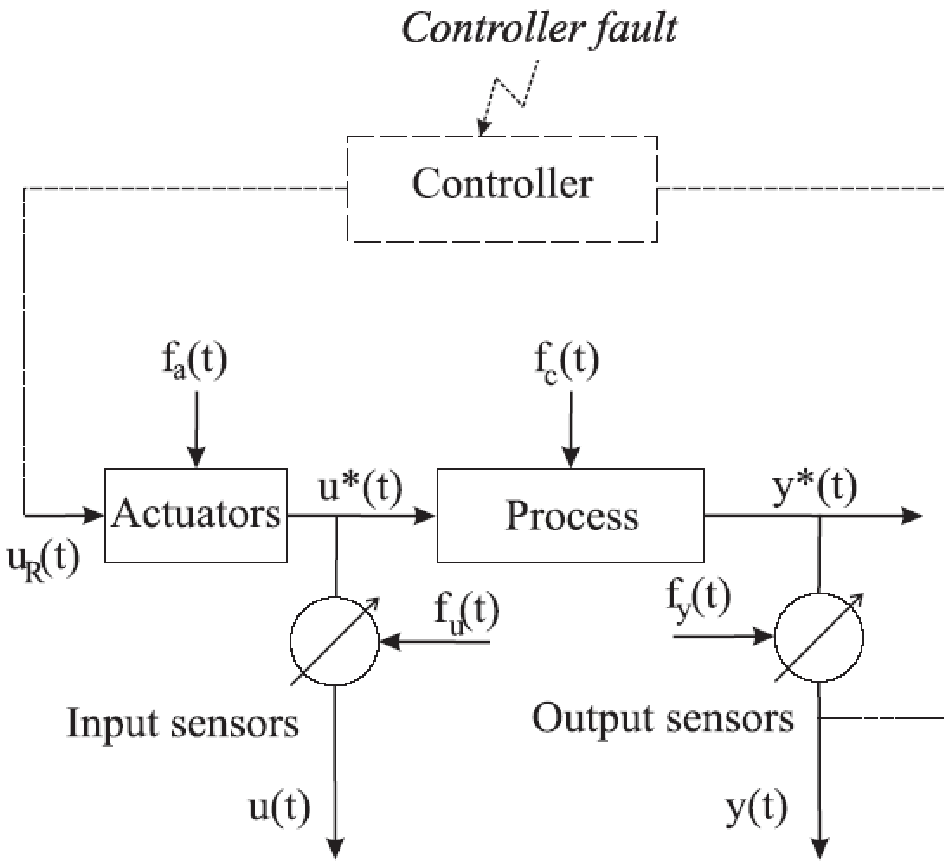



Figure 2.4: The controlled system and fault topology.

# Model-based FDI Techniques (Cont'd)

**Fault Location:**

- Actuators
- Process or system components
- Input sensors
- Output sensors
- Controllers

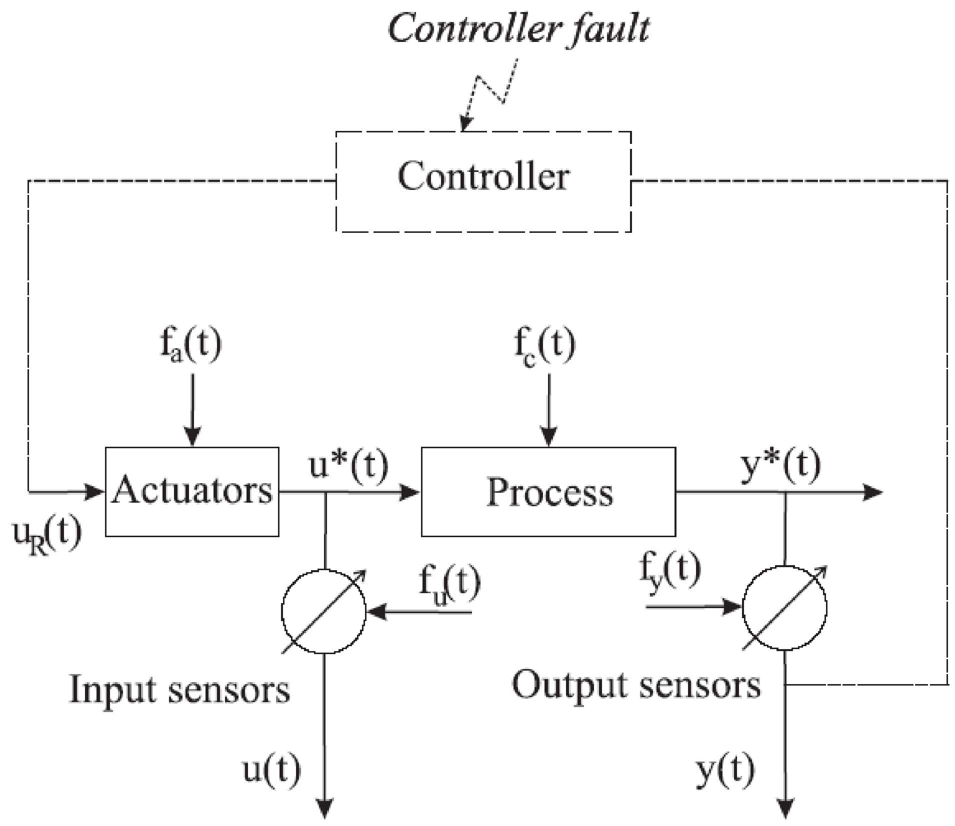



Figure 2.4: The controlled system and fault topology.

# Model-based FDI Techniques (Cont'd)

## Fault and System Modelling

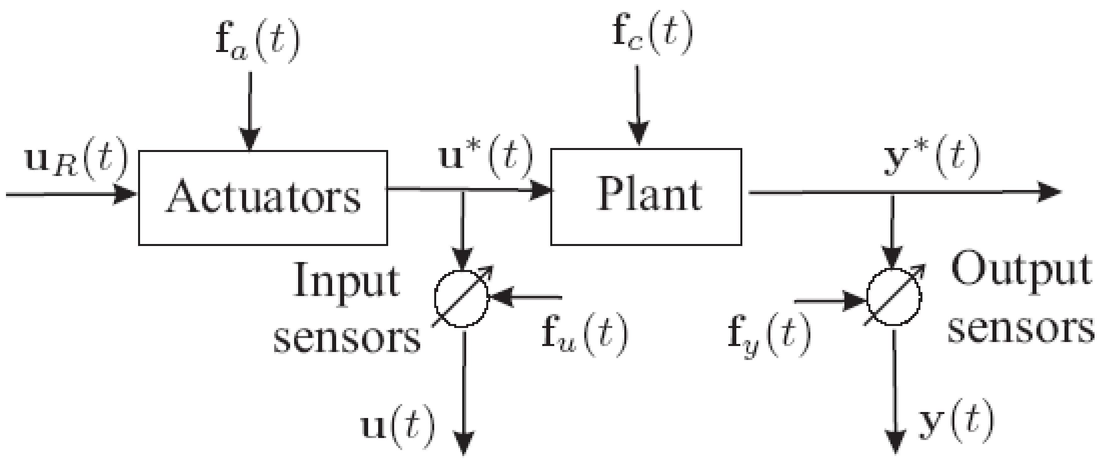



Figure 2.5: The monitored system and fault topology.

$$\begin{cases} \mathbf{x}(t+1) & = & \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}^*(t) \\ \mathbf{y}^*(t) & = & \mathbf{C}\mathbf{x}(t) \end{cases}$$

# Model-based FDI Techniques (Cont'd)

## Fault and System Modelling

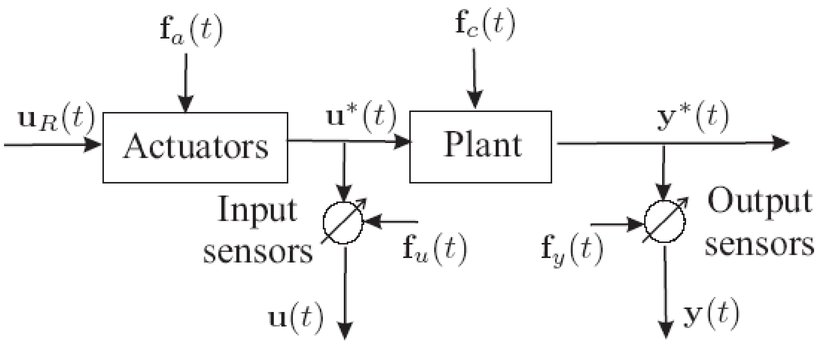



Figure 2.5: The monitored system and fault topology.

$$\begin{cases} \mathbf{x}(t+1) & = & \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}^*(t) \\ \mathbf{y}^*(t) & = & \mathbf{C}\mathbf{x}(t) \end{cases}$$

$$\mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}^*(t) + \mathbf{f}_c(t)$$

$$\mathbf{f}_c(t) = I_i \Delta a_{ij} x_j(t)$$

$$\begin{cases} \mathbf{u}(t) & = & \mathbf{u}^*(t) + \mathbf{f}_u(t) \\ \mathbf{y}(t) & = & \mathbf{y}^*(t) + \mathbf{f}_y(t) \end{cases}$$

