Structural analysis for fault diagnosis of equation based models

Erik Frisk <erik.frisk@liu.se>
Department of Electrical Engineering
Linköping University, Sweden

LINKÖPING UNIVERSITY

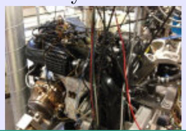
Jubilee Symposium — Future Directions of System Modeling and Simulation
Sept. 30, Lund, Sweden

Model based diagnosis, basic idea

faults $f(t)$

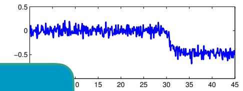
actuators $u(t)$

System



observation $y(t)$

residual $r(t)$



Industrial applications often more difficult ...

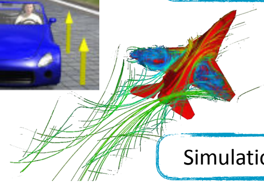
- Models are complex, non-linear, includes lookup-tables, to big to handle by hand, ...
- Fault isolation, not only fault detection
- Models are uncertain, which, by definition is not modeled

Modeling languages

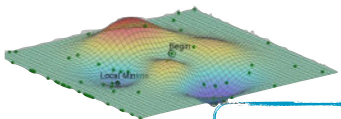
MODELICA MATLAB SIMULINK

- Simulink and Modelica are used (in industry) for
  - Mainly simulation
  - optimization
  - not so much for diagnosis analysis and design
- Support for Simulink and Modelica would make methods industrially more accessible
- We in Linköping has thought about this for some time; diagnostic methods useful for such models
- Maybe have to compromise between general applicability and optimality/guarantees/...

Models

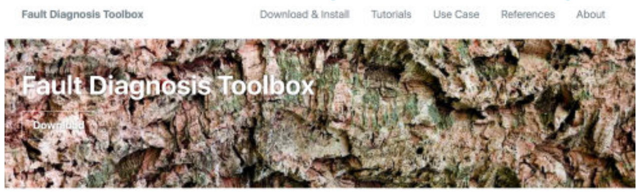


Simulation



Optimization

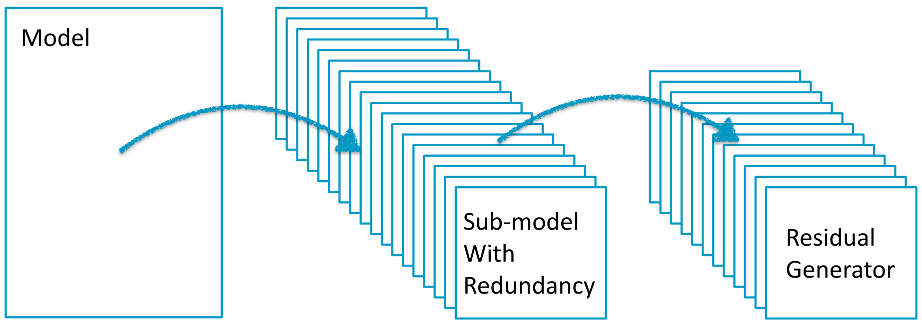
A Matlab toolbox – faultdiagnosistoolbox.github.io



Main designer, coding, and algorithms
Erik Frisk (http://users.isy.liu.se/fs/frisk/) <erik.frisk@liu.se>
Professor, Linköping University, Sweden

Coding and algorithms
Mattias Krysander (http://users.isy.liu.se/fs/matkr/) <mattias.krysander@liu.se>
Associate professor, Linköping University, Sweden

Basic approach to diagnosis system design



DAEs and equation based models for diagnosis

- Non-causal models — inherent in the diagnosis problem
  - A signal is known or unknown; it does not matter if it is an input or output signal to the system
  - x, z, f — unknown, known, and fault signals
    $$F(\dot{x}, x, z, f) = 0$$
- Submodels — inherently differential-algebraic

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, z, f) \\ \dot{x}_2 &= f_2(x_1, x_2, z, f) \\ y_1 &= h_1(x_1, x_2, z, f) \\ y_2 &= h_2(x_1, x_2, z, f) \end{aligned}$$

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, z, f) \\ y_1 &= h_1(x_1, x_2, z, f) \end{aligned}$$

Differential index and diagnosis filter design

- Standard definition on differential index is for just-determined models but can be directly extended to over-determined models, i.e., models with redundancy,
$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2, z, f) \\ y_1 &= h_1(x_1, x_2, z, f) \end{aligned}$$
- If sub-model is low-index, standard observer design techniques can be utilized for a fault detector in the form
$$\begin{aligned} \dot{\hat{x}}_1 &= g_1(\hat{x}_1, \hat{x}_2, z) + K g_r(\hat{x}_1, \hat{x}_2, z) \\ 0 &= g_a(\hat{x}_1, \hat{x}_2, z) \\ r &= g_r(\hat{x}_1, \hat{x}_2, z) \end{aligned}$$
- Thus low-index sub-models are of particular interest for detector synthesis

Structural models

Structural model

A structural model only models that variables are related!

