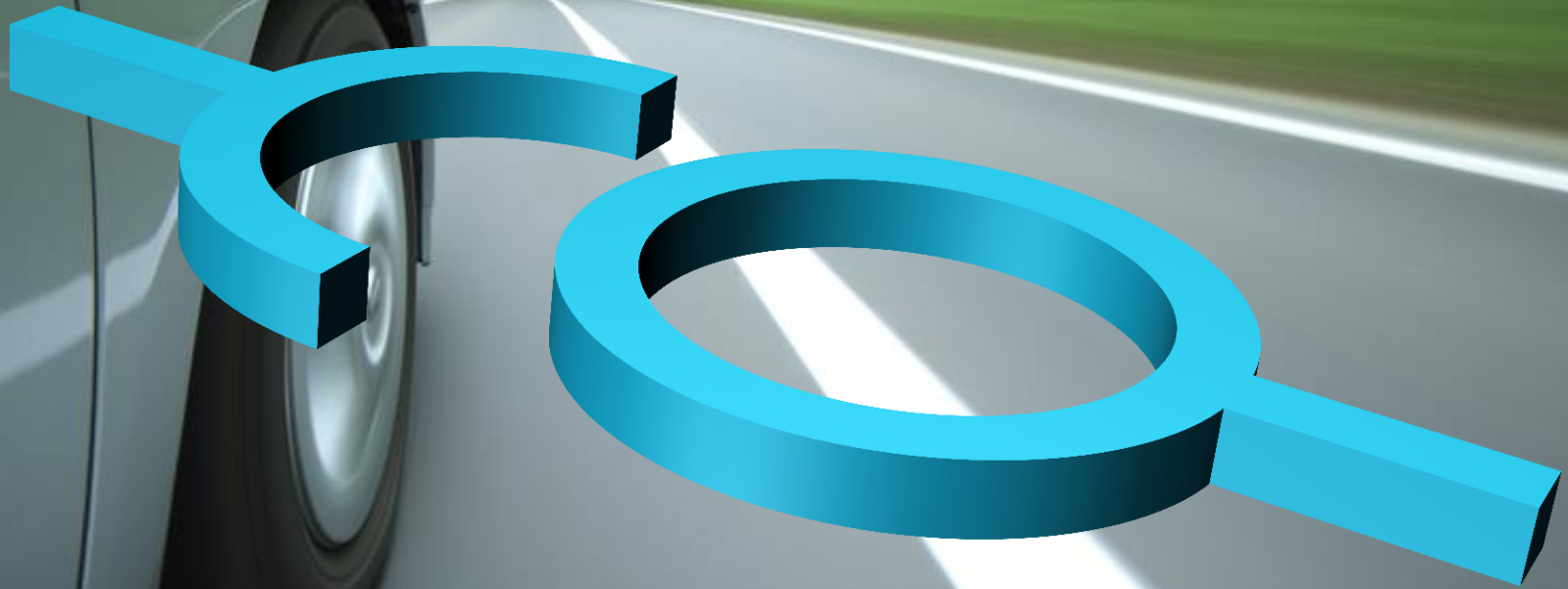


WHAT IS FMI AND WHAT CAN IT DO FOR YOU?

Maria Henningsson

Dept of Automatic Control April 22 2015

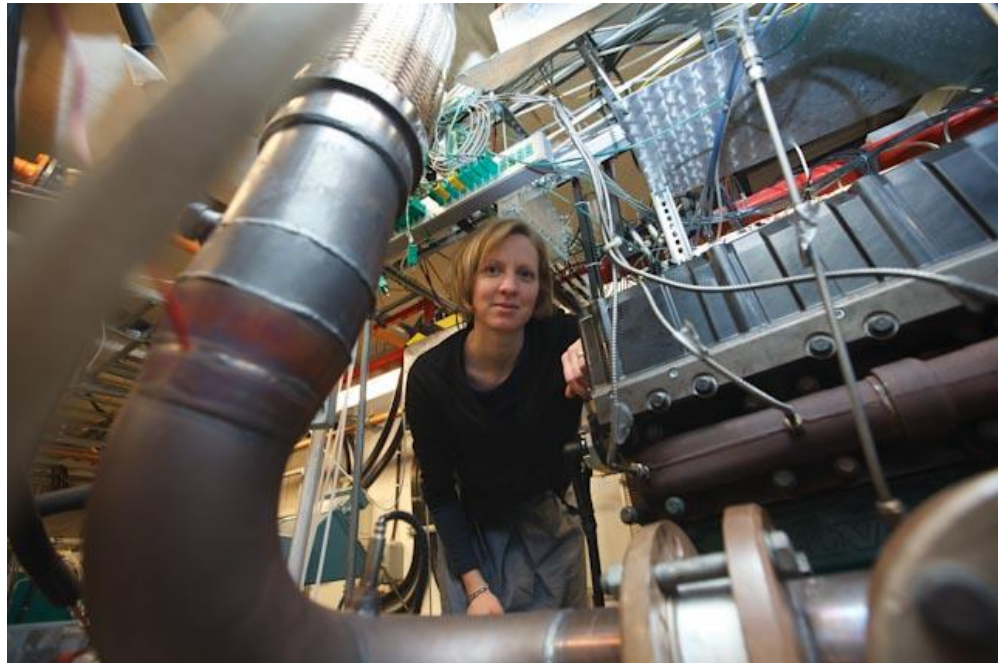


OUTLINE

- Why simulation models?
- Why FMI?
- What is FMI?
- FMI for control design: use case

About me

- PhD #92: Data-Rich Multivariable Control of Heavy-Duty Engines
- At Modelon since 2012
- Product manager for Modelon's FMI products



About Modelon

MODELON

Founded in Lund in 2005 by Hubertus Tummescheit, Jonas Eborn, Magnus Gäfvert, and Johan Andreasson

Head office at Ideon, Lund

Currently employs 8 PhDs from Automatic Control, LTH

Core business:

Simulation models and tools



MODELON LOCATIONS

Modelon Inc.

- Ann Arbor, MI
- Hartford, CT

Modelon AB

- Lund
- Gothenburg

Modelon GmbH

- Munich
- Stuttgart
- Hamburg

Modelon KK

- Tokyo

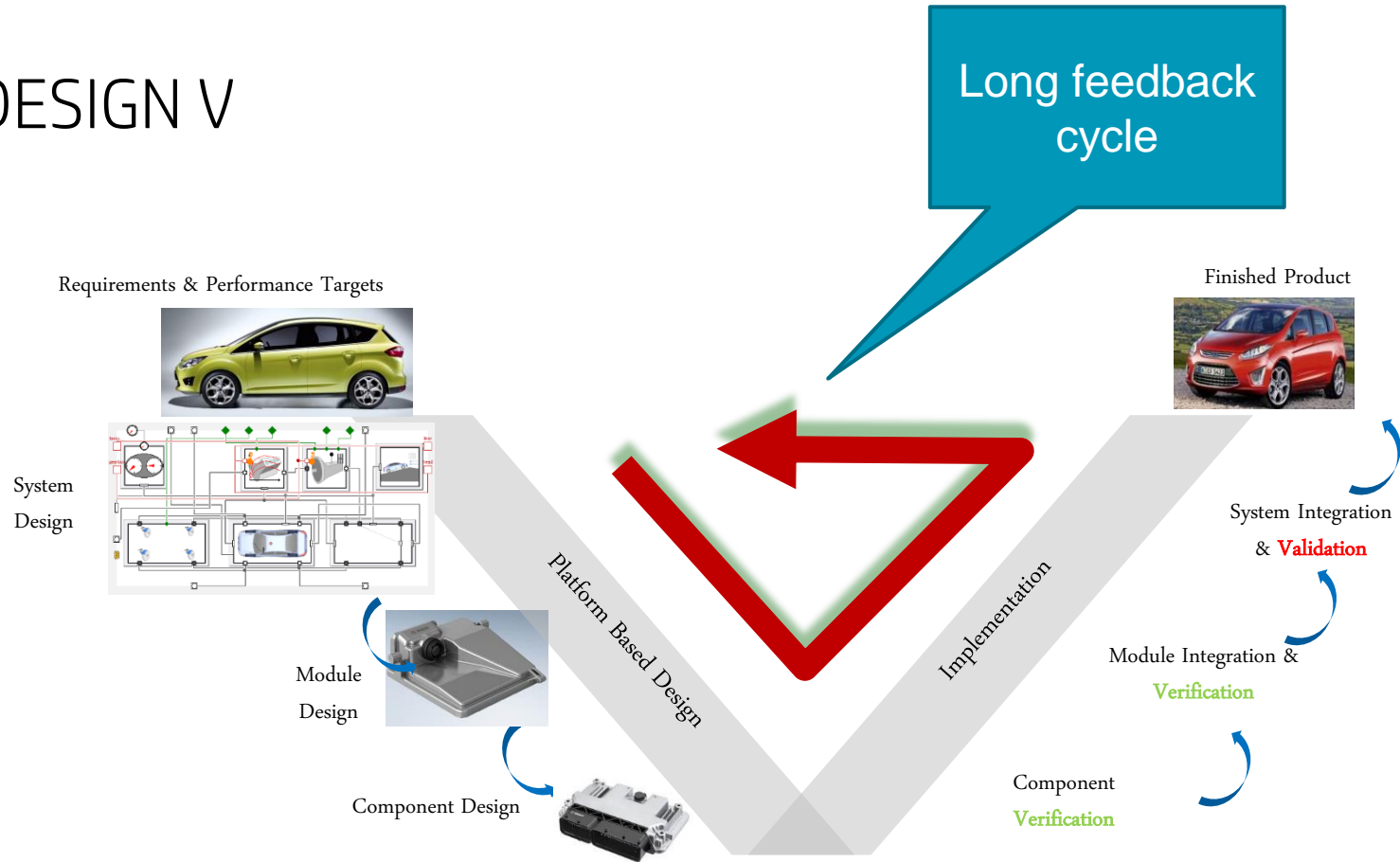
- ~90 engineers (MSc / PhD levels) dedicated to Modelica and FMI
- Global customers mostly among Fortune 500 technology companies in Automotive, Aerospace, Energy and Industrial Equipment
- >30% annual growth

Why simulation models?