# Model-based FDI Techniques (Cont'd)

## Fault and System Modelling

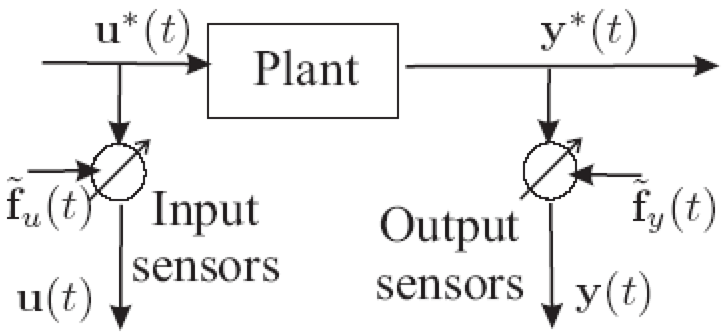


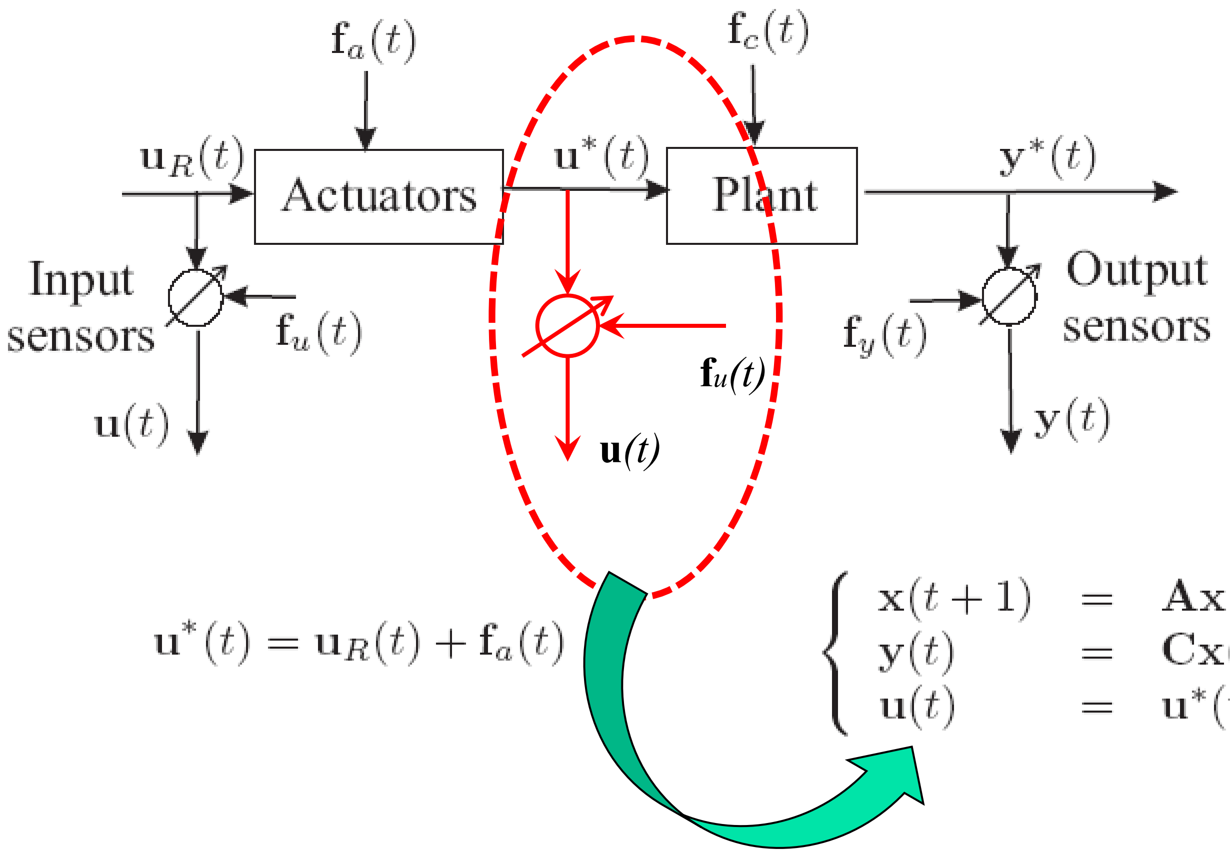

Figure 2.6: The structure of the plant sensors.

$$\begin{cases} \mathbf{u}(t) = \mathbf{u}^*(t) + \tilde{\mathbf{u}}(t) \\ \mathbf{y}(t) = \mathbf{y}^*(t) + \tilde{\mathbf{y}}(t) \end{cases}$$

$$\begin{cases} \mathbf{u}(t) = \mathbf{u}^*(t) + \tilde{\mathbf{u}}(t) + \mathbf{f}_u(t) \\ \mathbf{y}(t) = \mathbf{y}^*(t) + \tilde{\mathbf{y}}(t) + \mathbf{f}_y(t) \end{cases}$$

# Model-based FDI Techniques (Cont'd)

## Fault and System Modelling




$$\mathbf{u}^*(t) = \mathbf{u}_R(t) + \mathbf{f}_a(t)$$

$$\begin{cases} \mathbf{x}(t+1) & = & \mathbf{A}\mathbf{x}(t) + \mathbf{f}_c(t) + B\mathbf{u}^*(t) \\ \mathbf{y}(t) & = & \mathbf{C}\mathbf{x}(t) + \mathbf{f}_y(t) \\ \mathbf{u}(t) & = & \mathbf{u}^*(t) + \mathbf{f}_u(t) \end{cases}$$

# Model-based FDI Techniques (Cont'd)

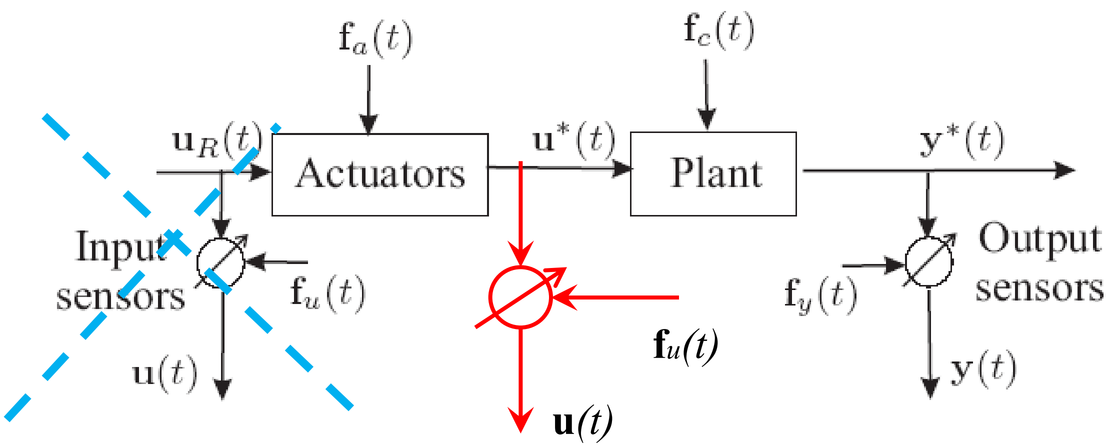



Figure 2.7: Fault topology with actuator input signal measurement.

- Modelling of Faulty Systems

$$f(t) = [f_a^T,\ f_u^T,\ f_c^T,\ f_y^T]^T \in \Re^k$$

$$\begin{cases} x(t+1) = Ax(t) + f_c(t) + Bf_a(t) + Bu^*(t) \\ y(t) = Cx(t) + f_y(t) \\ u(t) = u^*(t) + f_u(t) \end{cases}$$

$$\begin{cases} x(t+1) = Ax(t) + Bu^*(t) + L_1 f(t) \\ y(t) = Cx(t) + L_2 f(t) \\ u(t) = u^*(t) + L_3 f(t) \end{cases}$$

# Model-based FDI Techniques (Cont'd)

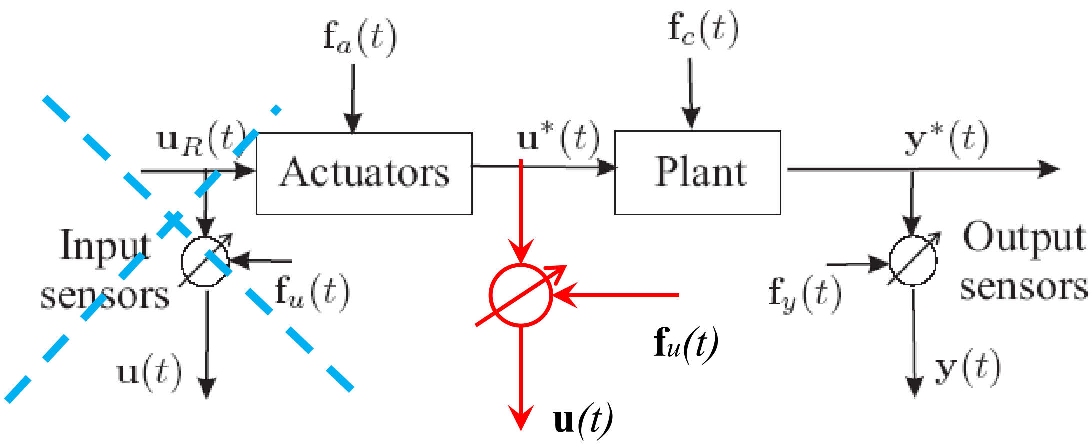



Figure 2.7: Fault topology with actuator input signal measurement.

- Modelling of Faulty Systems

$$\mathbf{y}(z) = \mathbf{G}_{yu^*}(z)\mathbf{u}^*(z) + \mathbf{G}_{yf}(z)\mathbf{f}(z)$$

- Transfer function description:

$$\begin{cases} \mathbf{G}_{yu^*}(z) & = & \mathbf{C}(zI - \mathbf{A})^{-1}\mathbf{B} \\ \mathbf{G}_{yf}(z) & = & \mathbf{C}(zI - \mathbf{A})^{-1}\mathbf{L}_1 + \mathbf{L}_2 \end{cases}$$

# Residual Generator Structure

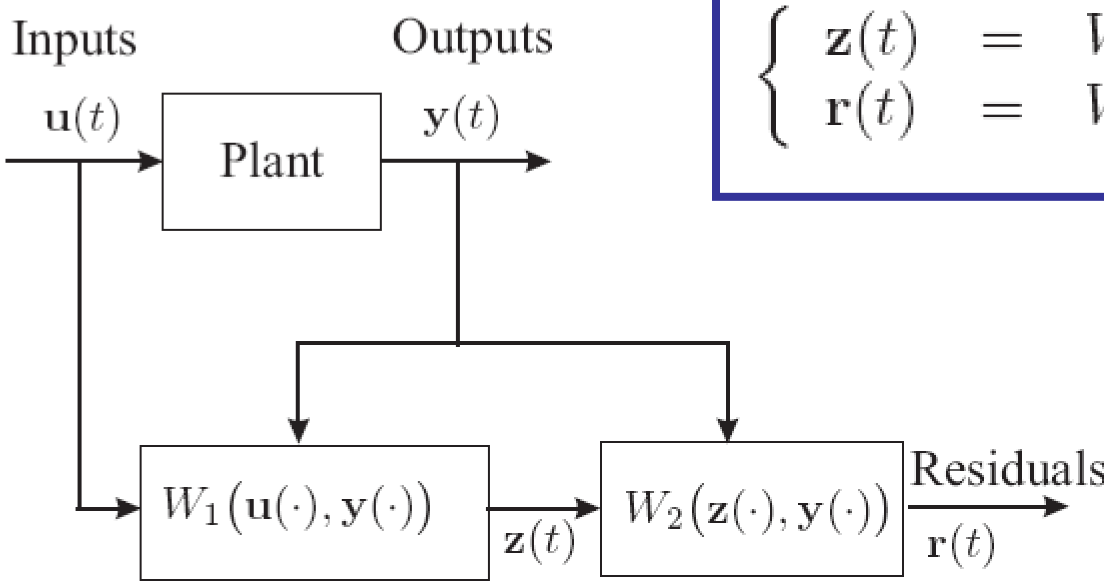



Figure 2.8: Residual generator general structure.

$$\begin{cases} \mathbf{z}(t) = W_1(\mathbf{u}(\cdot), \mathbf{y}(\cdot)) \\ \mathbf{r}(t) = W_2(\mathbf{z}(\cdot), \mathbf{y}(\cdot)) = \mathbf{0}. \end{cases}$$

Under fault-free assumptions, the residual signal *r(t)* is "almost" zero

# Residual General Structure (Cont'd)

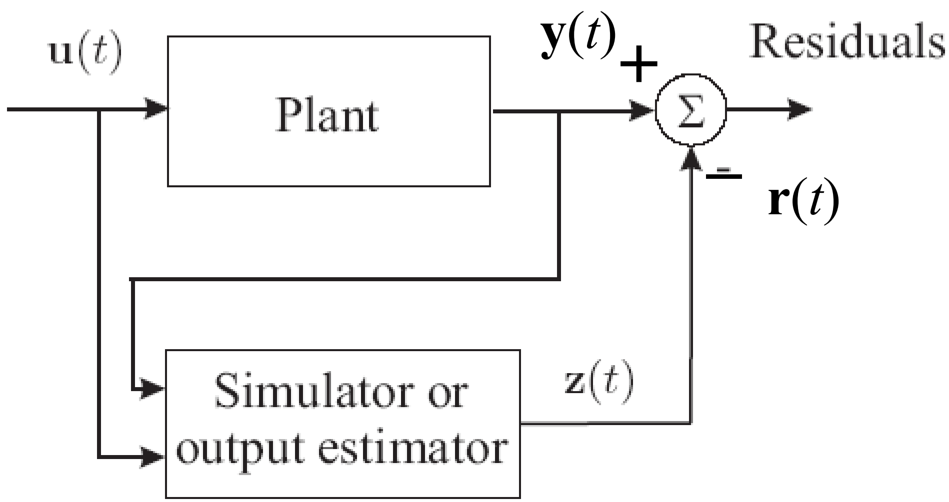



Figure 2.9: Residual generation via system simulator.