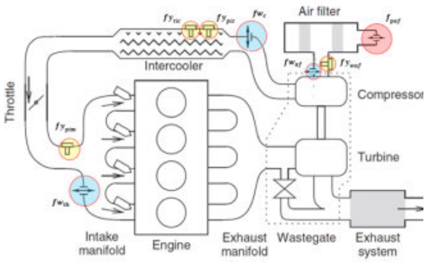
Example relating variables: V, i, ω

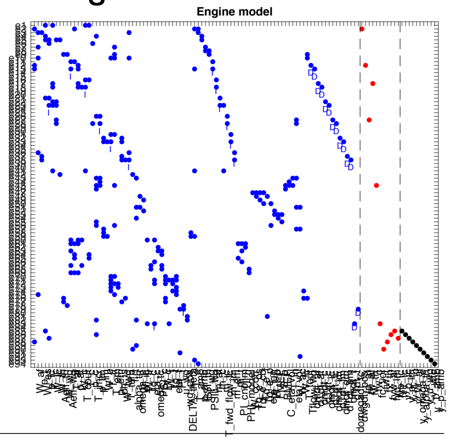
$$e_1 : V = i R(1 + f_R) + L\frac{di}{dt} + K_a i\,\omega$$

	Unknown variables														
	i	θ	ω	α	T	T_m	T_l	f_R	f_i	f_ω	f_T	V	y_i	y_ω	y_T
e_1	X		X					X				X			

- Coarse model description, no parameters or analytical expressions
- Can be obtained early in design process with little engineering effort
- Large-scale model analysis possible using graph theoretical tools
- Very useful!

Structural representation of engine model

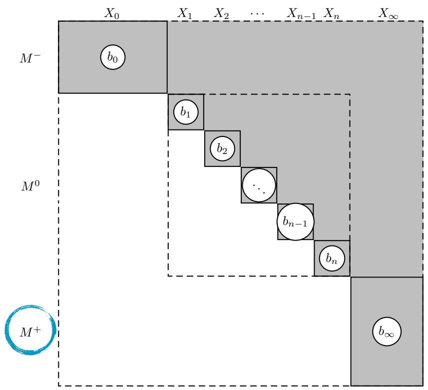




- Incidence matrix of variable dependency graph
  - Edges represents connections
  - I/D-edges correspond to differentiation and integration

Fundamental algorithmic tool: Dulmage-Mendelsohn decomposition

- Basic tool in many structural analysis algorithms
- Smart reordering or rows(equations) and columns (variables)
- Partitions the model into three parts
  - Under determined
  - Exactly determined
  - Over determined
- The overdetermined part with redundancy is the one interesting for diagnosis



Outline of the talk

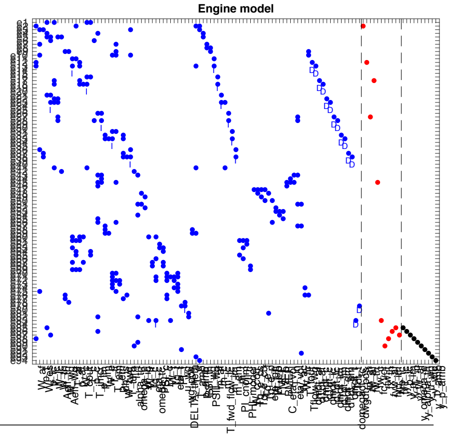
1. Diagnosability and sensor placement analysis
2. Testable (sub-)models and detector synthesis
3. A Modelica perspective
4. An automotive use-case

Presentation will be more what than how

Diagnosability analysis and sensor selection

Diagnosability analysis - Problem formulation

- Given a dynamic model
  - Q1: Which faults are structurally detectable?
  - Q2: What are the structural isolability properties of the model?



Structurally detectable and isolable faults

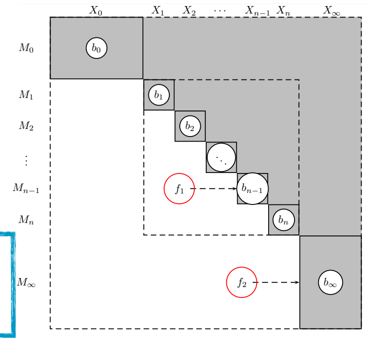
- Let e_{f_i} be the equation that is affected by fault f_i
- A fault f_i is (structurally) detectable iff

  $$e_{f_i} \in M^+$$

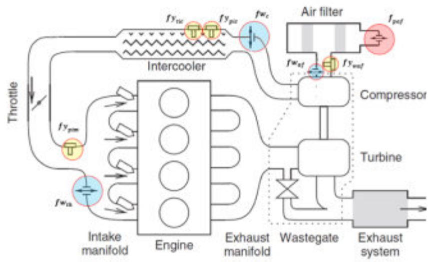
  - Fault f_1 not detectable, f_2 is detectable
- A fault f_i is isolable form a fault f_j iff

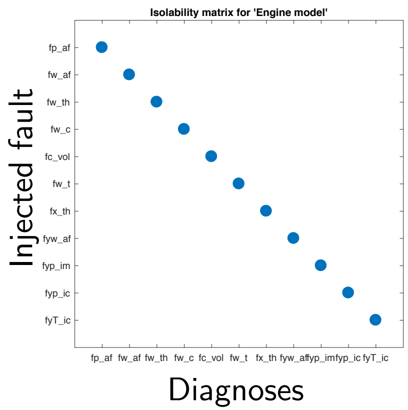
  $$e_{f_i} \in (M \backslash e_{f_j})^+$$