Why simulation models?

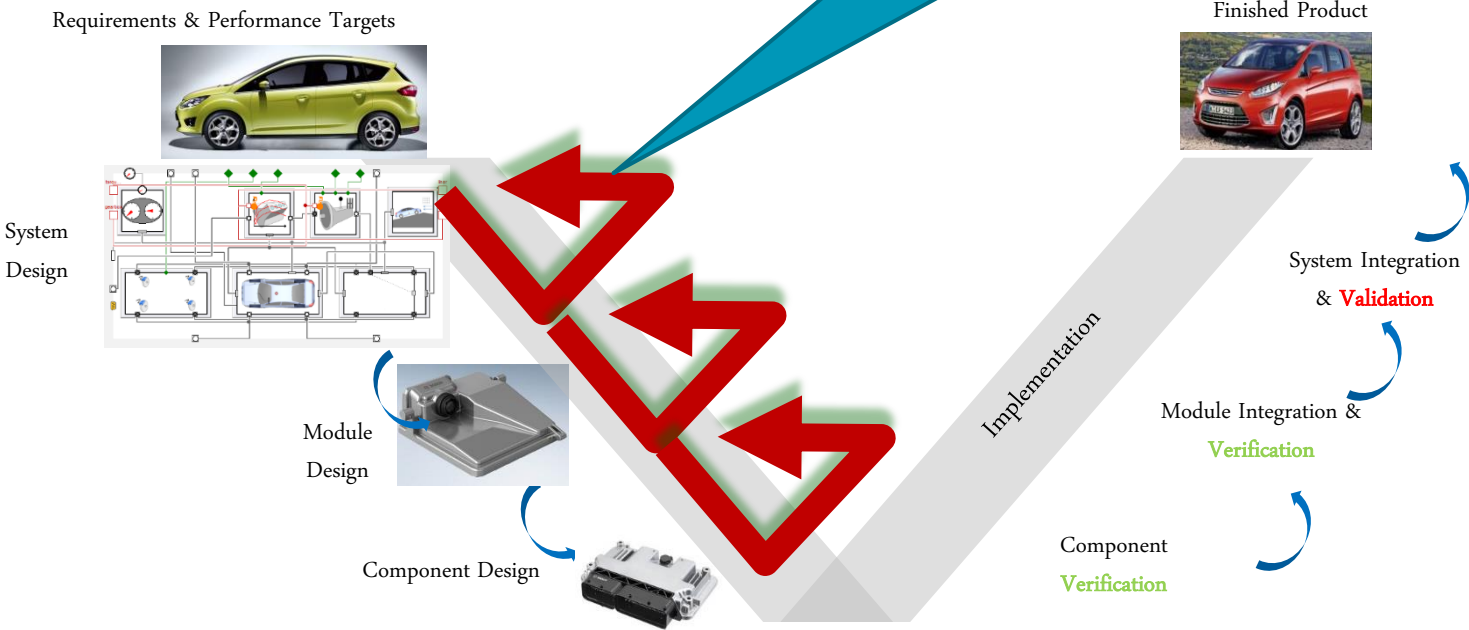
Industry perspective...

DESIGN V

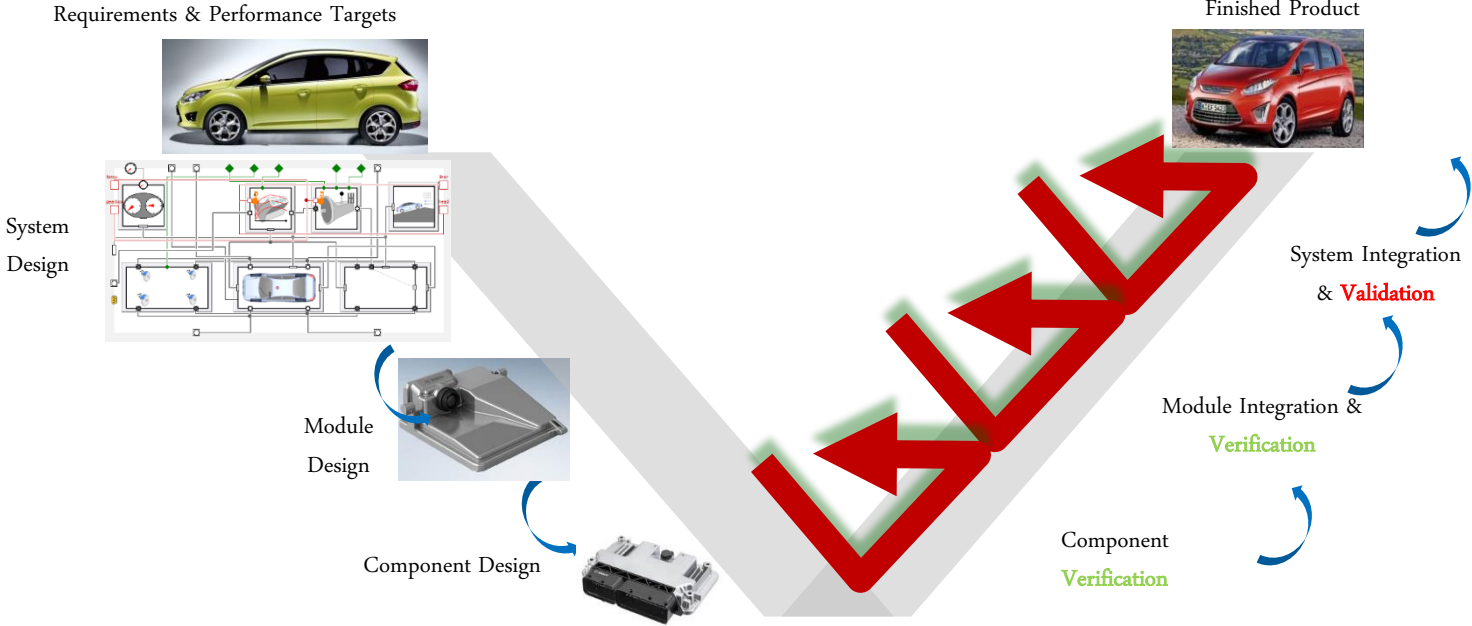


CONTINUOUS VALIDATION AND VERIFICATION

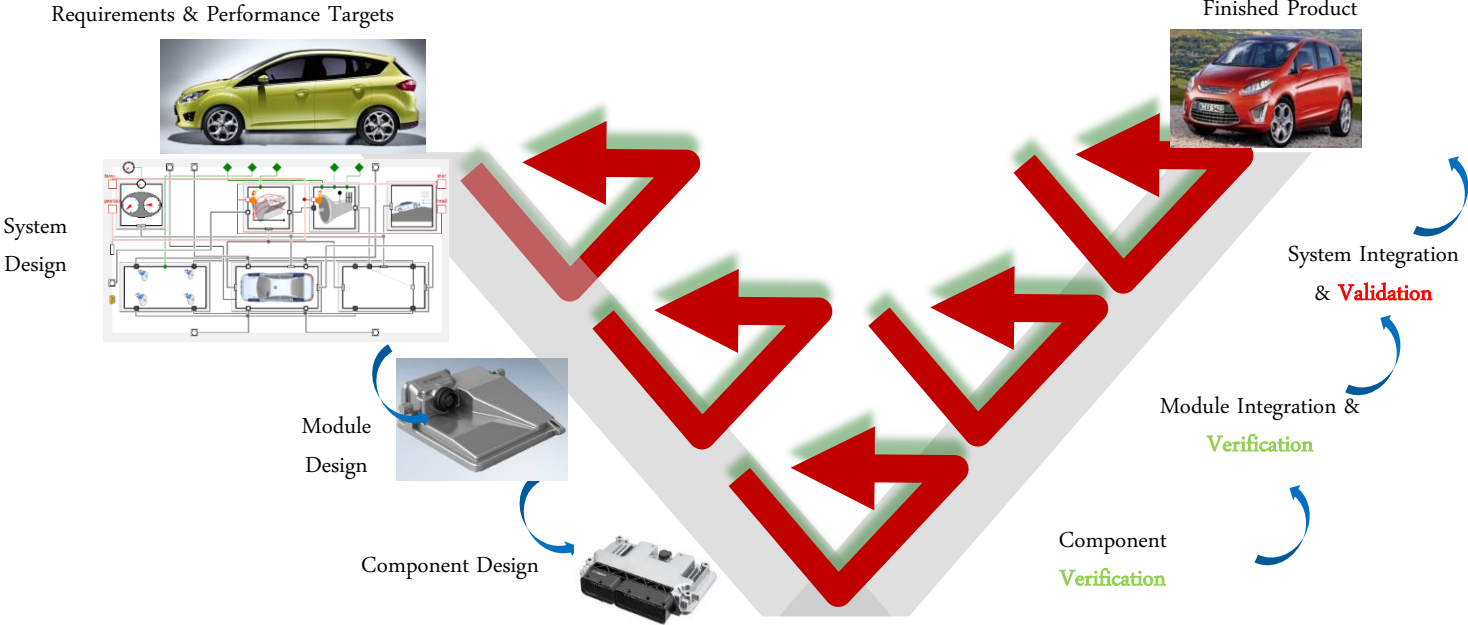
Continuous feedback on requirements through virtual tests



CONTINUOUS INTEGRATION WITH SIL, HIL, SIMULATORS



SHORTEN THE FEEDBACK CYCLES!



Why simulation models?

Your perspective...