Residual generation via *system simulator*

$$\mathbf{r}(t) = \mathbf{y}(t) - \mathbf{z}(t)$$

*z(t)* is the *simulated plant* output

# Residual General Structure (Cont'd)

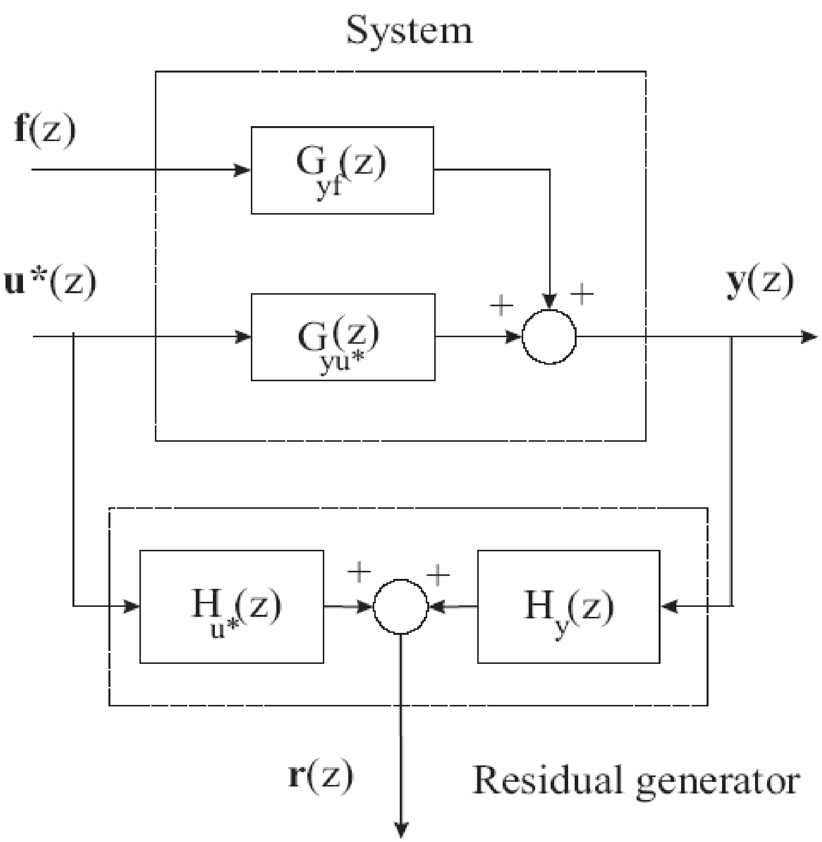



Figure 2.10: Residual generator general structure.

$$\mathbf{y}(z) = \mathbf{G}_{yu^*}(z)\mathbf{u}^*(z) + \mathbf{G}_{yf}(z)\mathbf{f}(z)$$

## Residual generator:

$$\mathbf{r}(z) = \left[ \; \mathbf{H}_{u^*}(z) \quad \mathbf{H}_y(z) \; \right] \left[ \begin{array}{c} \mathbf{u}^*(z) \\ \mathbf{y}(z) \end{array} \right] =$$

$$= H_{u^*}(z)\mathbf{u}^*(z) + \mathbf{H}_y(z)\mathbf{y}(z)$$

$$\mathbf{r}(t) = \mathbf{0} \text{ if and only if } \mathbf{f}(t) = \mathbf{0}$$

## Constraint conditions: *design*

$$\mathbf{H}_{u^*}(z) + \mathbf{H}_y(z)\mathbf{G}_{yu^*} = \mathbf{0}$$

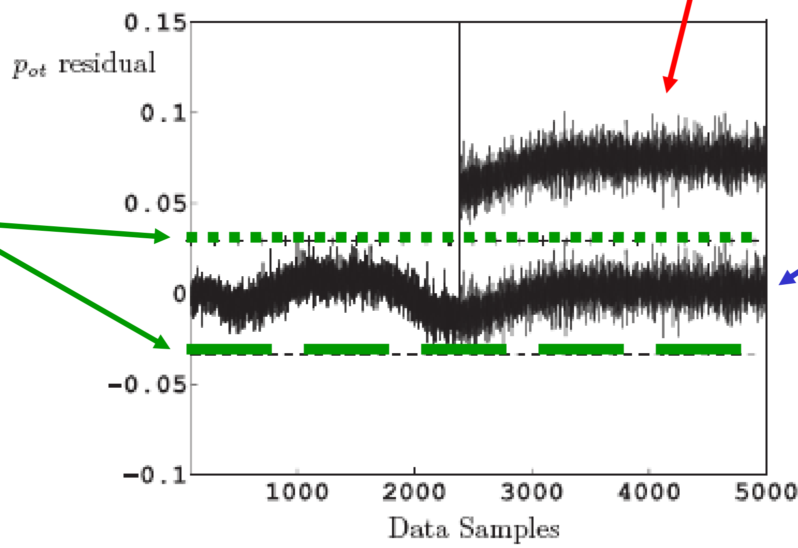
# General Residual Evaluation

$$\begin{cases} J(\mathbf{r}(t)) \leq \varepsilon(t) & \text{for} \quad \mathbf{f}(t) = \mathbf{0} \\ J(\mathbf{r}(t)) > \varepsilon(t) & \text{for} \quad \mathbf{f}(t) \neq \mathbf{0} \end{cases}$$

Faulty residual

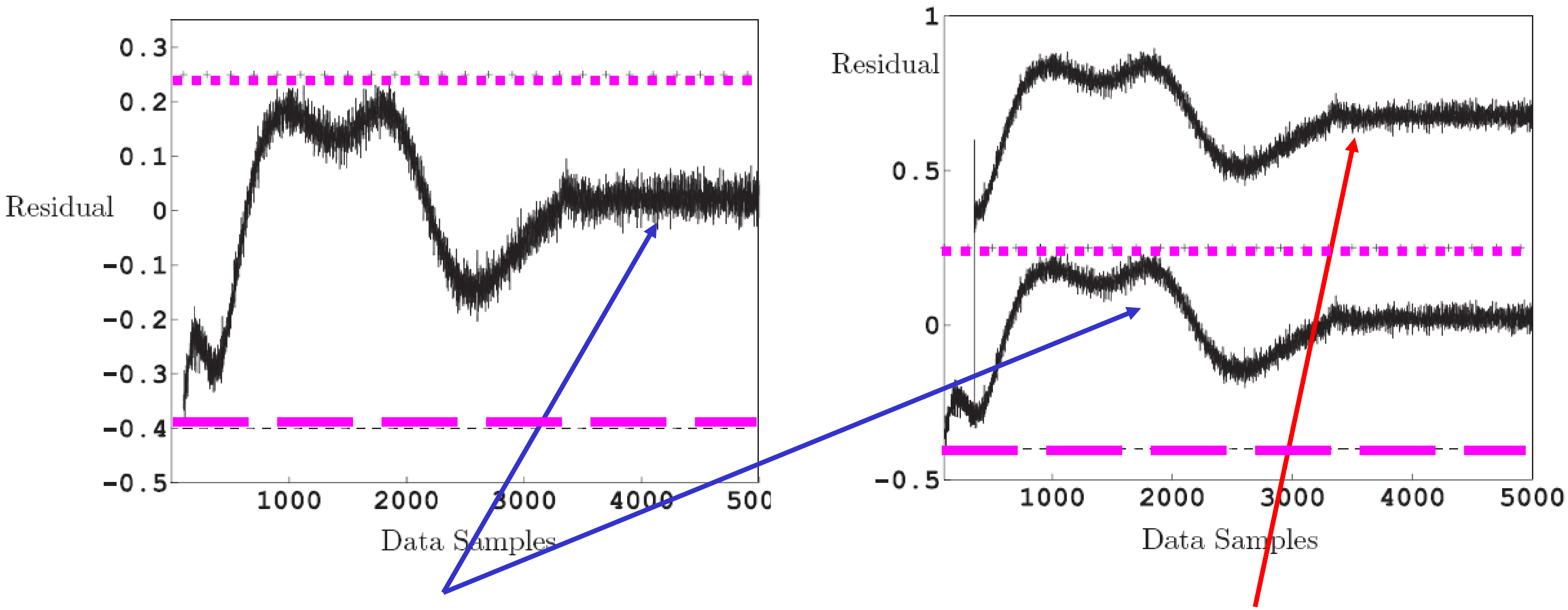
Fault free residual

Detection thresholds $\varepsilon(t)$



# General Residual Evaluation *(example)*

## *Detection thresholds*




**Fault free residual** **Fault-free & faulty residuals**

# Residual Generation Techniques

- **Fault detection via parameter estimation**
- **Observer-based approaches**
- Parity (vector) relations

# Fault Detection via Parameter Estimation

# Parameter Estimation

- Parameter estimation for fault detection
- The process parameters are not known at all, or they are not known exactly enough. They can be determined with parameter estimation methods
- The basic structure of the model has to be known
- Based on the assumption that the faults are reflected in the physical system parameters
- The parameters of the actual process are estimated on-line using well-known parameter estimations methods

# Parameter Estimation (Cont'd)

- The results are thus compared with the parameters of the reference model obtained initially under fault-free assumptions
- Any discrepancy can indicate that a fault may have occurred

- **An approach for modelling the input-output behaviour of the monitored system will be recalled and exploited for fault detection**

# Equation Error (EE) Approach

- SISO model

$$y(t) = \boldsymbol{\Psi}^T \boldsymbol{\Theta}$$

- Parameter vector

$$\boldsymbol{\Theta}^T = [a_1 \dots a_n,\ b_1 \dots b_n]$$

- Regression vector

$$\boldsymbol{\Psi}^T = [y(t-1) \dots y(t-n) \quad u(t-1) \dots u(t-n)]$$

# Equation Error Method

- *Equation error*

$$e(t) = y(t) - \boldsymbol{\Psi}^T \boldsymbol{\Theta}$$

- Model of the process ($Z$-transform)

$$\frac{y(t)}{u(t)} = \frac{B(z)}{A(z)}$$

- Estimated polynomials

$$e(t) = \hat{B}(z)u(t) - \hat{A}(z)y(t)$$

# Equation Error Method (Cont'd)

- Estimation of the process model: **LS**

$$\hat{\boldsymbol{\Theta}} = [\boldsymbol{\Psi}^T \ \boldsymbol{\Psi}]^{-1} \boldsymbol{\Psi}^T y$$

- LS minimisation

$$\begin{cases} J(\boldsymbol{\Theta}) & = & \sum_t e^2(t) = \boldsymbol{e}^T \boldsymbol{e} \\ \frac{d \ J(\boldsymbol{\Theta})}{d \ \boldsymbol{\Theta}} & = & \mathbf{0}. \end{cases}$$