- Take home: Structural diagnosability can be determined by a series of Dulmage-Mendelsohn decompositions (fast)

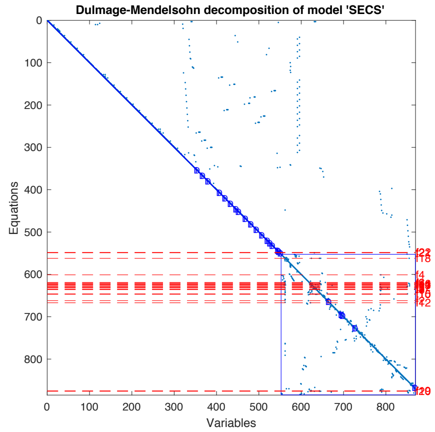


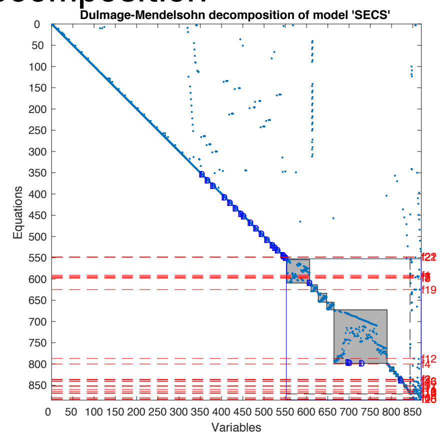
Diagnosability of an engine model



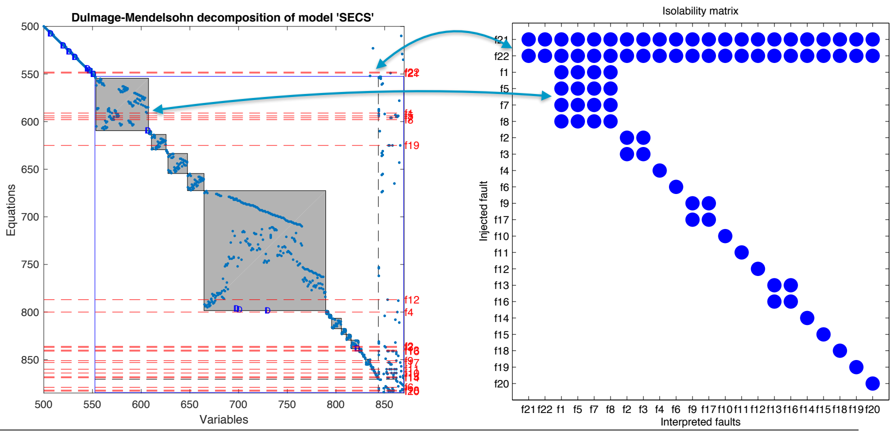


A more detailed structure decomposition





Isolability and a more detailed structure decomposition



Minimal sensor sets and problem formulation

Given:

- A set $\mathcal{P}$ of possible sensor locations
- A detectability and isolability performance specification

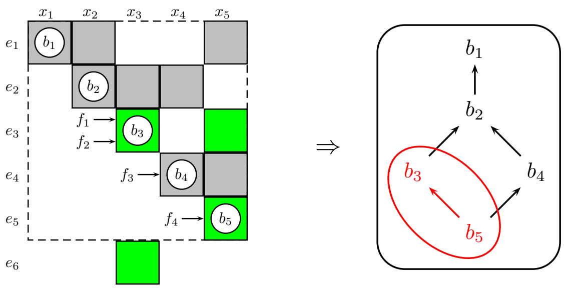
Minimal Sensor Set

A multiset S, defined on $\mathcal{P}$, is a minimal sensor set if the specification is fulfilled when the sensors in S are added, but not fulfilled when any proper subset is added.

Problem Statement

Find all minimal sensor sets with respect to a required isolability specification and possible sensor locations for any linear differential-algebraic model

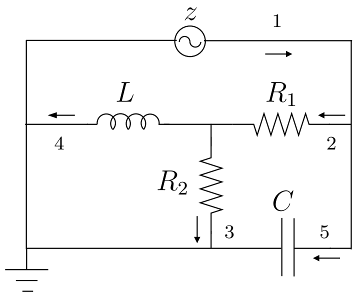
Basic principle for finding sensors



1. A Dulmage-Mendelsohn decomposition (again)
2. Define a partial order on equations that explicitly identifies variables to measure