Modelon

APPLIED CONTROL

APPLIED CONTROL

- Waiting for the test rig to work in the lab?

APPLIED CONTROL

- Waiting for the test rig to work in the lab?
- Waiting for data or experiment results from an industrial partner?

APPLIED CONTROL

- Waiting for the test rig to work in the lab?
- Waiting for data or experiment results from an industrial partner?
- Waiting for another EU project work package to deliver a model?

CONTROL THEORY

CONTROL THEORY

- Want to be sure that your research is relevant?

CONTROL THEORY

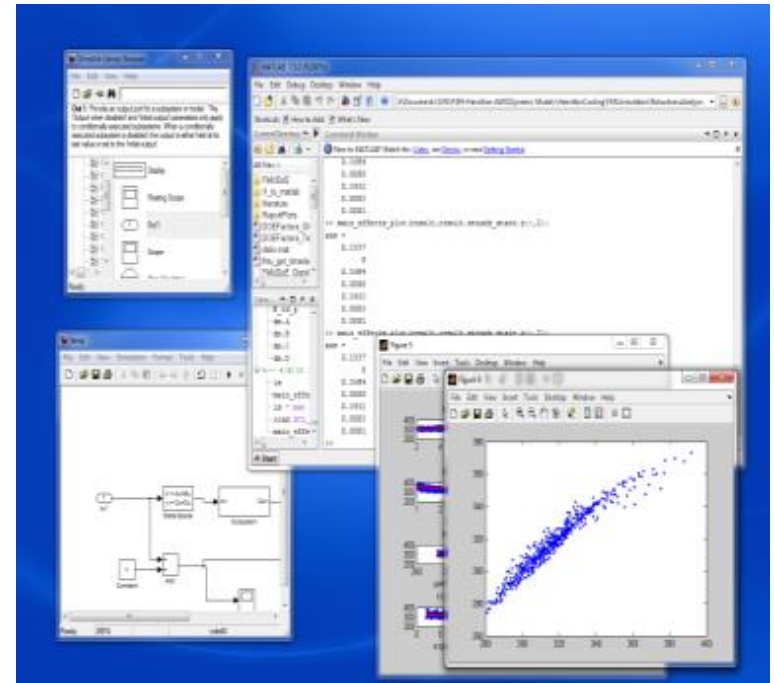
- Want to be sure that your research is relevant?
- Do you need to show that your research is relevant?

CONTROL THEORY

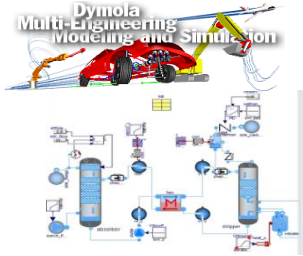
- Want to be sure that your research is relevant?
- Do you need to show that your research is relevant?
- Want to prepare for a future industry career?

MODELS FOR CONTROL DESIGN

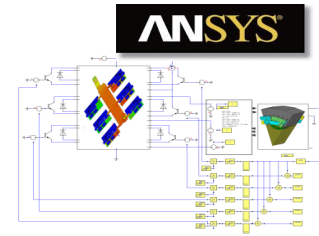
- Simple models
- Linearizations
- Understanding dominating parameter effects
- Understanding system variability
- Identify worst cases



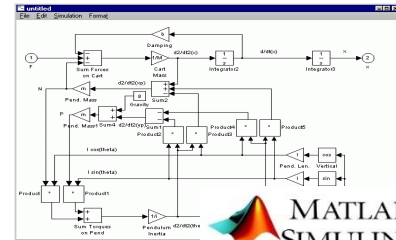
SIMULATION TOOL LANDSCAPE



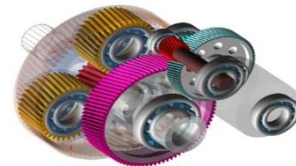
DASSAULT SYSTEMES



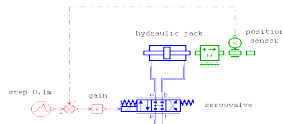
Requires models for
C/C++
memory



Wolfram SystemModeler



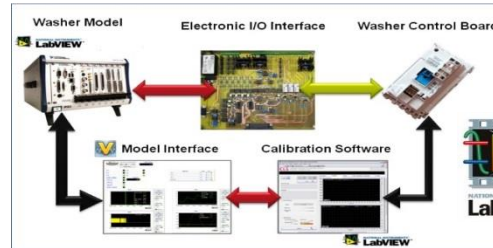
OpenModelica



LMS
A Siemens Business



SIMULATION X
Powered by ITI



Adams



MSC Software

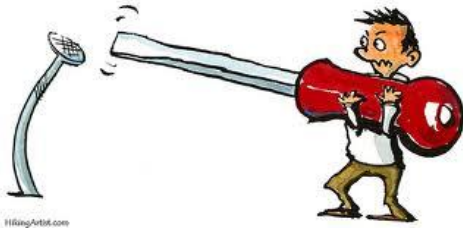


WHAT'S STOPPING YOU?

- What are the best-of-breed simulation tools for your application domain?
- Do you use them?
- Why not?

Why FMI?

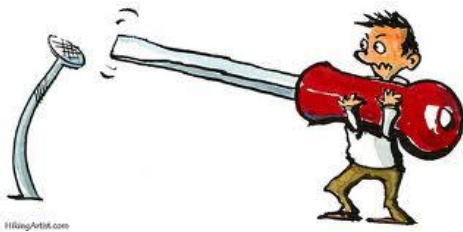
CAE TOOLS



1st generation

CAE TOOLS

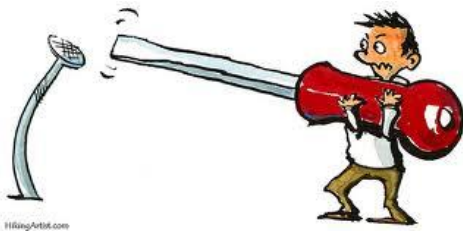
1st generation



2nd generation



CAE TOOLS



1st generation



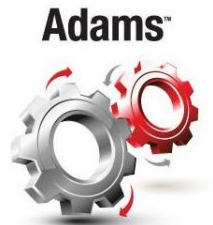
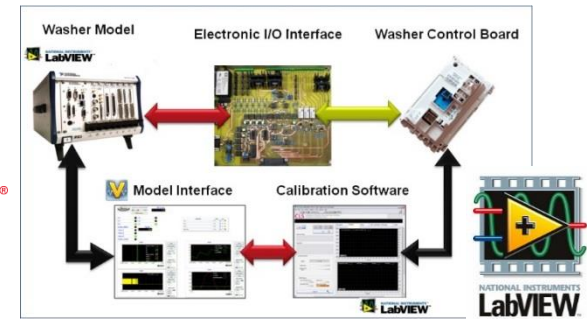
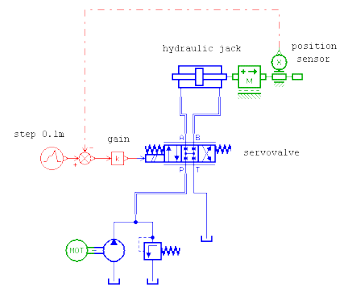
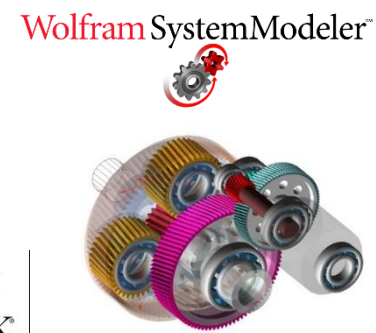
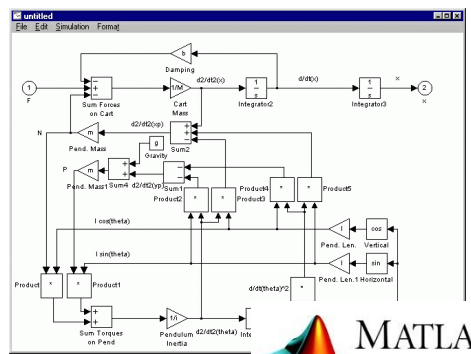
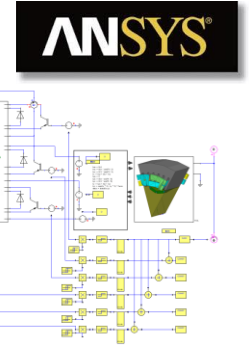
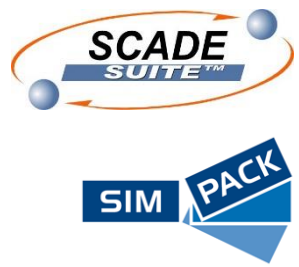
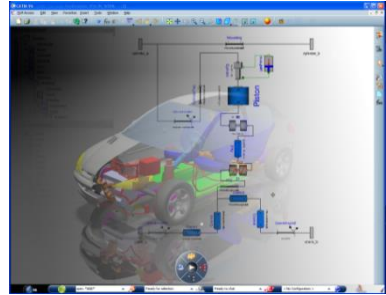
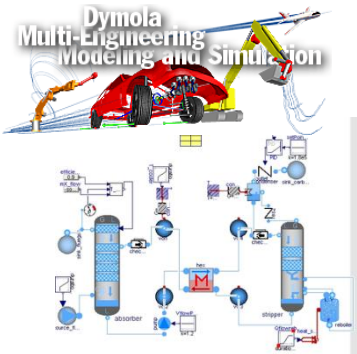
3rd generation?



2nd generation

- Multiple physical domains
- Multiple vendors
- In-house solutions
- Legacy tools
- Individual/group preferences
- Suppliers
- Partners
- Customers

ALL CONNECTED?



What is FMI?

WHAT IS FMI?

FMI – Functional Mock-up Interface

- Open interface standard for model exchange between different modeling and simulation environments.