# Equation Error Method (Cont'd)

- Estimation of the process model: **R**LS

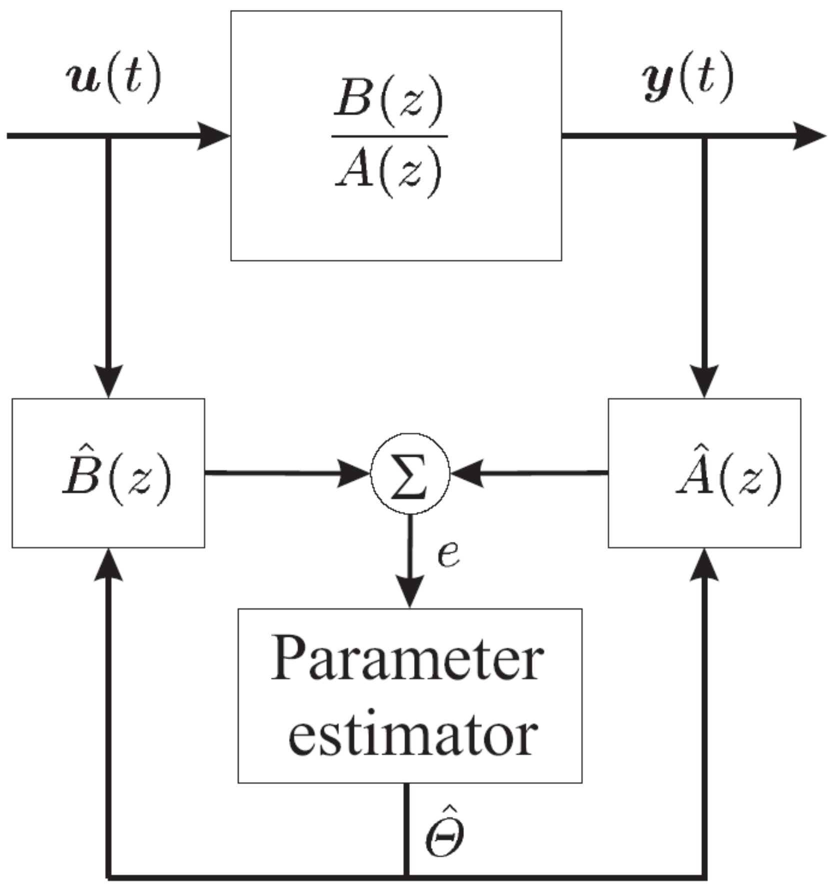
$$\hat{\boldsymbol{\Theta}}(t+1) = \hat{\boldsymbol{\Theta}}(t) + \gamma(t)\left[y(t+1) - \boldsymbol{\Psi}^T(t+1)\hat{\boldsymbol{\Theta}}(t+1)\right]$$

- Estimate recursive adaptation

$$\begin{cases} \gamma(t) & = & \frac{1}{\boldsymbol{\Psi}^T(t+1)\boldsymbol{P}(t)\boldsymbol{\Psi}(t+1)+1}\boldsymbol{P}(t)\boldsymbol{\Psi}(t+1) \\ \boldsymbol{P}(t+1) & = & \left[I - \gamma(t)\boldsymbol{\Psi}^T(t+1)\right]\boldsymbol{P}(t). \end{cases}$$

*Note: see on-line estimation approach*

# Parameter Estimation via EE




- Recursive estimation of the transfer function polynomials
- Equation error
- Parameter estimation via recursive algorithm
- **RLS**

# Links Between Input-Output and State-Space Discrete-Time LTI Models

# Input-Output Model (EE)

- The input to output discrete-time model behaviour can be mathematically described by a set of ARX Multi-Input Single-Output (MISO) models

$$y_i^*(t) = \sum_{j=1}^{n} \alpha_{i,j} y_i^*(t-j) + \sum_{j=1}^{r} \sum_{k=1}^{n} \beta_{i,j,k} u_j^*(t-k) + \varepsilon_i(t) \; , \quad i = 1, \ldots, m$$

# Input-Output Model (EE)

- *m* is the number of the output variables
- The order *n* and the parameters $\alpha_{i,j}$ and $\beta_{i,j,k}$ with $i = 1,\ldots,m$, of the model are determined by the identification approach
- The term $\varepsilon_i(t)$ takes into account the modelling error (EE)

$$y_i^*(t) = \sum_{j=1}^{n} \alpha_{i,j} y_i^*(t-j) + \sum_{j=1}^{r} \sum_{k=1}^{n} \beta_{i,j,k} u_j^*(t-k) + \varepsilon_i(t)\ , \quad i = 1, \ldots, m$$

# State-Space Equivalent Model

- The input-output EE model has a state space "realisation" as follows:

$$\begin{cases} \boldsymbol{x}_i(t+1) &= \boldsymbol{A}_i \boldsymbol{x}_i(t) + \boldsymbol{B}_i \boldsymbol{u}^*(t) + \boldsymbol{B}_{\omega_i} \varepsilon_i & i = 1, \cdots, m \\ \boldsymbol{y}_i^*(t) &= \boldsymbol{C}_i \boldsymbol{x}_i(t) + \boldsymbol{D}_{\omega_i} \varepsilon_i, & t = 1, 2, \cdots \end{cases}$$

- The matrices ($A_i$, $B_i$, $C_i$, $B_{\omega_i}$, $D_{\omega_i}$) of a state space representation in canonical form of the *n*-th order system are defined as follows:

# State-Space Matrices

$$\boldsymbol{A}_i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_{i,1} & \alpha_{i,2} & \alpha_{i,3} & \cdots & \alpha_{i,n} \end{bmatrix},$$

$$\boldsymbol{C}_i = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix},$$

# State-Space Matrices (Cont’d)

$$\begin{bmatrix} \boldsymbol{B}_i & \boldsymbol{B}_{\omega_i} \\ \boldsymbol{0} & \boldsymbol{D}_{\omega_i} \end{bmatrix} = \boldsymbol{S}_i^{-1} \begin{bmatrix} \beta_{i,1,1} & \cdots & \beta_{i,r,1} & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \beta_{i,1,n} & \cdots & \beta_{i,r,n} & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{S}_i = \begin{bmatrix} -\alpha_{i,1} & -\alpha_{i,2} & \cdots & -\alpha_{i,n} & 1 \\ -\alpha_{i,2} & -\alpha_{i,3} & \cdots & 1 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\alpha_{i,n} & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Note that the matrix $\boldsymbol{S}_i$ is always non-singular

# State-Space Matrices (Cont'd)

## In Matlab:

TF2SS Transfer function to state-space conversion.

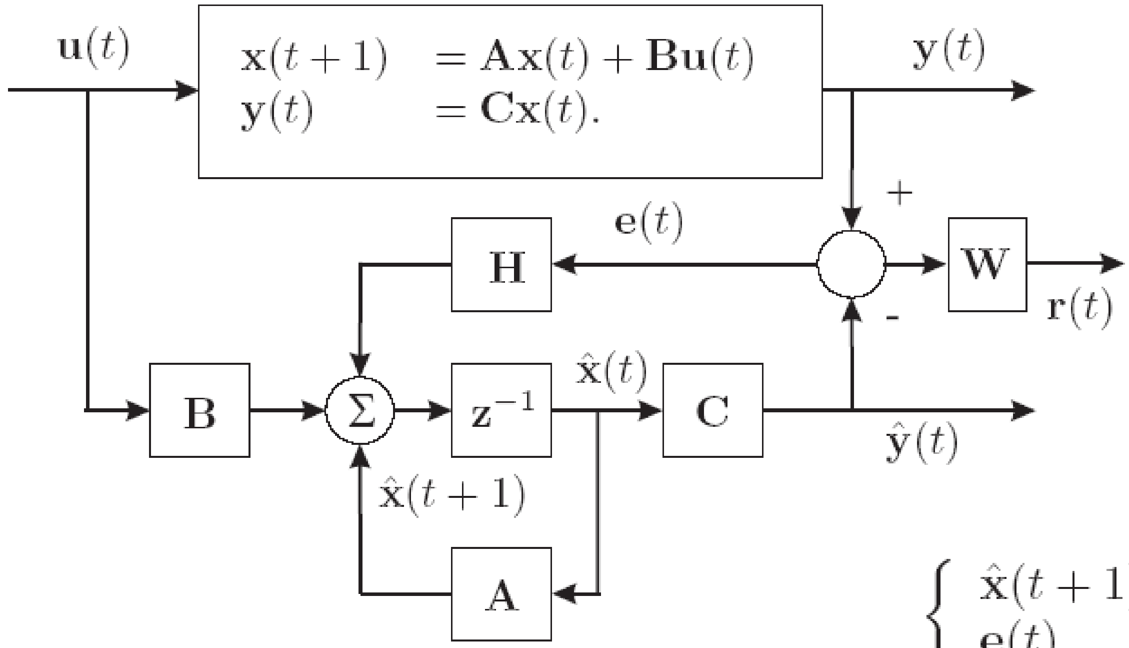
[A,B,C,D] = TF2SS(NUM,DEN) calculates the state-space representation from a single input. Vector DEN must contain the coefficients of the denominator in descending powers of s. Matrix NUM must contain the numerator coefficients with as many rows as there are outputs y

**Note: for MIMO models, use ss and tf functions**

# Observer-based Approaches

# Residual General Structure

- Observer-based approach




Plant model

$$\begin{cases} \mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t). \end{cases}$$

Observer model

$$\begin{cases} \hat{\mathbf{x}}(t+1) = \mathbf{A}\hat{\mathbf{x}}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{H}\mathbf{e}(t) \\ \mathbf{e}(t) = \mathbf{y}(t) - \mathbf{C}\hat{\mathbf{x}}(t). \end{cases}$$

*Output estimation approach!*

# Residual Generator Structure

Plant model

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t). \end{cases}$$

Observer model

$$\begin{cases} \hat{\mathbf{x}}(t+1) &= \mathbf{A}\hat{\mathbf{x}}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{H}\mathbf{e}(t) \\ \mathbf{e}(t) &= \mathbf{y}(t) - \mathbf{C}\hat{\mathbf{x}}(t). \end{cases}$$

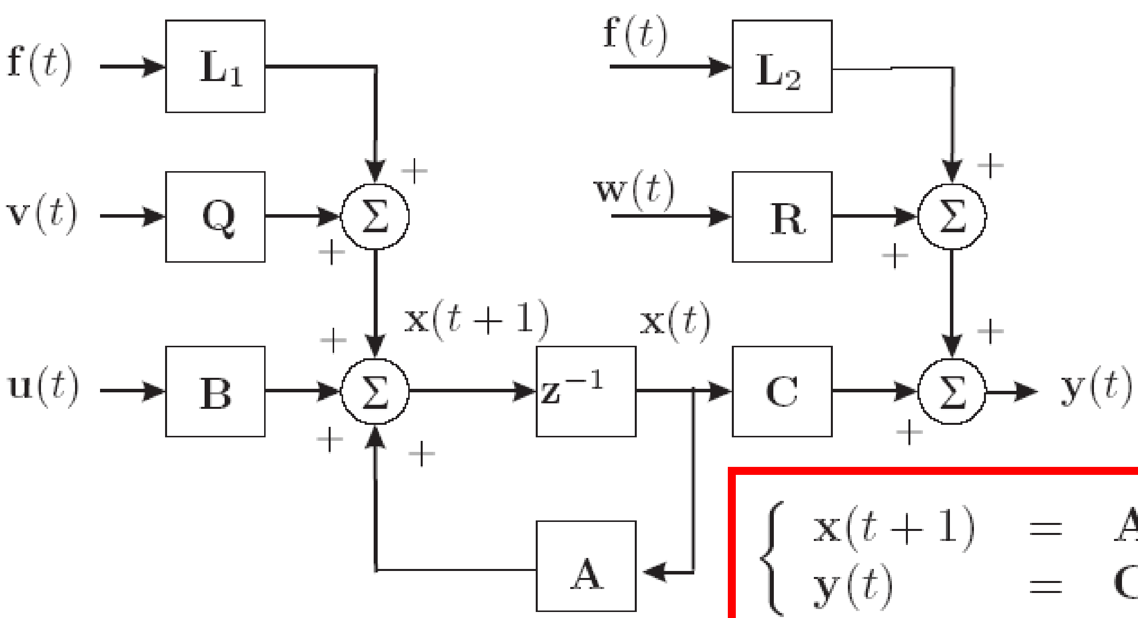
State estimation model

$$\begin{cases} \mathbf{e}_x(t) &= \mathbf{x}(t) - \hat{\mathbf{x}}(t) \\ \mathbf{e}_x(t+1) &= (\mathbf{A} - \mathbf{H}\mathbf{C})\mathbf{e}_x(t). \end{cases}$$

State estimation property $\lim_{t \to \infty} \mathbf{e}_x(t) = \mathbf{0}$ *(fault-free case!!!)*

# Residual Generator Property

**+ disturbance signals and fault**




$$\begin{cases} \mathbf{x}(t+1) & = & \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{Q}\mathbf{v}(t) + \mathbf{L}_1\mathbf{f}(t) \\ \mathbf{y}(t) & = & \mathbf{C}\mathbf{x}(t) + \mathbf{R}\mathbf{w}(t) + \mathbf{L}_2\mathbf{f}(t) \end{cases}$$

Figure 2.14: MIMO process with faults and noises.