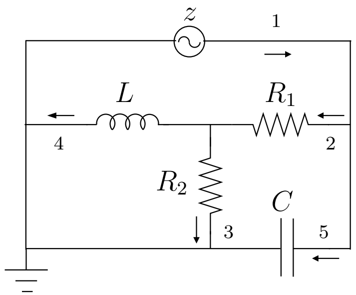
Example: An electrical circuit

A small electrical circuit with 5 components that may fail



Example: An electrical circuit

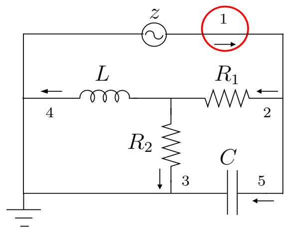
A small electrical circuit with 5 components that may fail



$$
\begin{aligned}
v_1 &= v_5 & v_5 &= v_2 + v_3 \\
i_1 &= i_2 + i_5 & i_1 &= i_3 + i_4 + i_5 \\
v_1 &= z & v_2 &= R_1 i_2 \\
v_4 &= L\frac{d}{dt} i_4 & i_5 &= C\frac{d}{dt} v_5 \\
v_3 &= v_4 & v_3 &= R_2 i_3
\end{aligned}
$$

- 10 equations, 2 states, 5 faults, 1 known signal
- Possible measurements: currents and voltages

Examples of results of the analysis



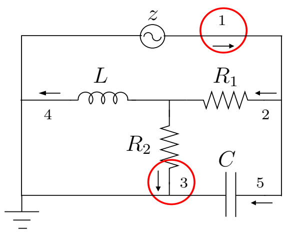
Example run 1

Objective	Achieve detectability
Possible measurement	voltages and currents

7 minimal solutions

$\{i_1\}, \{i_2, i_5\}, \{i_3, i_5\}, \{i_4, i_5\}, \{i_5, v_2\}, \{i_5, v_3\}, \{i_5, v_4\}$

Examples of results of the analysis



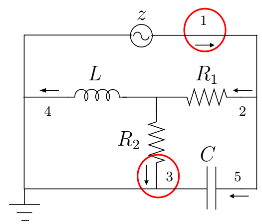
Example run 2

Objective	Achieve full isolability
Possible measurement	voltages and currents

5 minimal solutions

$\{i_1, i_3\}, \{i_1, i_4\}, \{i_2, i_3, i_5\}, \{i_2, i_4, i_5\}, \{i_3, i_4, i_5\}$

Examples of results of the analysis



Example run 3

Objective	Achieve full isolability, new sensors may fail
Possible measurement	voltages and currents

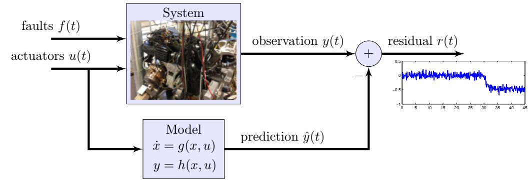
7 minimal solutions

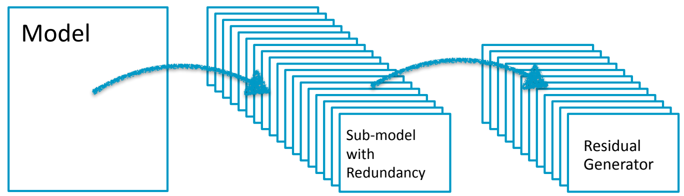
$\{i_1, i_1, i_3\}, \{i_1, i_1, i_4\}, \{i_1, i_3, i_5\}, \{i_1, i_4, i_5\},$

$\{i_2, i_3, i_5, i_5\}, \{i_2, i_4, i_5, i_5\}, \{i_3, i_4, i_5, i_5\}$

Testable sub-models and detector synthesis

Model based diagnosis, basic ideas





Basic principle - systematic utilization of redundancy

1 equation, 1 unknown, no redundancy

$$x = g(u)$$

Basic principle - systematic utilization of redundancy

2 equations, 1 unknown, 1 residual generator

$$x = g(u) \qquad r_1 = y_1 - g(u)$$
$$y_1 = x$$

Basic principle - systematic utilization of redundancy

3 equations, 1 unknown, 3 residual generators

$$x = g(u) \qquad r_1 = y_1 - g(u)$$
$$y_1 = x \qquad r_2 = y_2 - g(u)$$
$$y_2 = x \qquad r_3 = y_2 - y_1$$

Basic principle - systematic utilization of redundancy

4 equations, 1 unknown, 6 (minimal) residual generators

$$x = g(u) \qquad r_1 = y_1 - g(u)$$
$$y_1 = x \qquad r_2 = y_2 - g(u)$$
$$y_2 = x \qquad r_3 = y_2 - y_1$$
$$y_3 = x \qquad r_4 = y_3 - g(u)$$
$$r_5 = y_3 - y_1$$
$$r_6 = y_3 - y_2$$

Basic principle - systematic utilization of redundancy

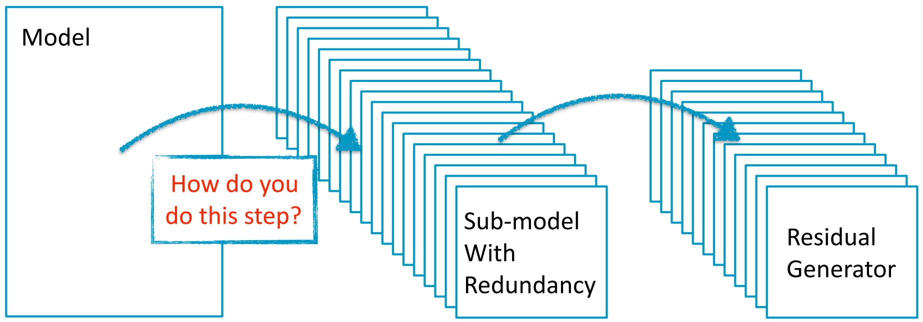
4 equations, 1 unknown, 6 (minimal) residual generators

$$x = g(u) \qquad r_1 = y_1 - g(u)$$
$$y_1 = x \qquad r_2 = y_2 - g(u)$$
$$y_2 = x \qquad r_3 = y_2 - y_1$$
$$y_3 = x \qquad r_4 = y_3 - g(u)$$
$$r_5 = y_3 - y_1$$
$$r_6 = y_3 - y_2$$