- Consists of
 - Model Interface: Set of C functions for equation evaluation.
 - Model Description Schema: XML Schema defining an XML file containing variable definitions and model meta data.

- Developed as a part of the MODELISAR project
 - ITEA2 European project to improve the design of systems and embedded software in vehicles.
 - <http://www.fmi-standard.org/>

WHAT IS AN FMU?

FMU – Functional Mock-up Unit

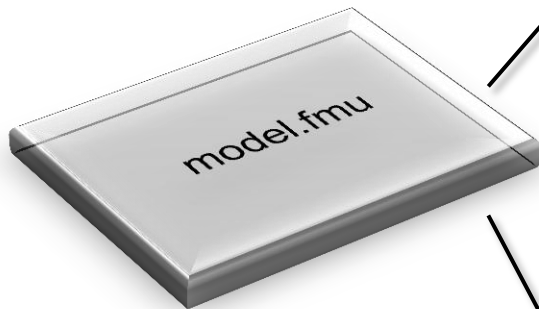
- Implementation of the FMI.
- Zip-file with a specific structure consisting of
 - Binary file (usually a DLL-file)
 - XML Model Description file.
 - Additional model data (images, documentation).
- Co-Simulation or Model Exchange
- Simulator calls FMI functions to create and run the FMU.



Name	Type
binaries	File folder
modelDescription	XML Document

FMU DISTRIBUTION

An FMU is distributed in a zip-file (with the extension .fmu) which contains:



- The Model Description File (modelDescription.xml). Includes model information (e.g. variables) that is not needed for simulation.
- The binaries and/or C sources of the Model Interface (including the needed run-time libraries used in the model and/or Dynamic link libraries (DLL) for one or several target machines).
- Additional model data (like tables, maps) in model specific file formats.

FMI FLAVORS

Model Exchange (ME)



Co-Simulation (CS)

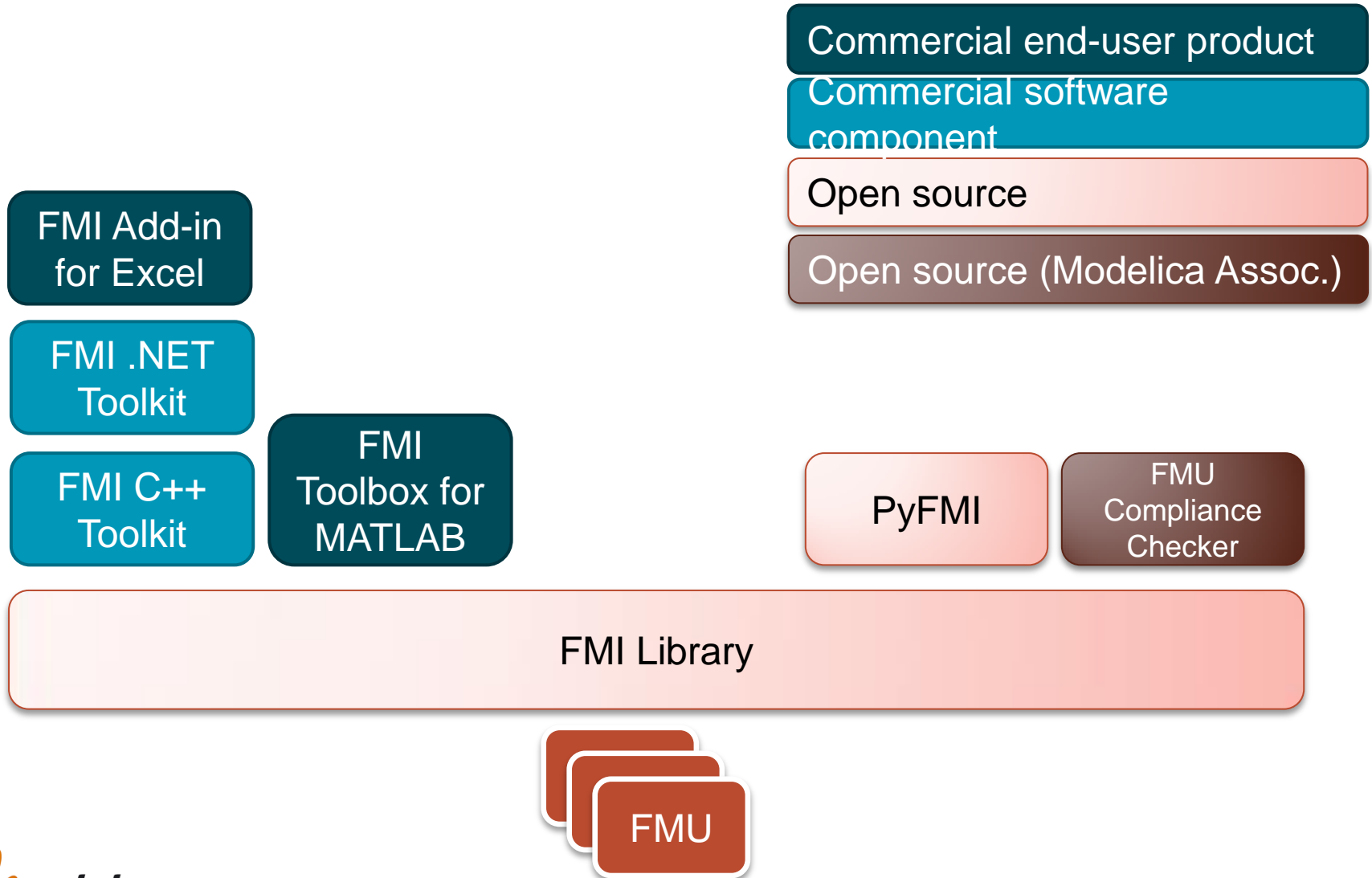


FMI at Modelon

MODELON AND FMI

- FMI adopted as core technology and business
- Active engagement
 - Standard development
 - Quality and compliance
 - Tool development
- FMI backbone in customer solutions and toolchain integration
- Partnerships for FMI toolchain integration
 - Multiple global software vendors
- Multiple research projects
 - Europe and USA

MODELON FMI SOFTWARE STACK



MODELON OFFERINGS

Products

FMI toolbox for
MATLAB®



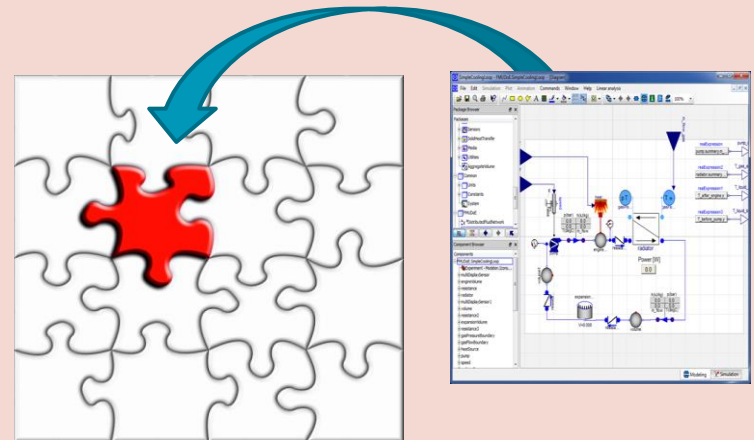
FMI Add-in for
Excel



PyFMI
Python
interface

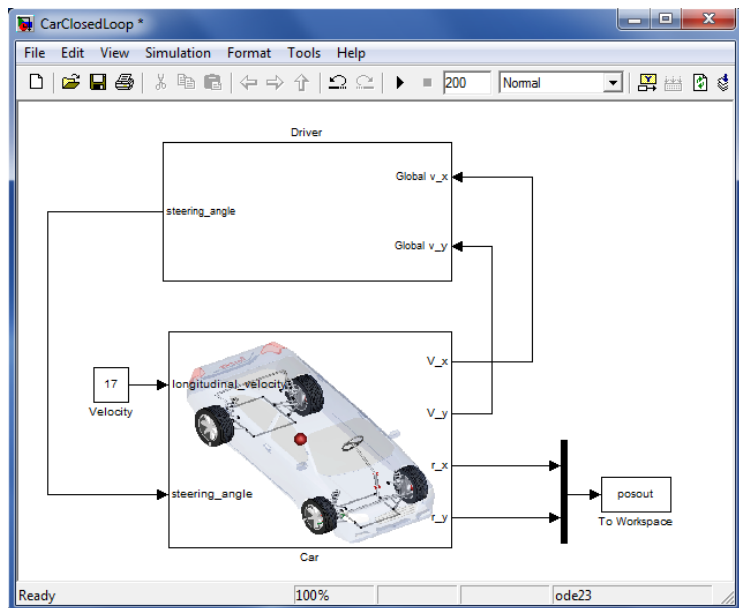
Customized solutions

- Fit Modelica/FMI models into customer's toolchains
- Based on Modelon FMI products
- Create an interface to models that matches customer's terminology and needs



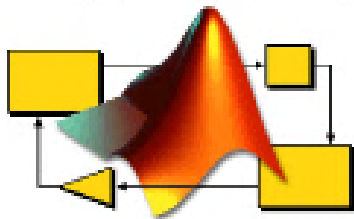
FMI TOOLBOX

for MATLAB®



- FMI technology in MATLAB
- Import FMUs (ME/CS) in Simulink
- Import FMUs (ME/CS) in MATLAB scripts
- Graphical interface for model configuration
- FMU-ME (ME/CS) export from Simulink models

MathWorks



Partner

2013-06-05

MODELON BECOMES A PRODUCT PARTNER OF THE MATHWORKS CONNECTIONS PROGRAM

FMI Toolbox strengthens MATLAB® and Simulink® as open integration platform in model-based systems engineering.