# Residual Generator Property (Cont'd)

*+ fault signals*

System model

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{Q}\mathbf{v}(t) + \mathbf{L}_1\mathbf{f}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{R}\mathbf{w}(t) + \mathbf{L}_2\mathbf{f}(t) \end{cases}$$

Observer model

$$\mathbf{e}_x(t+1) = (\mathbf{A} - \mathbf{H}\mathbf{C})\mathbf{e}_x(t) + \mathbf{L}_1\mathbf{f}(t) - \mathbf{H}\mathbf{L}_2\mathbf{f}(t)$$

Output estimation error with faults *but noise-free*

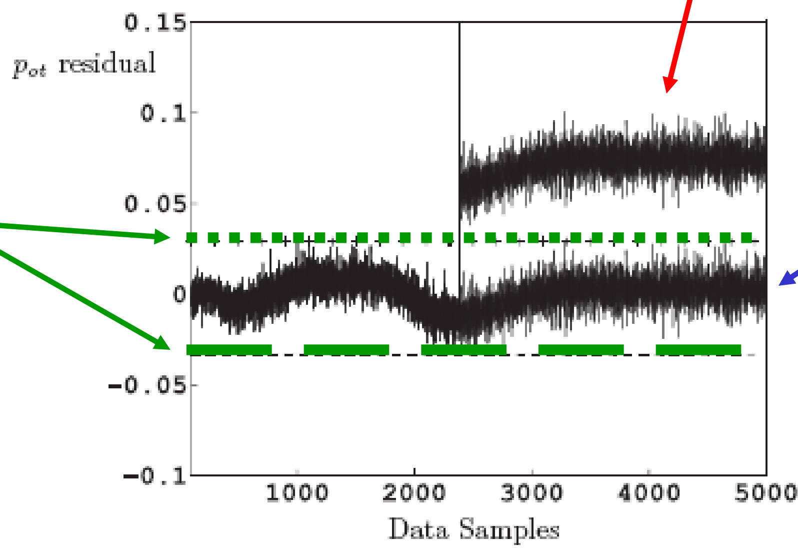
$$\mathbf{e}(t) = \mathbf{C}\mathbf{e}_x(t) + \mathbf{L}_2\mathbf{f}(t).$$

Both *e(t)* and $e_x$(t) are suitable residuals!

# General Residual Evaluation

$$\begin{cases} J(\mathbf{r}(t)) \leq \varepsilon(t) & \text{for} \quad \mathbf{f}(t) = \mathbf{0} \\ J(\mathbf{r}(t)) > \varepsilon(t) & \text{for} \quad \mathbf{f}(t) \neq \mathbf{0} \end{cases}$$

Detection thresholds $\varepsilon(t)$



# Change Detection & Residual Evaluation

$$\begin{cases} J(\mathbf{r}(t)) \leq \varepsilon(t) & \text{for} \quad \mathbf{f}(t) = 0 \\ J(\mathbf{r}(t)) > \varepsilon(t) & \text{for} \quad \mathbf{f}(t) \neq 0 \end{cases}$$

$$J(r(t)) \equiv |r(t)|$$

Detection thresholds

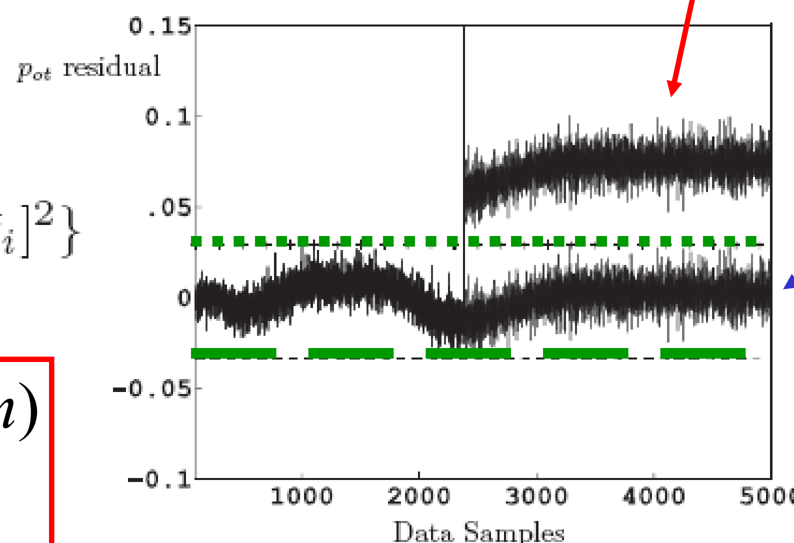
$\varepsilon(t)$

$$\bar{r}_i = E\{r_i(t)\}; \qquad \bar{\sigma}_i^2 = E\{[r_i(t) - \bar{r}_i]^2\}$$

$$\varepsilon(t) = \bar{r}_i \pm \delta \times \bar{\sigma}_i \qquad (i = 1, \cdots, m)$$

with $\delta \geq 3$

Faulty residual




Fault free residual

# Residual Generation

- ✓ Output Observers
- ➢ Recall the output observer design
- ✓ Fault Detection
  - ➢ Fault Isolation, *i.e. where is the fault?*

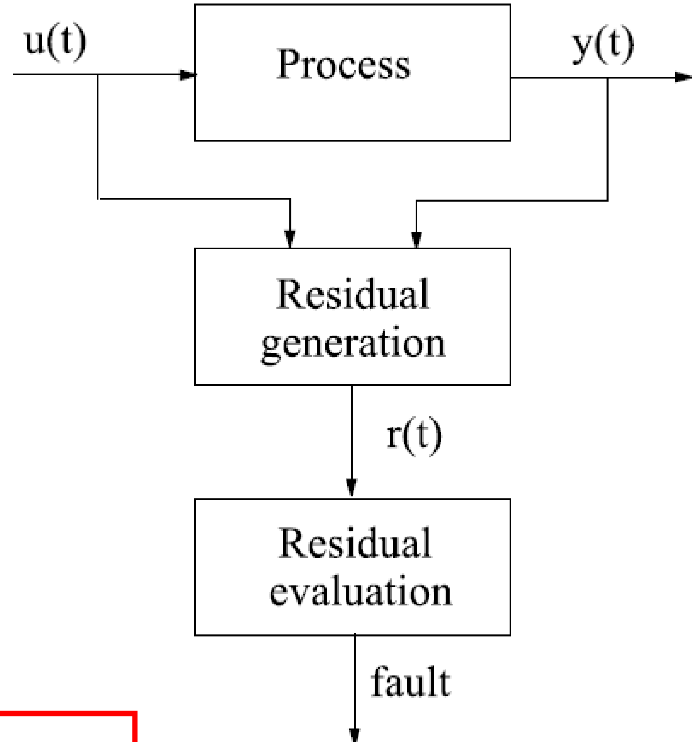
# Output Observer

## Process model with faults

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}(\mathbf{u}(t) + \mathbf{f}_c(t)) + \mathbf{f}_s(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t), \end{aligned}$$

## Input-output sensor faults

$$\left.\begin{aligned} \mathbf{u}(t) &= \mathbf{f}_u(t) + \mathbf{u}^*(t) \\ \mathbf{y}(t) &= \mathbf{f}_y(t) + \mathbf{y}^*(t) \end{aligned}\right\}$$




## Observer for the i-th output $y_i(t)$

$$\mathbf{x}^i(t+1) = \mathbf{A}_i\mathbf{x}^i(t) + \mathbf{B}_i\mathbf{u}(t) + \mathbf{K}_i\left(y_i(t) - \mathbf{C}_i\mathbf{x}^i(t)\right)$$

$(A_i, B_i, C_i)$ is the state-space process model

# Output Observer for *Fault Detection*

Given the observer model

$$\mathbf{x}^i(t+1) = \mathbf{A}_i \mathbf{x}^i(t) + \mathbf{B}_i \mathbf{u}(t) + \mathbf{K}_i \left( y_i(t) - \mathbf{C}_i \mathbf{x}^i(t) \right)$$

Under fault-free assumptions

$$r_i(t) = y_i^*(t) - \hat{y}_i(t) = \mathbf{C}_i \left( \mathbf{x}_i(t) - \mathbf{x}^i(t) \right) \text{ is equal to zero.}$$

Fault detection logic

fixed threshold $\epsilon$,

$$\left. \begin{array}{lll} \mathbf{r}(t) \leq \epsilon & \text{for} & \mathbf{f}(t) = \mathbf{0} \\ \mathbf{r}(t) > \epsilon & \text{for} & \mathbf{f}(t) \neq \mathbf{0} \end{array} \right\}$$

$\mathbf{f}(t)$ being a generic failure vector.

# Output Observer for *Fault Isolation*

$$\left.\begin{array}{lcl}\mathbf{x}(t+1) & = & \mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t)+\mathbf{B}\mathbf{f}_u(t)\\ \mathbf{y}(t) & = & \mathbf{C}\mathbf{x}(t)+\mathbf{f}_y(t)\end{array}\right\}$$

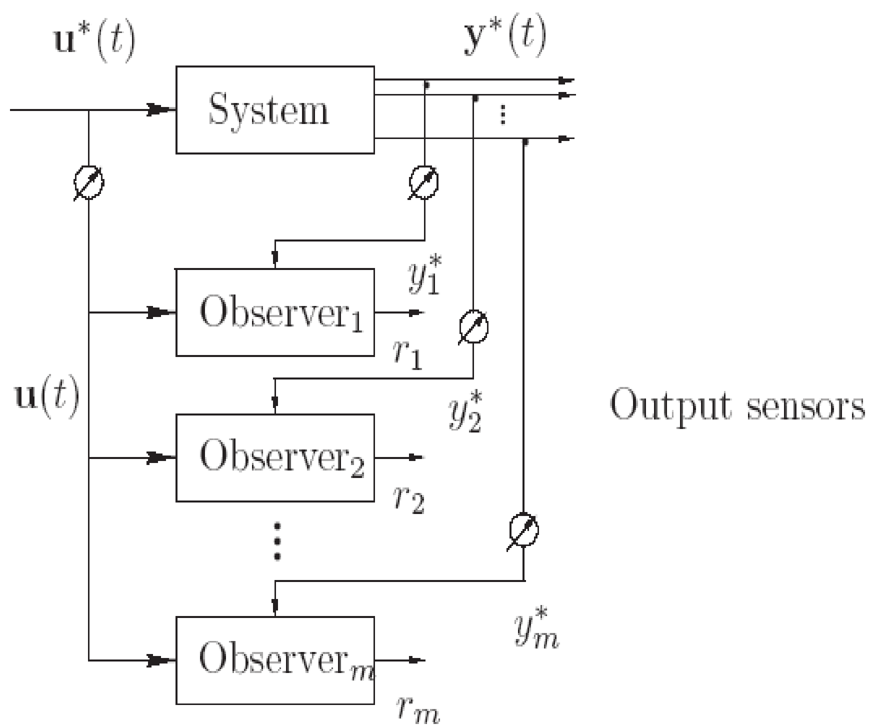
Process model

$$\mathbf{x}^i(t+1)=\mathbf{A}_i\mathbf{x}^i(t)+\mathbf{B}_i\mathbf{u}(t)+ \\ +\mathbf{K}_i\left(y_i(t)-\mathbf{C}_i\mathbf{x}^i(t)\right)$$

$$r_i(t)=y_i^*(t)-\hat{y}_i(t)=\mathbf{C}_i\left(\mathbf{x}_i(t)-\mathbf{x}^i(t)\right)$$

$$y_i(t)=y_i^*(t)+f(t)$$

## Bank of output observers



# Output Observer for *Fault Isolation* (Cont'd)

**Bank of output observers**

$$\mathbf{x}^i(t+1) = \mathbf{A}_i\mathbf{x}^i(t) + \mathbf{B}_i\mathbf{u}(t) + \mathbf{K}_i\left(y_i(t) - \mathbf{C}_i\mathbf{x}^i(t)\right)$$