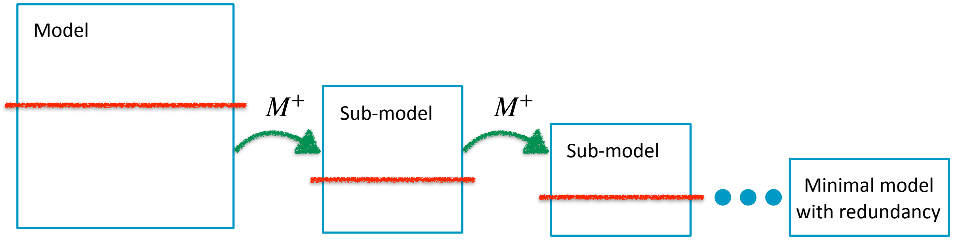
Answer:
Very much so, but careful analysis of DAE equations and their properties is essential

- Number of possibilities grows exponentially (here combinations)
- Not just $y - \hat{y}$
- Is this illustration relevant for more general cases?

Basic approach to diagnosis system design



Sketch – how to find all minimal models with redundancy



- Do this systematically; efficient way to find all MSO sets — Minimal Structurally Overdetermined set of equations
- A series of Dulmage-Mendelsohn operations — efficient
- Exponential in model redundancy — extensions for MSO exists to reduce solution set

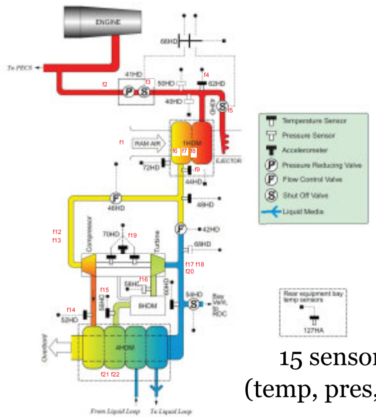
A Modelica perspective

Is Modelica a good language for this kind of analysis

- Started two masters theses together with Saab, Linköping
- Demonstrate automatic transformation of Modelica models into a format where existing fault diagnosis techniques are applicable.
- Describe how to make non-trivial diagnosis analysis for non-trivial Modelica models.

MODELICA

Use-case: Environmental Cooling System for Gripen Aircraft

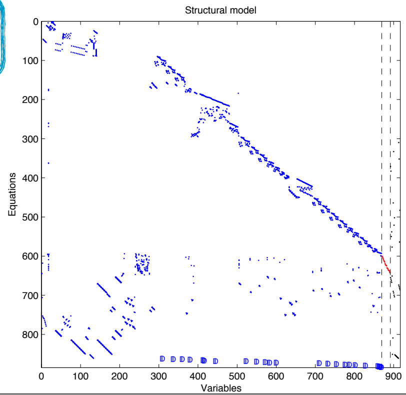


- Modelica model
- Uses standard libraries
- 1,000 - 10,000 equations

15 sensors
(temp, pres, rot)

Modelica and structural analysis

- Flat Modelica model is well suited for structural analysis
- Structural analysis requires non-repeated expressions
- Connecting components give, by construction, non-repeated expressions if the model is not simplified.
- State-space forms are typically not suitable