Lund, Sweden – (June 10, 2013) Modelon, a leading provider of open-standard experience solutions for next generation model-based system engineering, today announced it has become a member of the MathWorks Connections Program. This membership is based on the FMI Toolbox, an extension that brings MATLAB and Simulink into the center of the simulation software ecosystem formed by the Functional Mock-Up Interface (FMI) open standard for tool interoperability and model exchange.

The MathWorks Connections Program is available to third-party organizations that develop and distribute complementary, commercially available products and services based on [MATLAB](#) and [Simulink](#). These partner offerings address technical needs across a wide range of applications and industries worldwide with software and hardware products that extend the usage of MATLAB and Simulink. These solutions seamlessly integrate with MathWorks products and ensure ongoing compatibility in conjunction with new MathWorks releases.

"We are very pleased to be part of The MathWorks Connections Program," said Magnus Gäfvert, CEO with Modelon AB. "This is an opportunity that will enable us to align our solutions with one of the most widely used simulation and embedded function development software solutions in the world. It will allow us to leverage our open solution platform in a wide and extended range of application and industry contexts, and thereby saving time and reducing cost and risk for our customers in their systems engineering project challenges".

[FMI Toolbox](#) provides systems, control and simulation engineers with a solution for seamless exchange of simulation models, or Functional Mock-Up Units (FMUs), between any FMI compliant tool and MATLAB and Simulink, based on a proven implementation of the [FMI standard](#). It allows users to combine physical and functional models from different sources for integration on system level to assess and evaluate design choices and performance. The command line interface and the Simulink blockset offer users full flexibility to exploit the simulation and computational power of MATLAB and Simulink to create insight into the interactions and behavior of complex systems.

<http://www.modelon.com/news/>

The logo for Modelon, featuring the word "Modelon" in a stylized, italicized font. The "M" is large and orange, while the rest of the letters are black.

FMI for control design: use case

From Modelica Conference 2014:

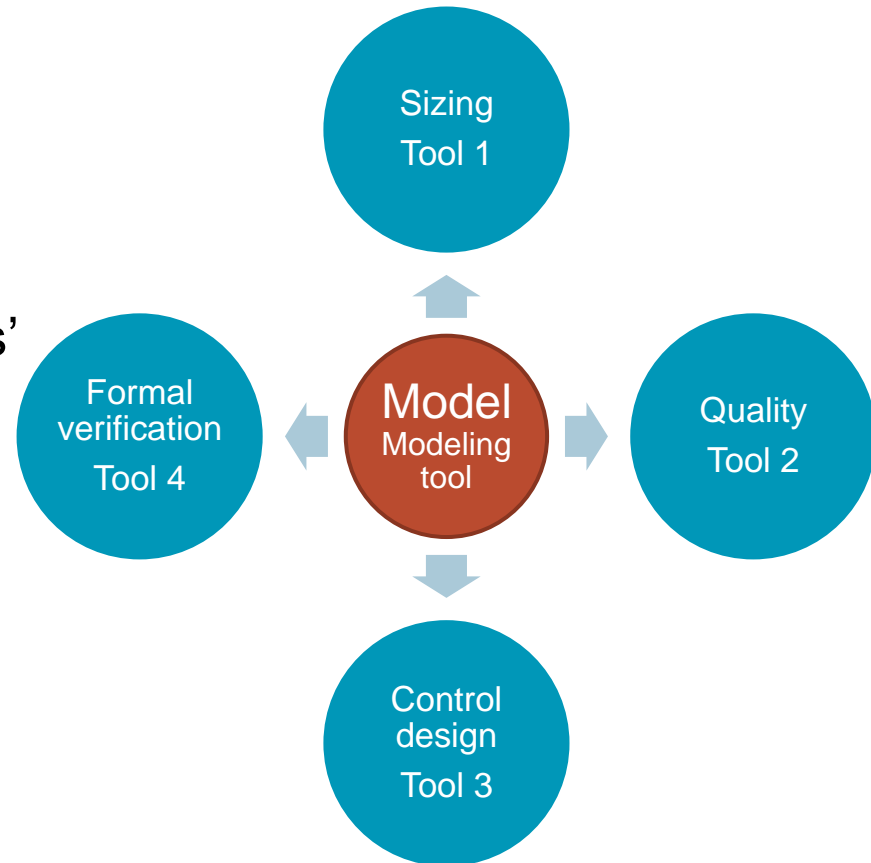
An FMI-Based Tool for Robust Design of Dynamical Systems

Maria Henningsson, Johan Åkesson, Hubertus Tummescheit

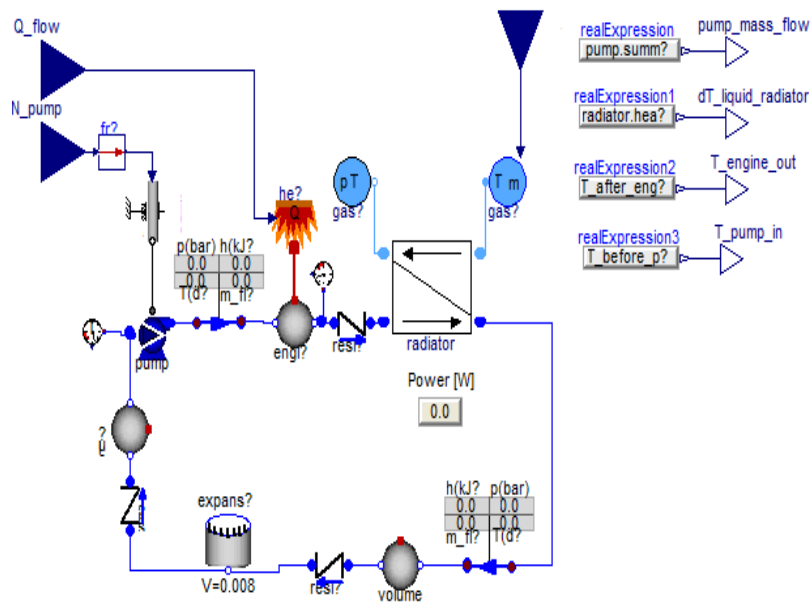
Modelon AB / Modelon Inc.

THE POWER OF FMI

- Can separate modeling and answering questions to different tools
- Use the best tool for each purpose
- Take advantage of engineers' skills in specific tools
- Facilitate cross-team use of the same model portfolio

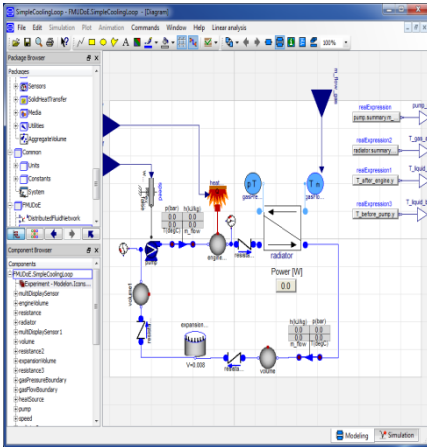


EXAMPLE: ENGINE COOLING SYSTEM



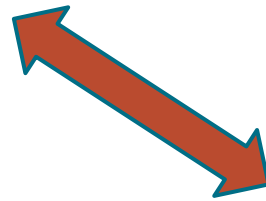
- Demo model from Modelon's Liquid Cooling Library
- 9 states
- Design variables:
 - Maximum pump speed
 - Radiator efficiency
 - Minimum air mass flow
- Requirements:
 - Engine-out coolant temp < 100 °C
 - Handle heat load of 100 kW
 - Ambient temperature operating range $[-20$ °C, 45 °C]
- Control engine-out coolant temperature by pump speed

HOW TO DO MODEL-BASED DESIGN?



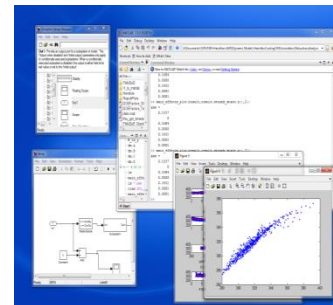
Modelica models

- Complex models
- Nonlinear
- Many parameters



Design and implementation of controllers in Matlab / Simulink

- Prefer simpler models
- Linearizations
- Understanding dominating parameter effects
- Understanding system variability
- Identify worst cases



TWO ACADEMIC PARADIGMS

Quality science/ Robust design / Six-sigma / Design-of-experiments

- Static models
- Many parameters
- Data-driven models
- Focus on workflows, processes and tools

Control engineering

- Dynamic models
- Few parameters
- Physics-based or data-driven models
- Focus on mathematical rigor

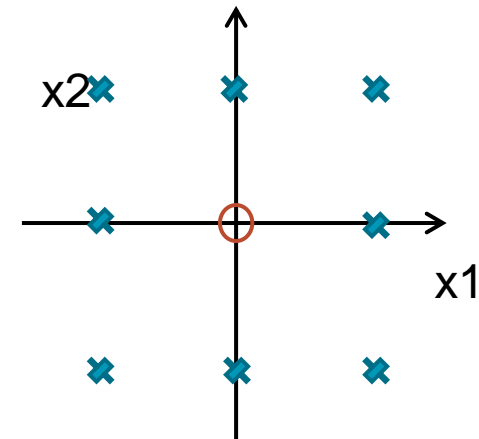
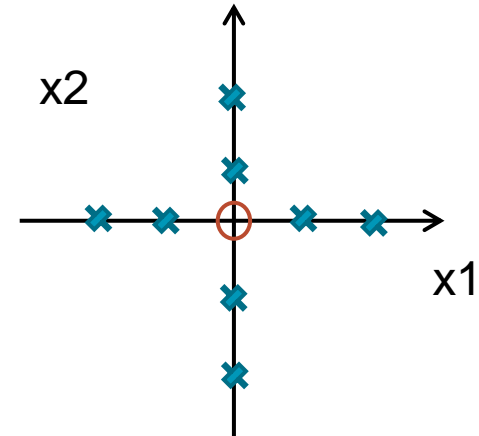
- Large potential in combining approaches
- Modelica and FMI is a suitable platform