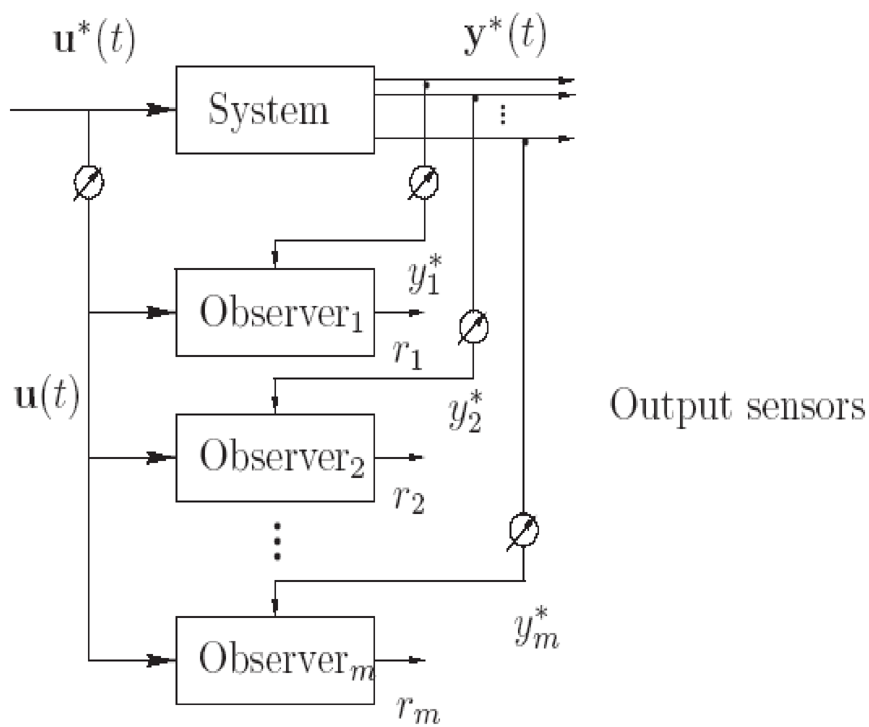
$$r_i(t) = y_i^*(t) - \hat{y}_i(t) = \mathbf{C}_i\left(\mathbf{x}_i(t) - \mathbf{x}^i(t)\right)$$

*Fault-free case:*

$$\lim_{t\to\infty} r_i(t) = \lim_{t\to\infty}\left(y_i(t) - C^i\mathbf{x}^i(t)\right) = 0$$

*Faulty case*

$$y_i(t) = y_i^*(t) + f(t)$$

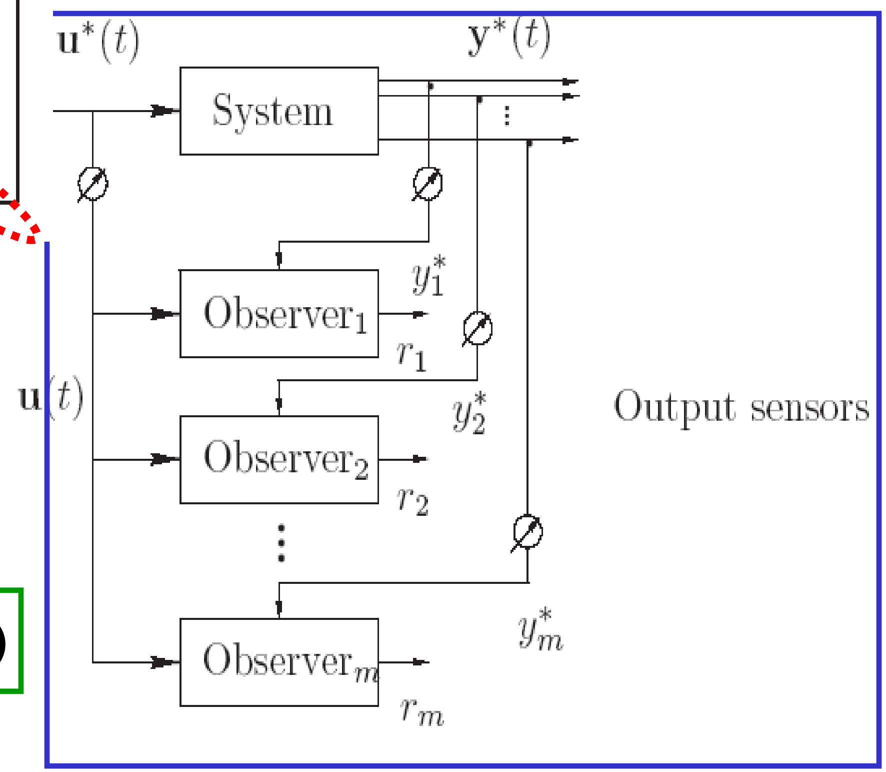


# Output Observer for *Fault Isolation* (Cont'd)

Table 4.1: Fault signatures.

| | $u_1$ | $u_2$ | $\ldots$ | $u_r$ | $y_1$ | $y_2$ | $\ldots$ | $y_m$ |
|---|---|---|---|---|---|---|---|---|
| $r_{O_1}$ | 1 | 1 | $\ldots$ | 1 | 1 | 0 | $\ldots$ | 0 |
| $r_{O_2}$ | 1 | 1 | $\ldots$ | 1 | 0 | 1 | $\ldots$ | 0 |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $r_{O_m}$ | 1 | 1 | $\ldots$ | 1 | 0 | 0 | $\ldots$ | 1 |

**Bank of output observers**




$$\mathbf{x}^i(t+1) = \mathbf{A}_i \mathbf{x}^i(t) + \mathbf{B}_i \mathbf{u}(t) + \quad + \mathbf{K}_i \left( y_i(t) - \mathbf{C}_i \mathbf{x}^i(t) \right)$$

*Fault-free case:*

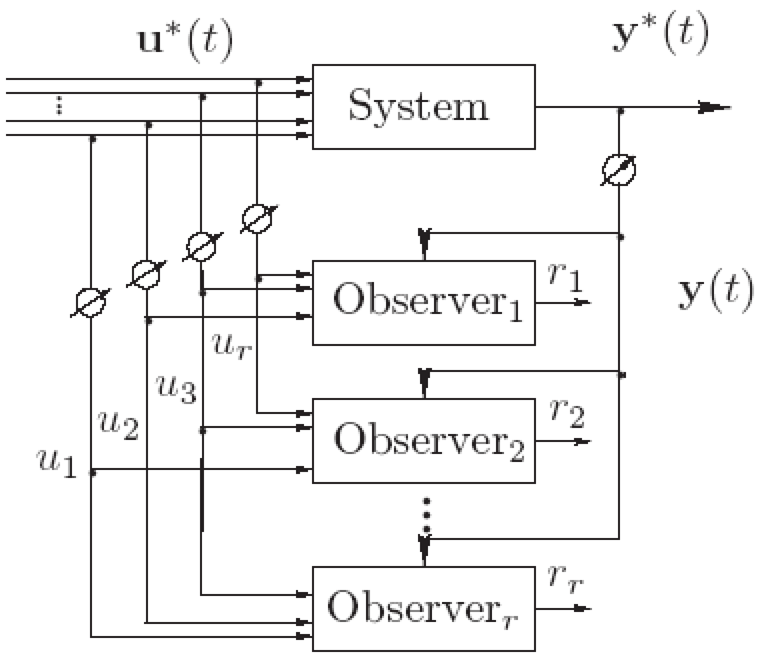
$$\lim_{t \to \infty} r_i(t) = \lim_{t \to \infty} \left( y_i(t) - C^i \mathbf{x}^i(t) \right) = 0$$

*Faulty case*

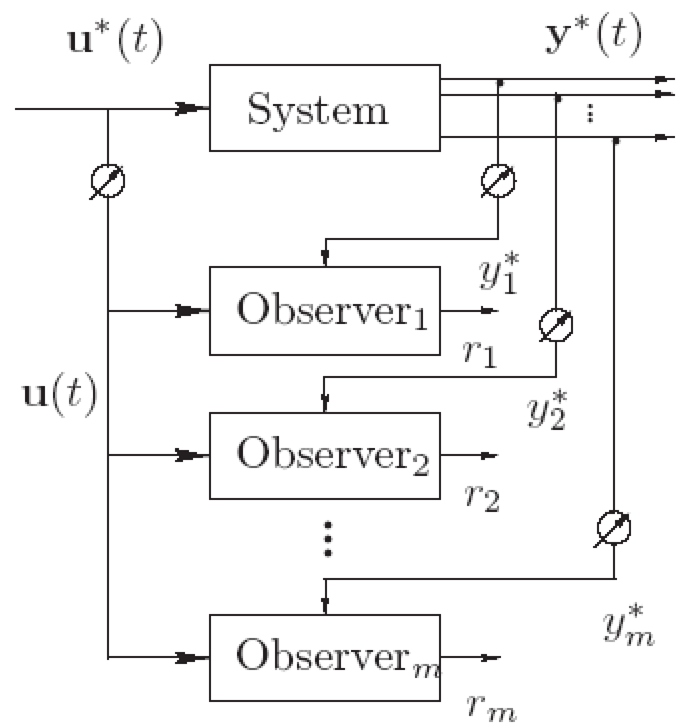
$$y_i(t) = y_i^*(t) + f(t)$$

$$\lim_{t \to \infty} r_i(t) \neq 0$$

# Multiple FDI




***Input sensor FDI***




***Output sensor FDI***

# Residual Disturbance *Robustness*

- Residuals decoupled from disturbance
- Robust residual generator
- Disturbance effect minimisation
- Measurement errors

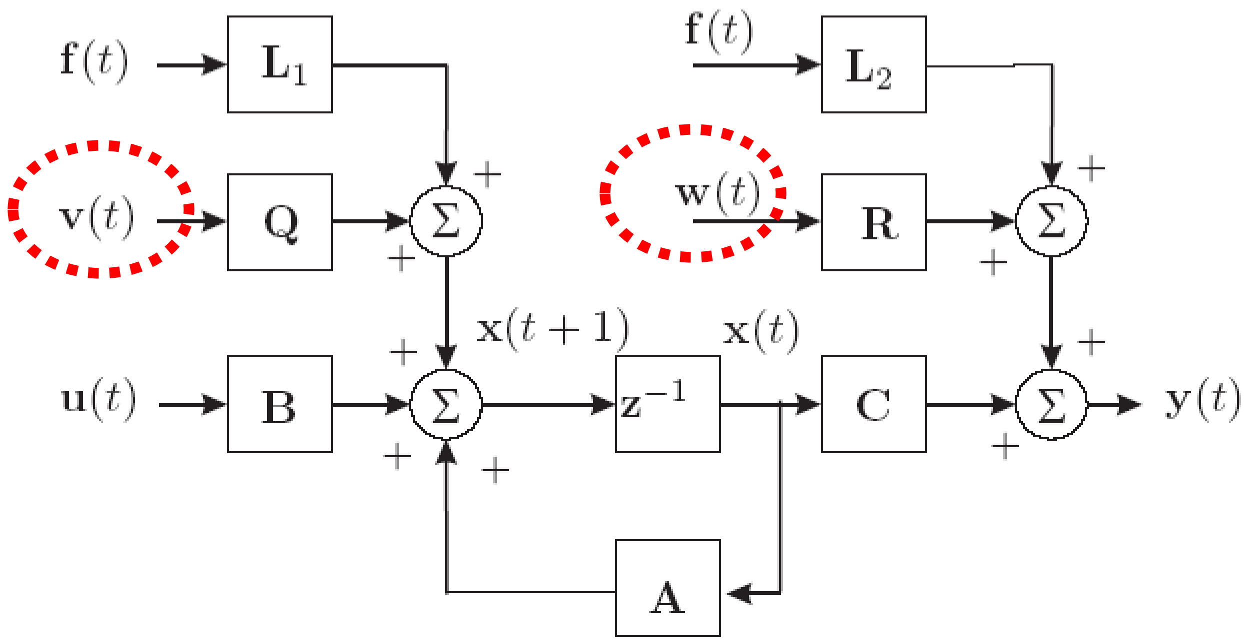



Figure 2.14: MIMO process with faults and noises.