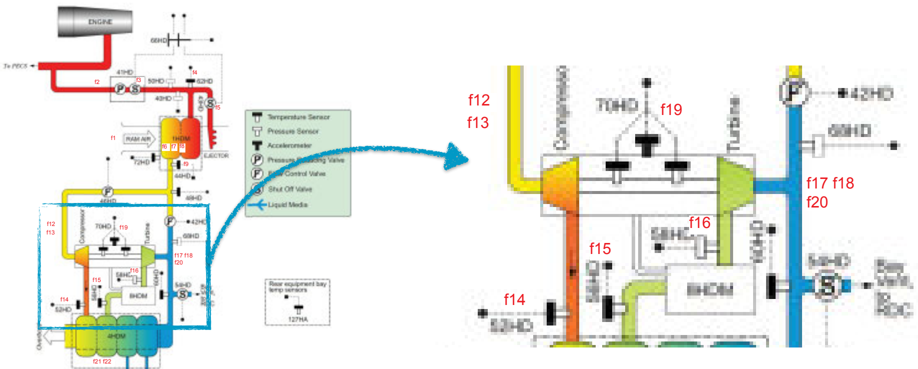
Toolchain

Modelica → XML representation → XML to FDT-format parser → Analysis in Matlab

$$y = x + 1$$

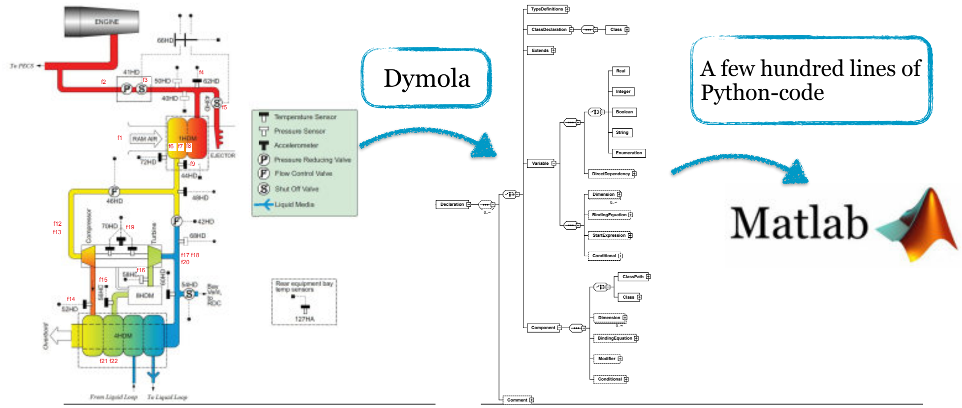
```
<SimpleEquation>
  <LHS>
    <ComponentReference>
      <Reference
        instanceName="y"/>
    </ComponentReference>
  </LHS>
  <RHS>
    <Binary Operator="+">
      <Left>
        <ComponentReference>
          <Reference
            instanceName="x"/>
        </ComponentReference>
      </Left>
      <Right>
        <Literal Value="1"/>
      </Right>
    </Binary>
  </RHS>
</SimpleEquation>
```


Modelica model with faults



Straightforward to extend existing components with fault models

Transformation: .mo → XML → .m



Conditionals

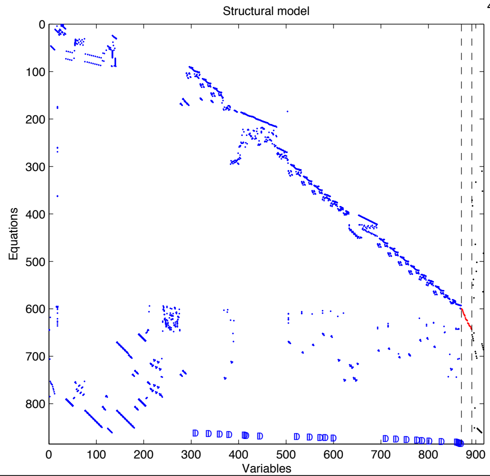
- Very common with conditionals in more complex models
- Common in the ECS model here
- Related to hybrid/switched systems
- Here a simple, and pragmatic approach

if $c(x_c) > 0$ **then**
  $x_1 = g_1(x_2)$;
else
  $x_1 = g_2(x_2, x_3)$;
end if;

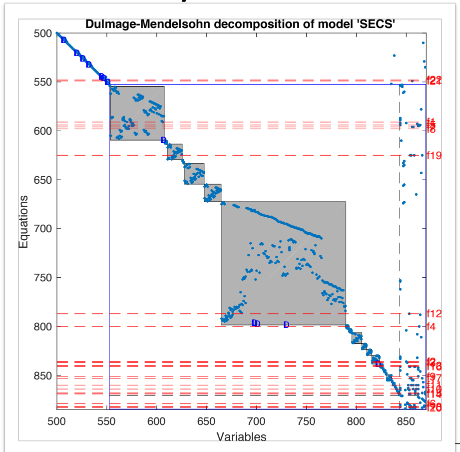
$$\begin{aligned} x_1 &= \mathrm{if}(c(x_c) > 0; g_1(x_2); g_2(x_2, x_3)) \\ &= f(x_c, x_2, x_3) \end{aligned}$$

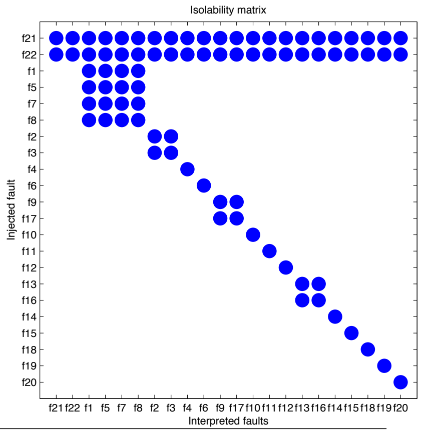
Model structure

- Gripen Environmental Cooling System model
  - 884 equations, 23 dynamic states
  - 22 faults
  - Degree of redundancy 15