DESIGN-OF-EXPERIMENTS (DOE)

- Run a limited set of experiments (or simulations)
- Identify crude models for how variables affect the behavior of a system in order to:
 - Optimize hardware design
 - Calibrate
 - Understand system variability

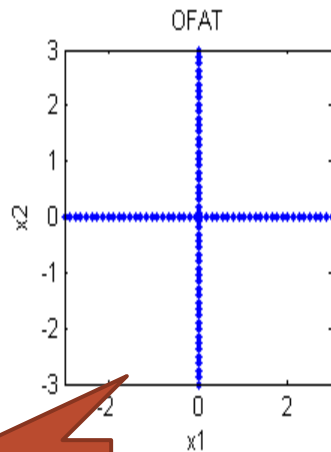
DOE: SIMPLISTIC APPROACHES

- One-factor-at-a-time (OFAT)
- Full-factorial design (gridding)

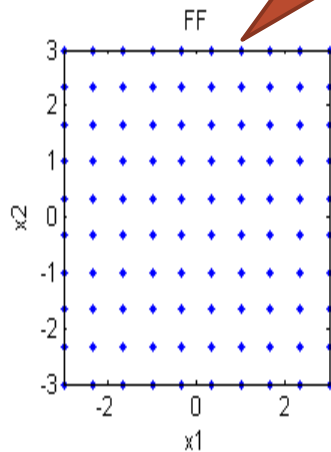


DOE DESIGNS

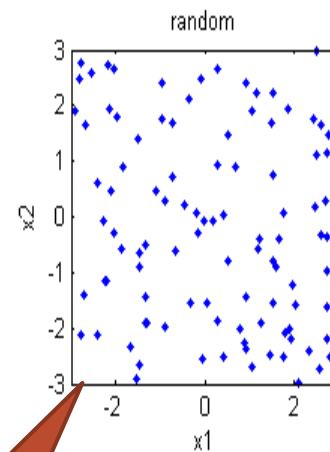
100 test points, two factors



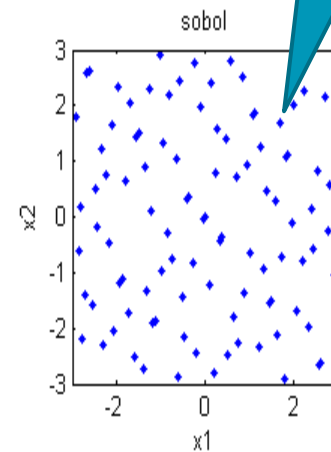
Common
ad-hoc
approach



Scales
poorly with
nbr of factors



cases poorly
covered in
higher



Space-filling
algorithms
Good
coverage,
also in higher
dimensions

A DOE TOOL FOR DYNAMIC SYSTEMS: REQUIREMENTS

User input

- FMU model
- DoE factors from model
- Ranges and distributions of factors
- Type of DoE design
- Response variables to analyze or visualize

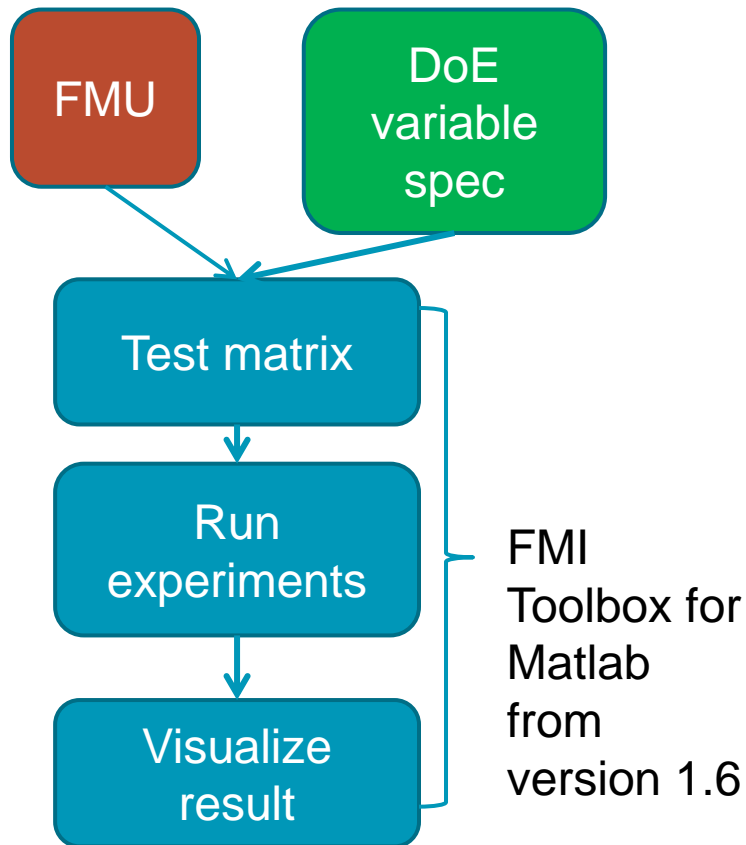
Tool tasks

- Construct test matrix
- Set FMU parameters
- Simulate at all points, find steady-state
- Find inputs to match specified outputs
- Catch and manage simulation errors
- Linearize system at test matrix points
- Provide support for visualizing results
- Construct meta-models for analysis

DOE IN THE FMI TOOLBOX FOR MATLAB

Modeling
tool

Excel /
Matlab



FMUDoESetup class

constructor input:

- FMU file name
- Excel file name
- [Excel sheet]
- [options]

methods:

- qmc
- mc
- fullfact
- custom

FMUDoEResult class

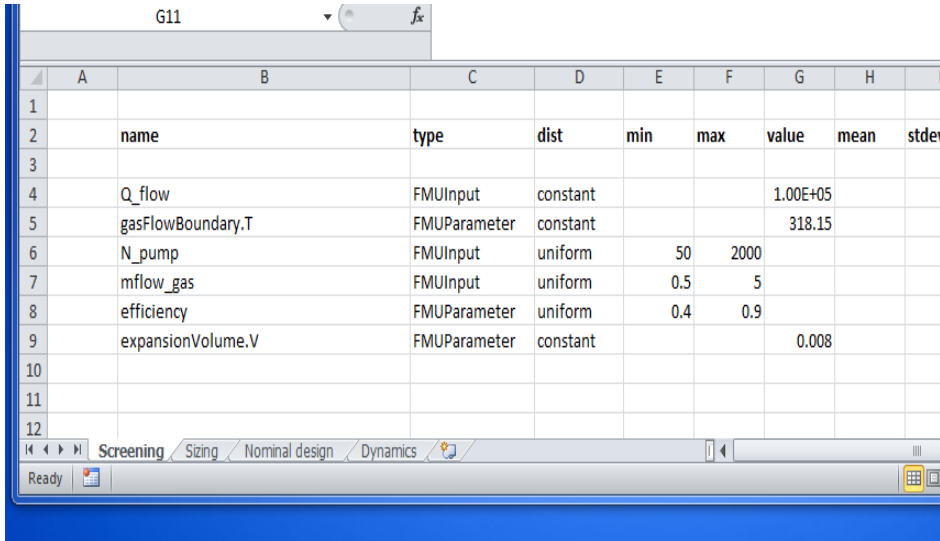
properties:

- generation_date
- model_data
- doe
- constants
- experiment_status
- steady_state
- linsys
- options
- comp_time

visualization methods:

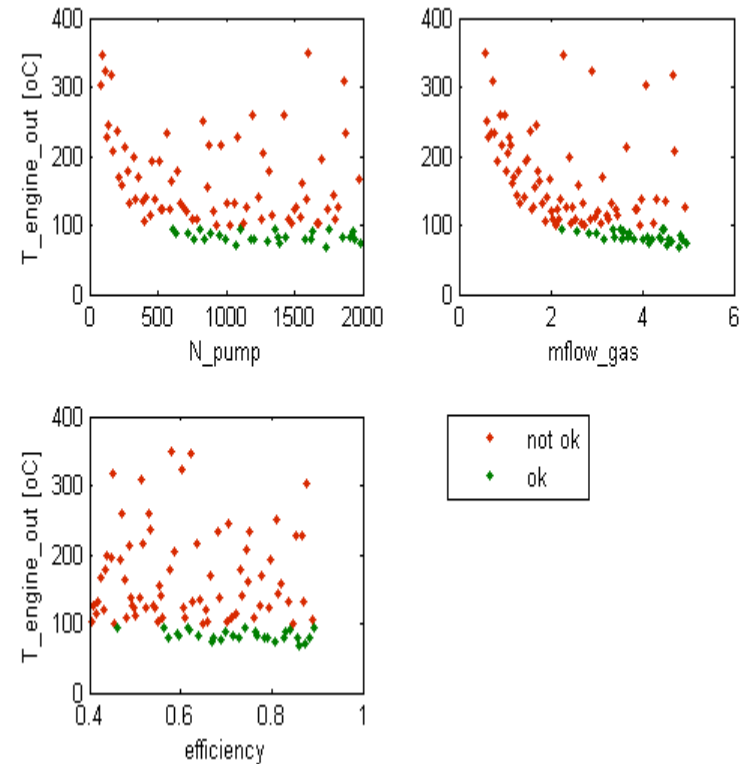
- main_effects
- bode
- step

SIZING: SCREENING DESIGN SPACE



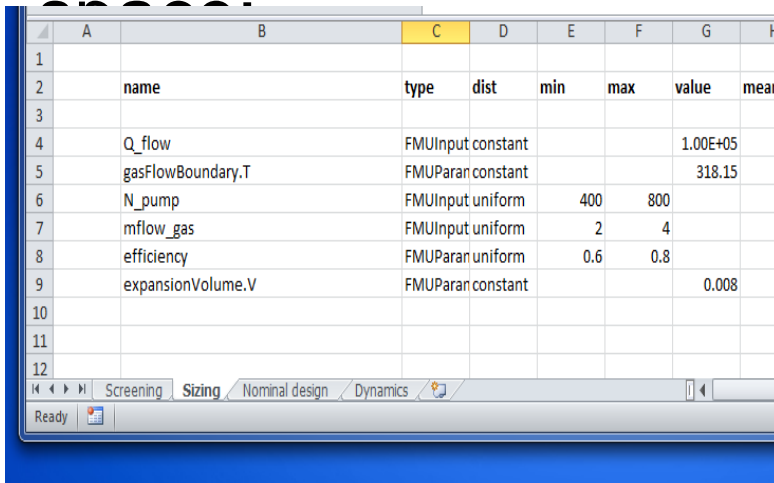
	A	B	C	D	E	F	G	H	I
1									
2		name	type	dist	min	max	value	mean	stdev
3									
4		Q_flow	FMUInput	constant			1.00E+05		
5		gasFlowBoundary.T	FMUParameter	constant			318.15		
6		N_pump	FMUInput	uniform	50	2000			
7		mflow_gas	FMUInput	uniform	0.5	5			
8		efficiency	FMUParameter	uniform	0.4	0.9			
9		expansionVolume.V	FMUParameter	constant			0.008		
10									
11									
12									

```
>> doe_setup = FMUdoESetup('CoolingLoop.fmu',  
'DesignParameters.xlsx', 'Screening');  
>> nbr_of_experiments = 100;  
>> result = doe_setup.qmc(nbr_of_experiments);  
>> T_engine_out = result.steady_state.y( : , 3)  
>> result.main_effects(result, 'T_liquid_ae', T_engine_out >  
100);
```



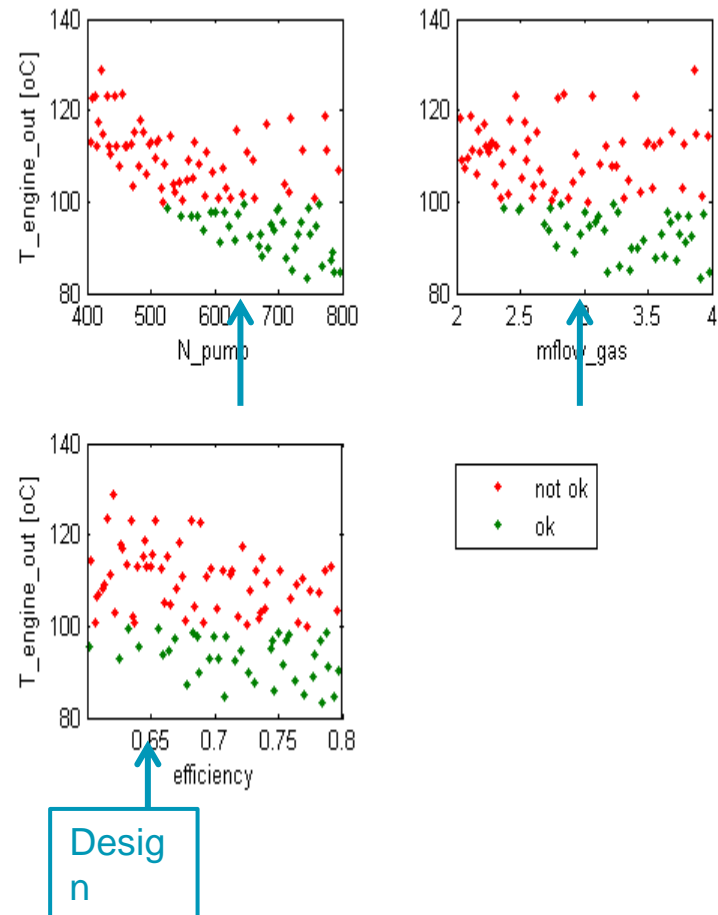
SIZING: DETERMINING A DESIGN

Zooming in to a smaller region of the design



	A	B	C	D	E	F	G	H
1								
2		name	type	dist	min	max	value	mean
3								
4		Q_flow	FMUInput	constant			1.00E+05	
5		gasFlowBoundary.T	FMUParan	constant			318.15	
6		N_pump	FMUInput	uniform	400	800		
7		mflow_gas	FMUInput	uniform		2	4	
8		efficiency	FMUParan	uniform	0.6	0.8		
9		expansionVolume.V	FMUParan	constant			0.008	
10								
11								
12								

```
>> doe_setup = FMUdoESetup('CoolingLoop.fmu',  
'DesignParameters.xlsx', 'Sizing');  
>> nbr_of_experiments = 100;  
>> result = doe_setup.qmc(nbr_of_experiments);  
>> T_engine_out = result.steady_state.y( : , 3)  
>> result.main_effects(result, 'T_liquid_ae', T_engine_out >  
100);
```

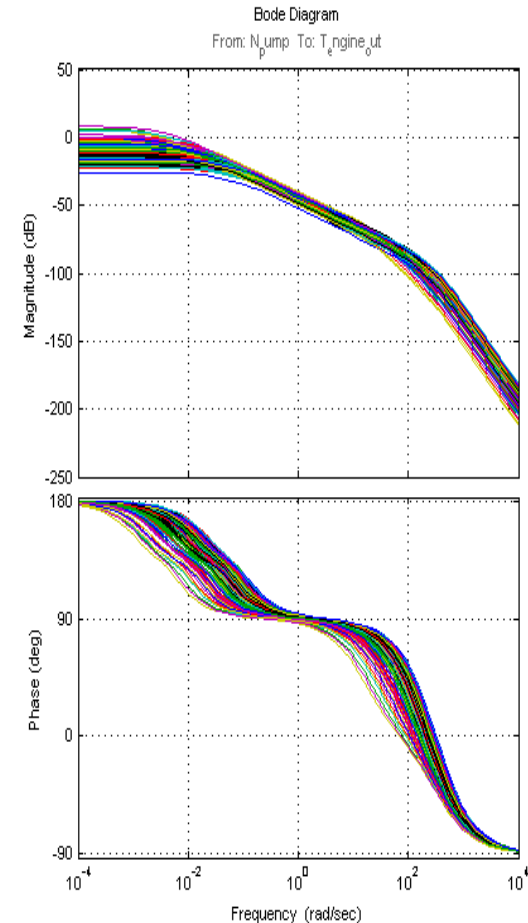


DYNAMICS

- input = pump speed
- output = engine-out coolant temperature

Bode plot for ensemble of linear systems

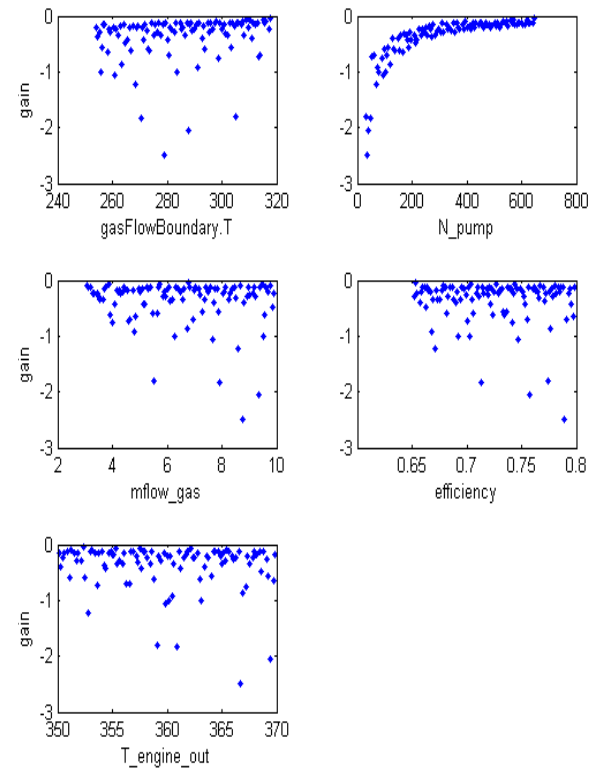
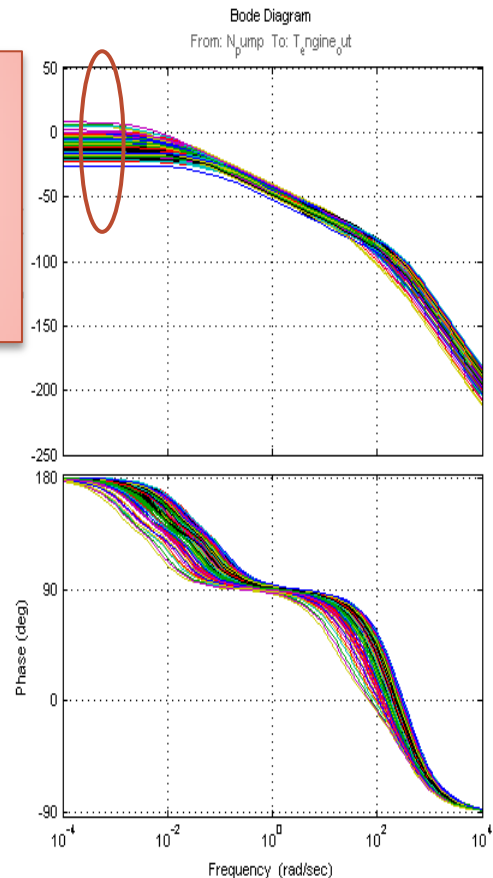
```
>> doe_setup = FMUDoESetup('CoolingLoop.fmu',  
'DesignParameters.xlsx', 'Dynamics');  
>> nbr_of_experiments = 100;  
>> result = doe_setup.qmc(nbr_of_experiments);  
>> result.bode(1, 1)
```



DYNAMICS: WHERE IS THE NONLINEARITY?