# FDI with *Noisy Measurements*

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{Q}\mathbf{v}(t) + \mathbf{L}_1\mathbf{f}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{R}\mathbf{w}(t) + \mathbf{L}_2\mathbf{f}(t) \end{cases}$$

✓ Model with fault and noise

$$\begin{cases} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{Q}\mathbf{v}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{R}\mathbf{w}(t) \end{cases}$$

➢ Model with noise only: Kalman filter!

# Fault Detection with Parity Equations

# Parity Relations for Fault Detection

- The basic idea of the parity relations approach is to provide a proper check of the **parity (consistency)** of the **measurements** acquired from the monitored system
- In the early development of fault diagnosis, the **parity vector (relation)** approach was applied to static or parallel redundancy schemes, which may be obtained **directly from measurements (hardware redundancy) or from analytical relations (analytical redundancy)**

# Parity Relations for Fault Detection

- In the case of hardware redundancy, two methods can be exploited to obtain redundant relations
- The first requires the use of **several sensors** having identical or **similar functions to measure the same variable**
- The second approach consists of **dissimilar sensors** to measure different variables but with **their outputs being relative to each other**
- **Analytical forms of redundancy**

# **Analytical** Redundancy

- Model ($M$) and process ($P$)

$$G_M(z) = \frac{\hat{\boldsymbol{A}}(z)}{\hat{\boldsymbol{B}}(z)} \qquad G_P(z) = \frac{\boldsymbol{A}(z)}{\boldsymbol{B}(z)}$$

- Error vector

$$\boldsymbol{r}(z) = \left( \frac{\boldsymbol{A}(z)}{\boldsymbol{B}(z)} - \frac{\hat{\boldsymbol{A}}(z)}{\hat{\boldsymbol{B}}(z)} \right) \boldsymbol{u}(z)$$

# Analytical Redundancy (OE)

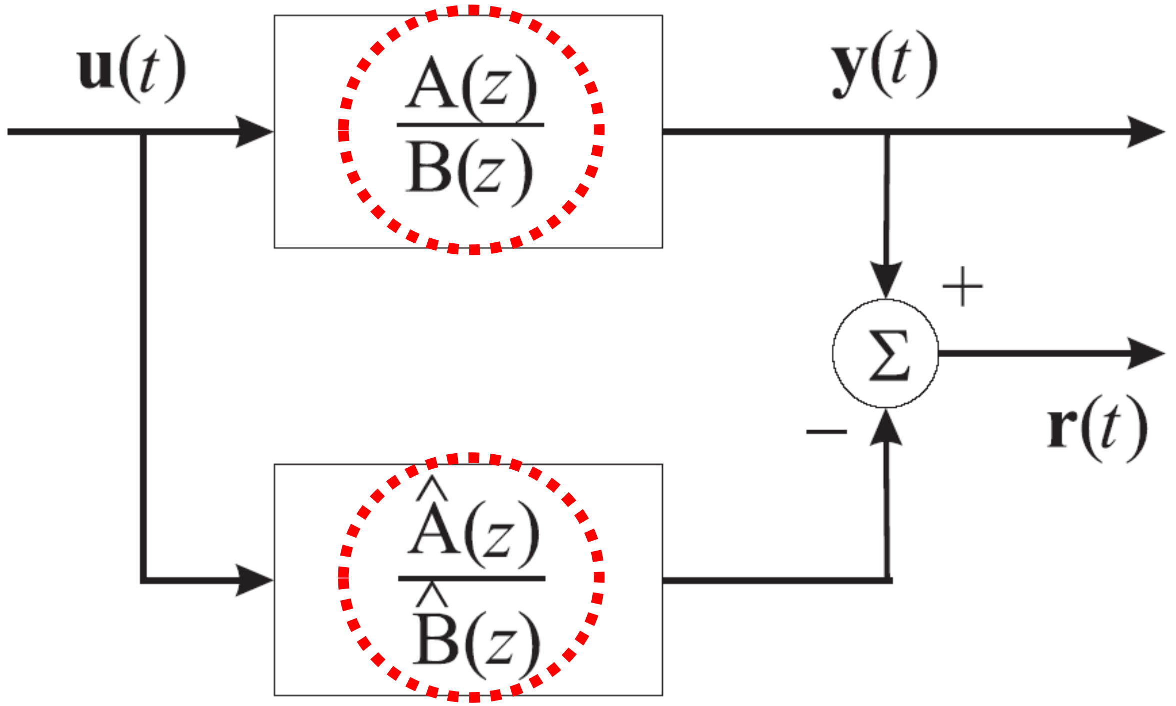
- Models, if:

$$\boldsymbol{G}_M(z) = \boldsymbol{G}_P(z) \;\; i.e. \;\; \frac{\hat{\boldsymbol{A}}(z)}{\hat{\boldsymbol{B}}(z)} = \frac{\boldsymbol{A}(z)}{\boldsymbol{B}(z)}$$

- Residual, with input and output faults

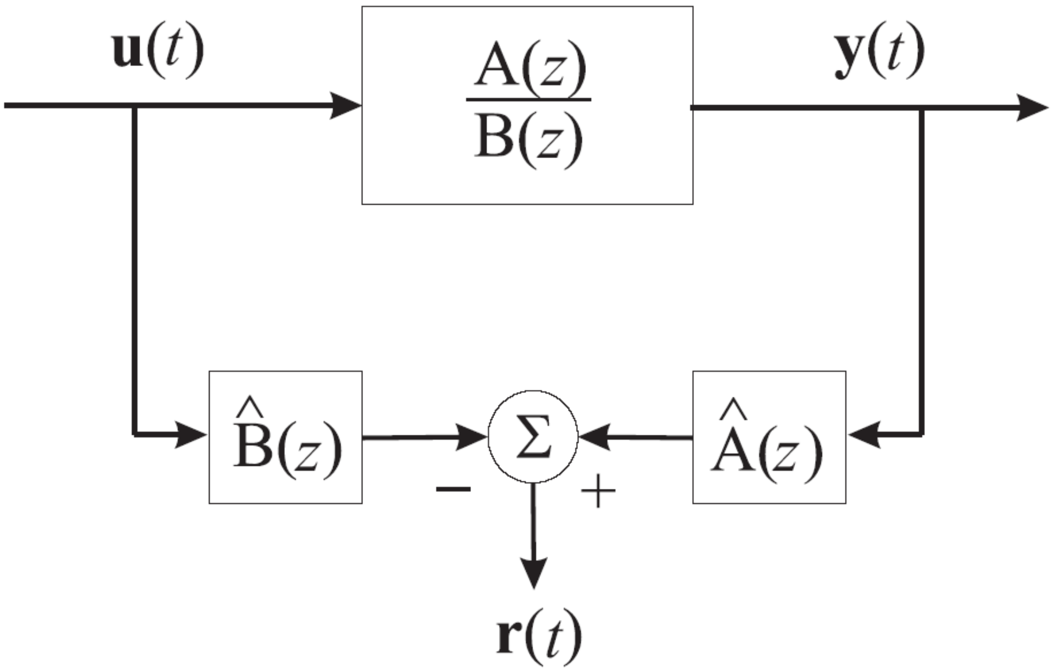
$$\boldsymbol{r}(z) = \frac{\boldsymbol{A}(z)}{\boldsymbol{B}(z)}\boldsymbol{f}_u(z) + \boldsymbol{f}_y(z)$$

# Analytical Redundancy (OE)




(a) Output error

# Parity Relation (EE)




(b) Equation error

# Parity Relations via **EE**

- According to EE, another possibility for generating a polynomial error:

$$\begin{aligned} \boldsymbol{r}(z) &= \hat{\boldsymbol{A}}(z)\boldsymbol{y}(z) - \hat{\boldsymbol{B}}(z)\boldsymbol{u}(z) \\ &= \boldsymbol{B}(z)\boldsymbol{f}_u(z) + \boldsymbol{A}(z)\boldsymbol{f}_y(z) \end{aligned}$$

# Parity Relations

- The previous equations that generate residuals and are called parity equations under the assumptions of fault occurrence and **of exact agreement between process and model**
- However, within the parity equations, the model parameters are assumed to be known and constant, whereas the **parameter estimations** can vary the parameters of the polynomials in order to minimise the residuals

# Change Detection and Symptom Evaluation

# Residual Evaluation

- When the residual generation stage has been performed, the second step requires the examination of symptoms in order to determine if any faults have occurred
- A decision process may consist of a simple threshold test on the instantaneous values of moving averages of residuals

# Residual Evaluation (Cont'd)

- On the other hand, because of the presence of noise, disturbances and other unknown signals acting upon the monitored system, the decision making process can exploits statistical methods
- In this case, the measured or estimated quantities, such as signals, parameters, state variables or residuals are usually represented by stochastic variables

# Residual Evaluation (Cont’d)

- Mean and standard values

$$\bar{r}_i = E\{r_i(t)\}; \qquad \bar{\sigma}_i^2 = E\{[r_i(t) - \bar{r}_i]^2\}$$

- Residuals or symptoms

$$\Delta r_i = E\{r_i(t) - \bar{r}_i\}; \qquad \Delta\sigma_i = E\{\sigma_i(t) - \bar{\sigma}_i\}$$

# Residual Evaluation (Cont'd)

- Fixed threshold selection

$$\Delta r_{tol} = \epsilon \bar{\sigma}_r, \qquad \epsilon \geq 2$$

- By a proper choice of ε, a compromise has to be made between the detection of small faults and false alarms
- More complex residual evaluation schemes

# Residual Generation Problem

- **Robustness issues!**
- Two design principles:
  - Uncertainty is taken into account at the residual design stage: active robustness in fault diagnosis
  - Passive robustness makes use of a residual evaluator with proper threshold selection methods (fixed or adaptive)

# Fault Diagnosis Technique Integration

# FDI Technique Integration

- Several FDI techniques have been developed and their application shows different properties with respect of the diagnosis of different faults in a process
- To achieve a reliable FDI technique, a good solution consists of a proper integration of several methods which take advantages of the different procedures
- Exploit a knowledge-based treatment of all available analytical and heuristic information