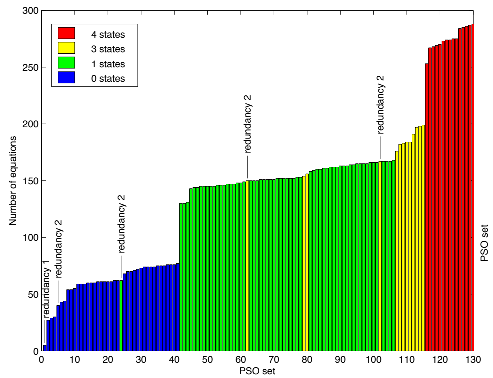


Isolability from structure

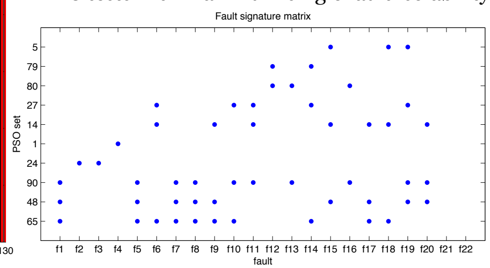




Residual generator properties



- Test selection to achieve maximum single fault isolability by a Greedy search:
- 1) Few states 2) Few equations
- 10 tests with maximum single fault isolability



Summary

- Straightforward to transfer Modelica models to analysis format (Matlab)
  - Here a limited part of the Modelica language
- Toolchain via XML representation (unfortunately, not standardized)

Modelica → XML representation → XML to FDT-format parser → Analysis in Matlab

- Possible to obtain non-trivial results
  - Fault detectability, fault isolability, fault detector analysis
- Saab developers also appreciate:
  - Gain additional insight in model structure
  - Find model weaknesses using fault diagnosis techniques

LINKÖPING UNIVERSITY

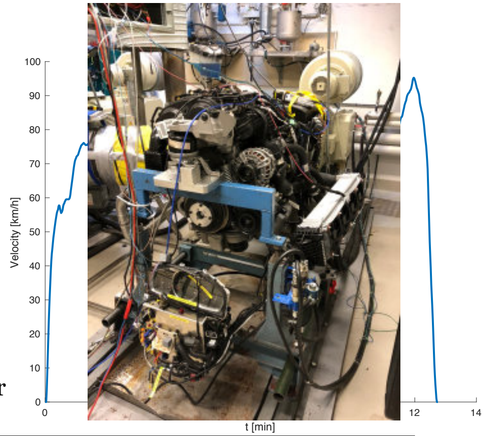
An automotive use-case

LINKÖPINGS UNIVERSITET

Volvo development control system
We can inject, physically and through the control system, many relevant faults

Test cell data

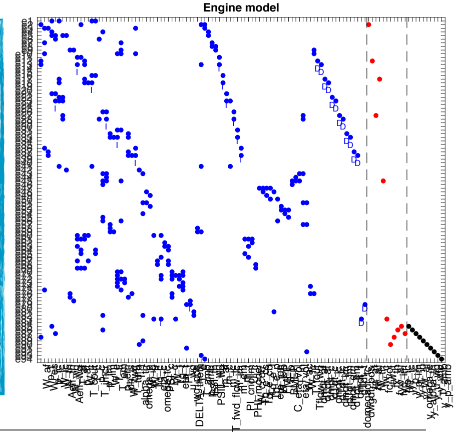
- Volvo production engine
- EPA HWFET cycle translated into load cycle for engine (rpm/torque)
- (here) 5 data sets:
  - 1. Fault free
  - Sensor faults
    - 2. Intake pressure
    - 3. Air-flow sensor
    - 4. Pressure after intercooler
    - 5. Temperature after intercooler



Structural model of the engine

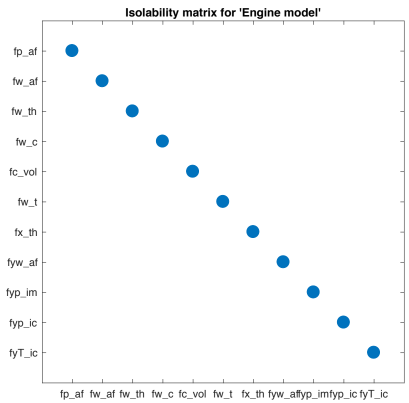
Matlab code

```
>> model

Model: Engine model

  Type: Symbolic, dynamic

  Variables and equations
    90 unknown variables
    10 known variables
    11 fault variables
    94 equations, including 14 differential constraints

  Degree of redundancy: 4
```



Isolability matrix for the engine

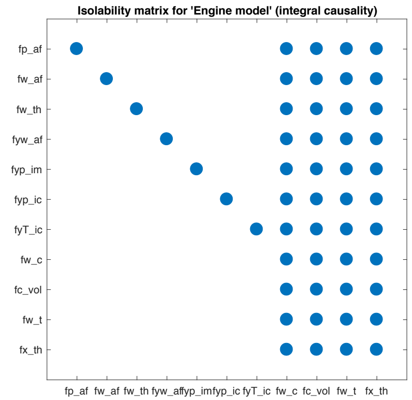
- Only redundancy 4 in the model
- All faults detectable
- a dot — fault fi not isolable from fj
- A diagonal is the ideal property
- Isolability matrix a simple summary of *single-fault* isolability properties
- Not a trivial result at all: with available sensors — all faults are (ideally) possible to isolate



Differential index and isolability analysis

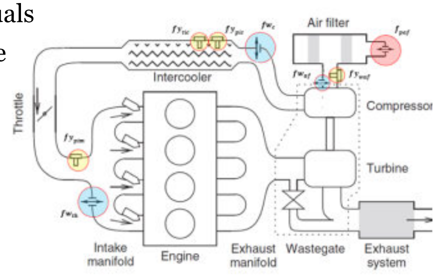
Question

What is possible with only observer techniques, i.e., no high index problems



Redundancy & testable sub-models in the engine model

- Redundancy 4 (4 output sensors)
- A $r = y - \hat{y}$ solution would give 4 residuals
- Due to the turbine feedback, many more possibilities exist
- In the model: 4496 MSO sets
  - 206 with low-index (4.6%)
- Choose wisely!