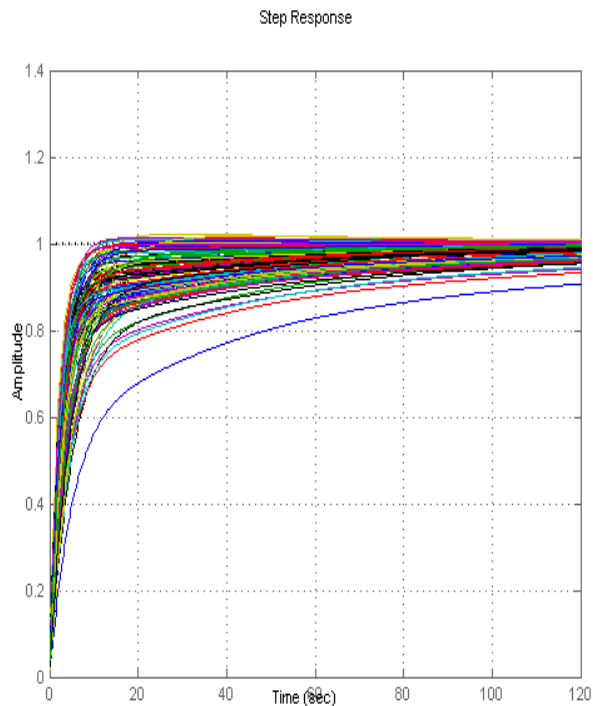
- Correlate feature of Bode plot with DoE factors

```
>> N = result.doe.nbr_of_experiments;  
>> ss_gain = zeros(N,1);  
>> for k = 1:1:N  
>>     ss_gain(k) =  
dcgain(result.linsys.sys{k} (1, 1));  
>> end  
>> result.main_effects(ss_gain, 'gain');
```



CONTROLLER EVALUATION

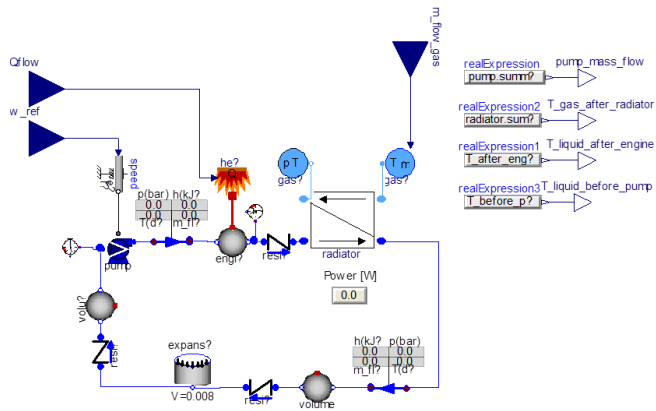
- Loop-shaping: PI-controller with $K = -50$, $T_i = 100$
- Closed-loop step response



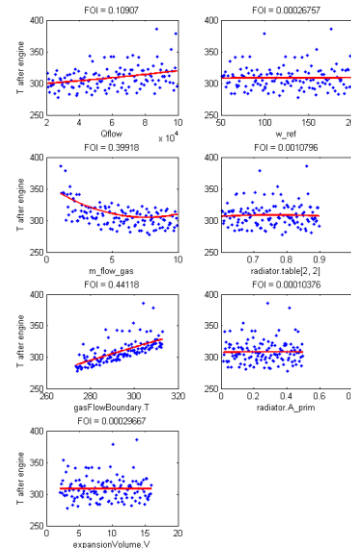
```
>> s = tf('s');  
>> Gc = -50*(1+1/(100*s));  
>> cl_sys = cell(N,1);  
>> for k = 1:1:N  
>>     Gp = result.linsys.sys{k}(1, 1);  
>>     cl_sys{k} = minreal(Gp*Gc/(1+Gp*Gc));  
>> end  
>> batch_step(cl_sys);
```


IN SHORT...

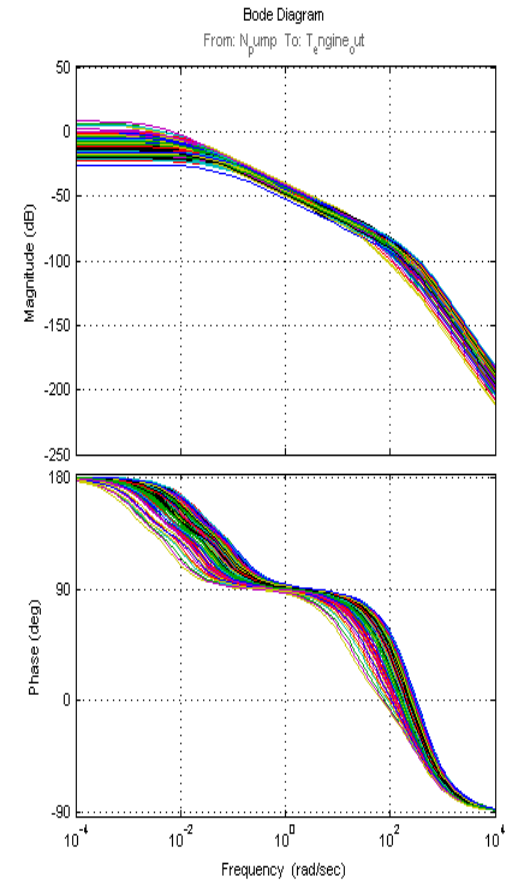
Get from here...



... to here...

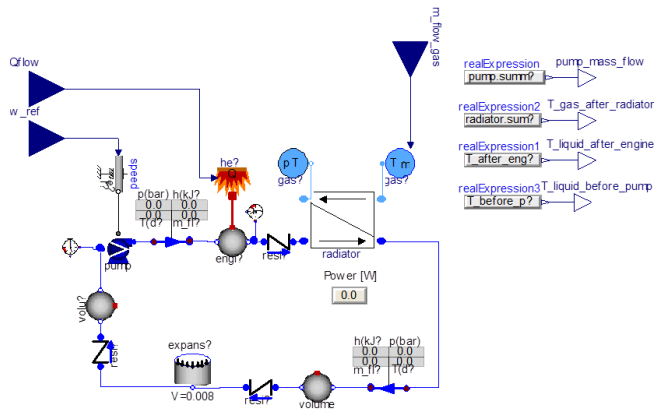


... and here...

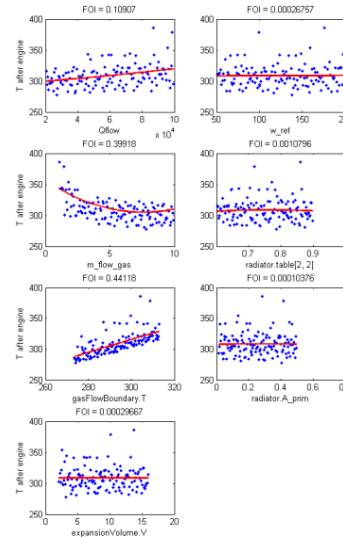


IN SHORT...

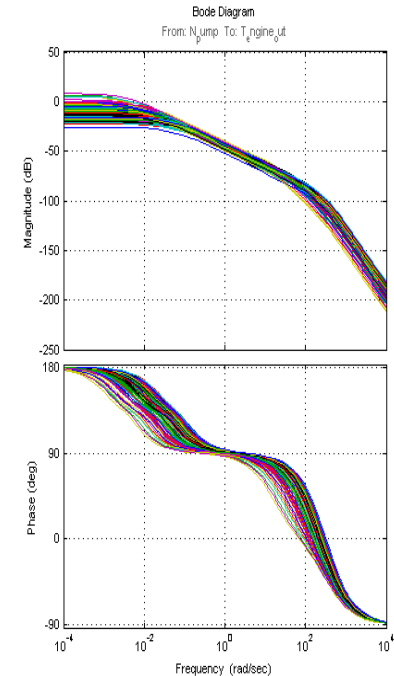
Get from here...



... to here...



... and here...



...in 4 Matlab commands!

```
>> doe_setup = FMUDoESetup('CoolingLoop.fmu', 'parameters.xlsx');
>> result = doe_setup.qmc(100);
>> result.main_effects('T_liquid_ae');
>> result.bode(1, 3, 100);
```

with plenty of options to customize the analysis

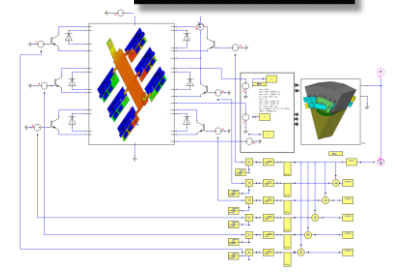
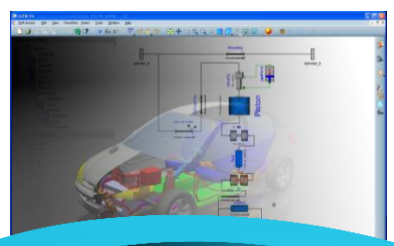
Conclusions

BENEFITS FROM LOOKING INTO SIMULATION MODELS

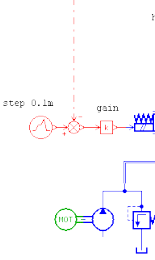
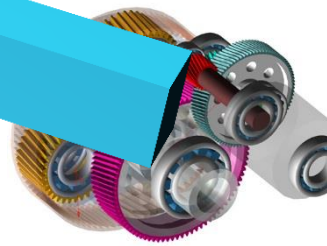
- Make your research process more robust and efficient
- Learn new skills for your future career
- Neglected research field: control design in the development process (workflows, methodologies, tools). Get there first!
- Plenty of relevant technical challenges/research topics: e.g. co-simulation and optimization of several FMUs

NEED HELP?

Let's talk!



```
#include <stdio.h>
int main()
{
    printf("Hello World!\n");
    return 0;
}
```



ALL CONNECTED!