# **Fuzzy Logic** for Residual Generation

- Classical fault diagnosis model-based methods can exploit state-space of input-output dynamic models of the process under investigation
- Faults are supposed to appear as changes on the system state or output caused by malfunctions of the components as well as of the sensors
- The main problem with these techniques is that the precision of the process model affects the accuracy of the detection and isolation system as well as the diagnostic sensitivity

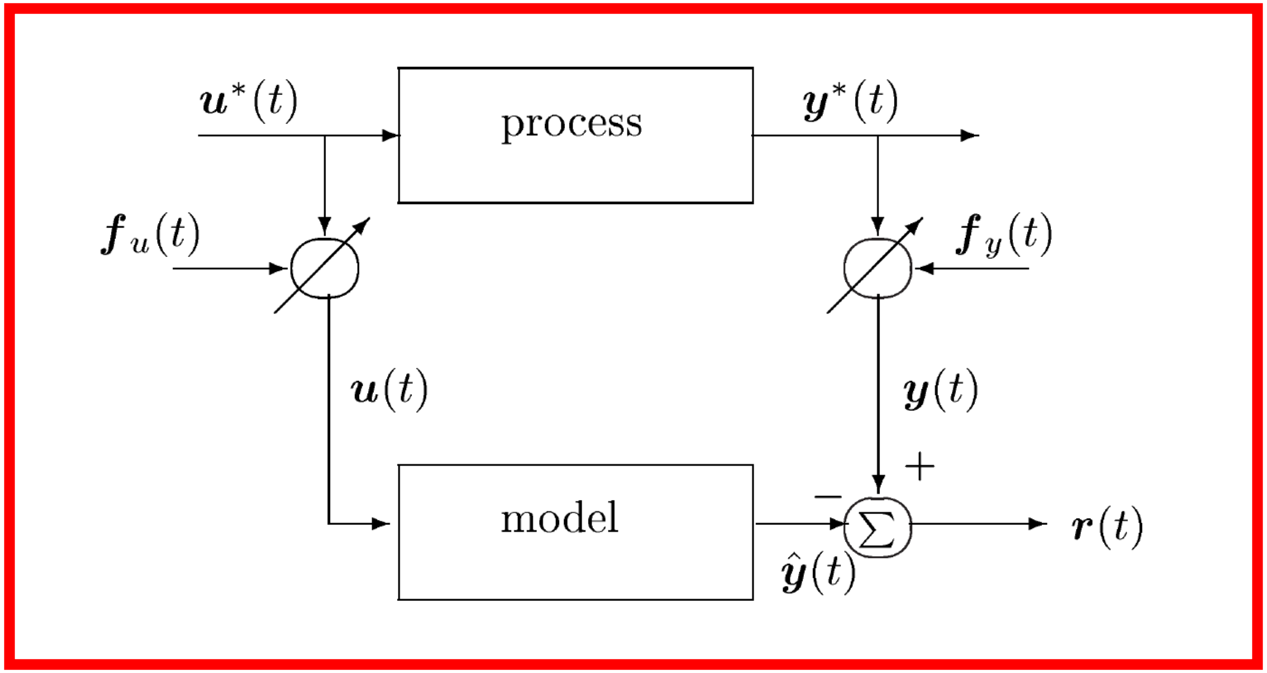
# Fuzzy Logic for Residual Generation (Cont'd)

- The majority of real industrial processes are nonlinear and cannot be modelled by using a single model for all operating conditions
- Since a mathematical model is a description of system behaviour, accurate modelling for a complex nonlinear system is very difficult to achieve in practice
- Sometimes for some nonlinear systems, it can be impossible to describe them by analytical equations

# Fuzzy Logic for Residual Generation (Cont'd)

- Sometimes the system structure or parameters are not precisely known and if diagnosis has to be based primarily on heuristic information, no qualitative model can be set up
- Because of these assumptions, fuzzy system theory seems to be a natural tool to handle complicated and uncertain conditions
- Instead of exploiting complicated nonlinear models, it is also possible to describe the plant by a collection of local affine fuzzy models, whose parameters are obtained by identification procedures

# Residual Generation via Fuzzy Models




Resisual signals: $\boldsymbol{r}(t) = \boldsymbol{y}(t) - \hat{\boldsymbol{y}}(t).$

# **Neural Networks** in Fault Diagnosis

- Quantitative model-based fault diagnosis generates symptoms on the basis of the analytical knowledge of the process under investigation
- In most cases this does not provide enough information to perform an efficient FDI, *i.e.*, to indicate the location and the mode of the fault
- A typical integrated fault diagnosis system uses both analytical and heuristic knowledge of the monitored system

# Neural Networks in Fault Diagnosis (Cont'd)

- The knowledge can be processed in terms of residual generation (analytical knowledge) and feature extraction (heuristic knowledge)
- The processed knowledge is then provided to an inference mechanism which can comprise residual evaluation, symptom observation and pattern recognition

# Neural Networks in Fault Diagnosis (Cont'd)

- Neural networks (NN) have been used successfully in pattern recognition as well as system identification, and they have been proposed as a possible technique for fault diagnosis, too
- NN can handle nonlinear behaviour and partially known processes because they learn the diagnostic requirements employing the information from the training data

# Neural Networks in Fault Diagnosis (Cont'd)

- NN are noise tolerant and their ability to generalise the knowledge as well as to adapt during use are extremely interesting properties
- FDI is performed by a NN using input and output measurements
  - NN is trained to identify the fault from measurement patterns
  - Classification of individual measurement pattern is not always unique in dynamic situations
- Fault diagnosis of dynamic plant is not practical and other approaches should be investigated

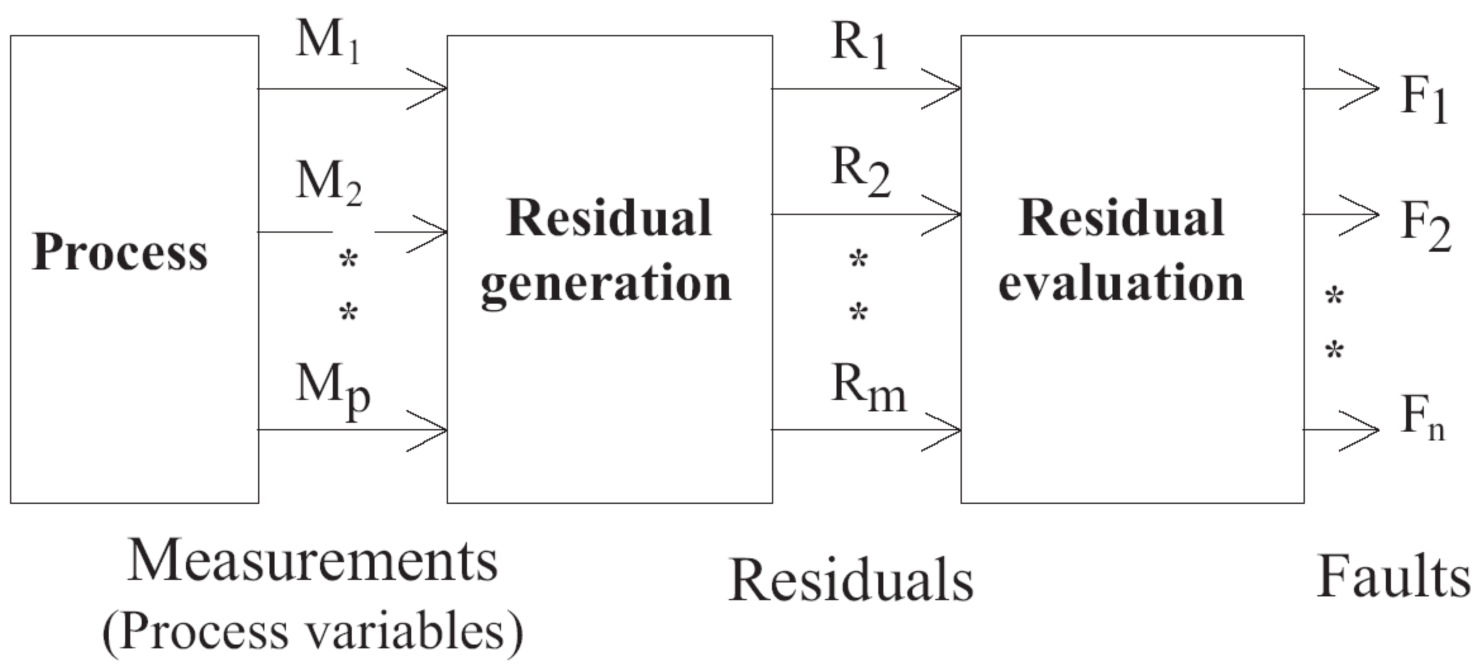
# Neural Networks in Fault Diagnosis (Cont'd)

- A NN could be exploited in order to find a dynamic model of the monitored system or connections from faults to residuals
- In the latter case, the NN is used as pattern classifier or nonlinear function approximator
- NN are capable of approximating a large class of functions for fault diagnosis of an industrial plant
- The identification of models for the system under diagnosis as well as the application of NN as function approximator will be shown

# Neural Networks in Fault Diagnosis (Cont'd)

- Quantitative and qualitative approaches have a lot of complementary characteristics which can be suitably combined together to exploit their advantages and to increase the robustness of quantitative techniques
- Partial knowledge deriving from qualitative reasoning is reduced by quantitative methods
- Further research on model-based fault diagnosis consists of finding the way to properly combine these two approaches together to provide highly reliable diagnostic information

# FDI with Neural Networks



# FDI with Neural Networks (Cont'd)

- As described in the figure, the fault diagnosis methodology consist of 2 stages
- In 1st stage, the fault has to be detected on the basis of residuals generated from a bank of output estimators, while, in the 2nd step, fault identification is obtained from pattern recognition techniques implemented via NN
- Fault identification represents the problem of the estimation of the size of faults occurring in a dynamic system

# FDI with Neural Networks (Cont’d)

- A NN is exploited to find the connection from a particular fault regarding system inputs and output measurements to a particular residual
- The output predictor generates a residual which does not depend on the dynamic characteristics of the plant, but only on faults
- NNs classify static patterns of residuals, which are uniquely related to particular fault conditions independently from the plant dynamics

# FDI with Neural Networks (Cont'd)

- NNs have been used both as predictor of dynamic models for fault diagnosis, and pattern classifiers for fault identification
- The most frequently applied neural models are the feed-forward perceptron used in multi-layer networks with static structure
- The introduction of explicit dynamics requires the feedback of some outputs through time delay units

# FDI with Neural Networks (Cont'd)

- Alternatively to static structure, NN with neurons having intrinsic dynamic properties can be used
- On the other hand, NN can be effectively exploited for residual signal processing, which is actually a static patter recognition problem

# FDI with Neural Networks (Cont'd)

- Fault signals create changes in several residuals obtained by using output predictors of the process under examination
- A neural network is exploited in order to find the connection from a particular fault regarding input and output measurements to a particular residual

# FDI with Neural Networks (Cont'd)

- The predictors generate residuals independent of the dynamic characteristics of the plant and dependent only on sensors faults
- Therefore, the neural network evaluates static patterns of residuals, which are uniquely related to particular fault conditions independently from the plant dynamics

# Conclusion – **Part 1**

- ✓ Model-Based FDI
- ✓ Analytical Redundancy
- ✓ State-Space Models
- ✓ Residual Generation
  - ✓ Dynamic Observers
    - ✓ Output observers
- ✓ Residual Evaluation/Change Detection