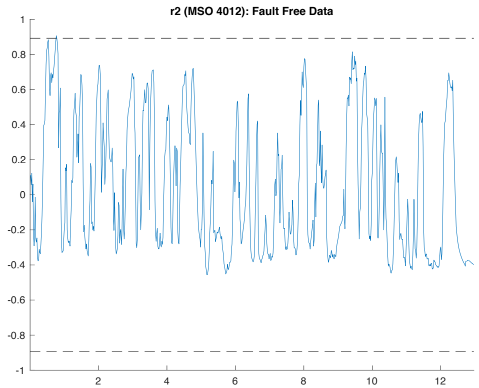
Test selection

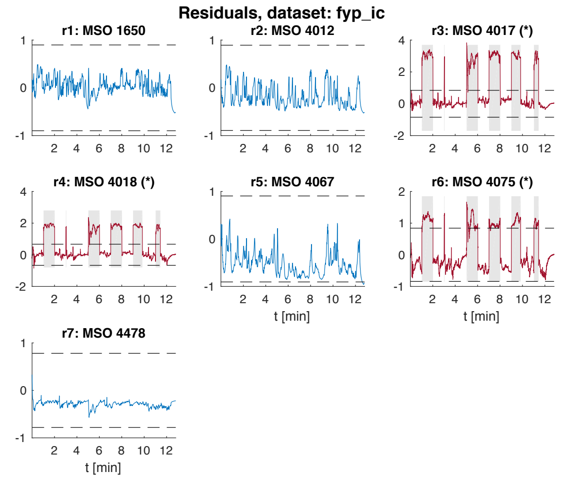
- Each MSO submodes with n equations, n possible residual generators
  - 4,496 MSO sets: 343,099 candidate residual generators
  - 206 low-index sets: 728 candidate residual generators (208 succeeded)
- Do not need many to isolate the faults ~ number of faults

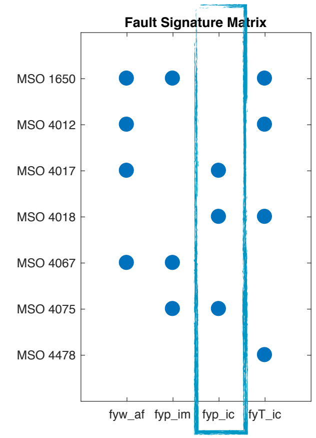
- If models were ideal all tests equally good
- Here: make test selection based performance on measured data
- C code essential for evaluation, Matlab versions just too slow
- Simple approach, based on Kullback-Leibler divergences. Restriction to 4 sensor faults gives 7 selected residuals.

Running residual generators

- Sampling rate 1 kHz
- Data set 12 minutes with 10 measurement signals
- Execution takes about 0.5 sec on this computer ($\approx$ 1400 $\times$ real-time)
- Simple thresholding based on false-alarm rate on no-fault data

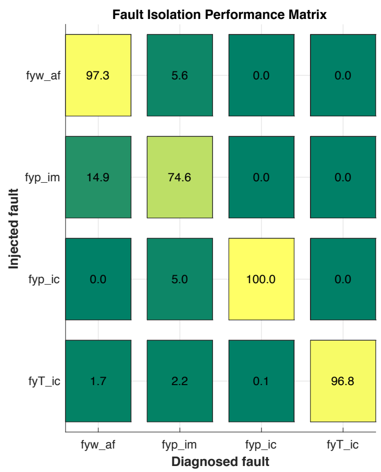






Isolation performance

- Performance measure
  $P(f_i \text{ in diagnosis} \mid f_j)$
- Ideally diagonal
- This non-tuned solution works well
- Some difficulty isolating a fault in the intake manifold pressure sensor (fyp_im) from a fault in the air-mass flow sensor (fyw_af)



Quick look back at the design

- Automated (or close to)
  - Modelling (structural and analytical)
  - Analysis of diagnosability and simulation properties
  - Test selection
  - Code generation
- No tuning; the designed residuals are nowhere near optimal
- Gives a candidate solution; suitable for an engineer to fine-tune
- Important that code is readable, understandable. Equation based models help here.

A Summary and Some Takeaways

- DAE:s are inherent in consistency based diagnosis of dynamic systems
- Graph theoretical tools very useful for diagnosis analysis — can be implemented in “general purpose” computer support tools
  - Core operation — graph algorithms; efficient for large models
- Analysis of Modelica models demonstrated via XML export
  - Extract as raw model as possible after flattening
  - Manipulation might affect structural results
- Proven useful for industrial examples — automotive production engine, Gas Turbine Engines for power production

Erik Frisk
(erik.frisk@liu.se)

www.liu.se