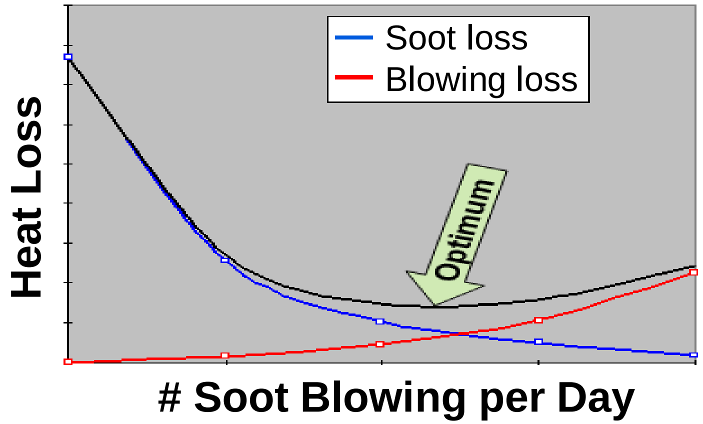
R. Franke, PSP-A24, 02.02.2015

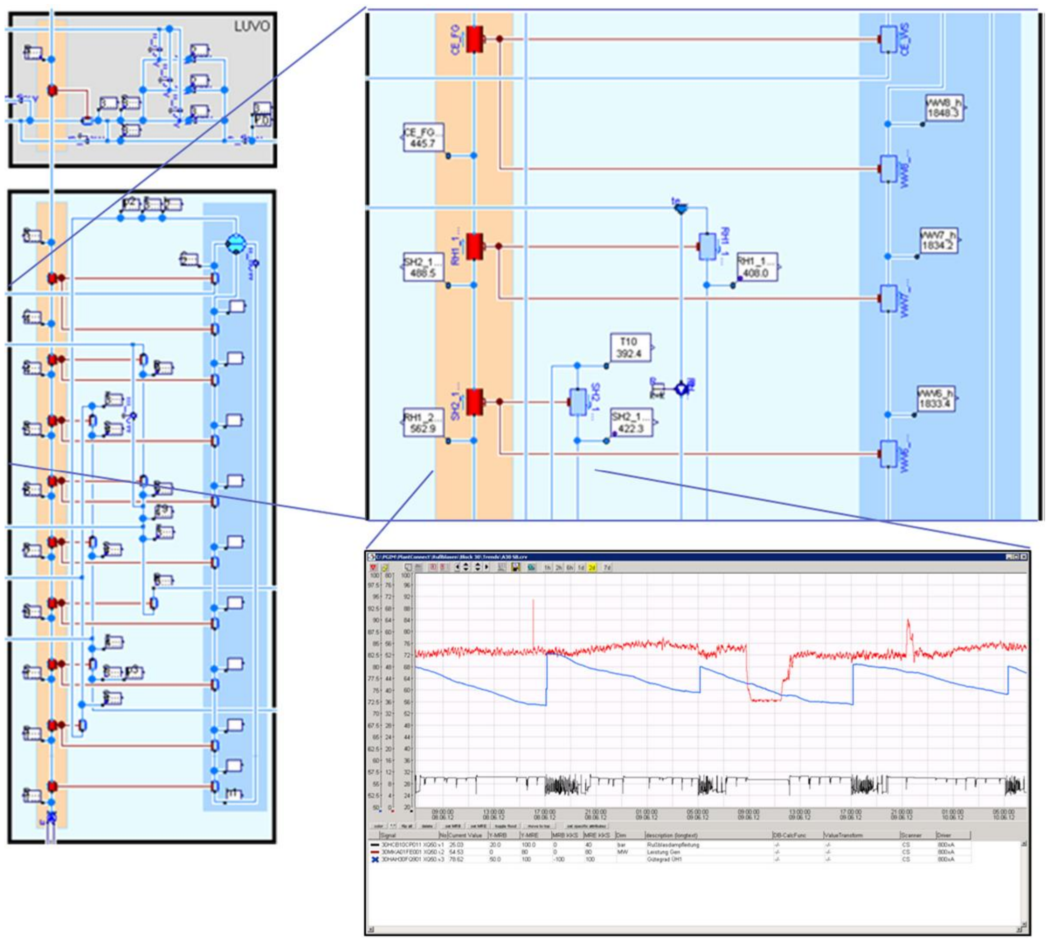
OpenModelica Annual Workshop 2015
Mathematical Optimization of Dynamic Systems with OpenModelica

Overview

- Introduction to ABB OPTIMAX
- DrumBoiler example
  1. Setpoint optimization
  2. Startup optimization
- Formulation of optimization programs in Modelica
- FMI
- Industrial applications
  - Industrial power plant exploiting FMI 2.0
  - Large-scale virtual power plant exploiting OpenModelica with FMI 2.0

OPTIMAX® SootBlowMax
Sootblowing optimization boiler





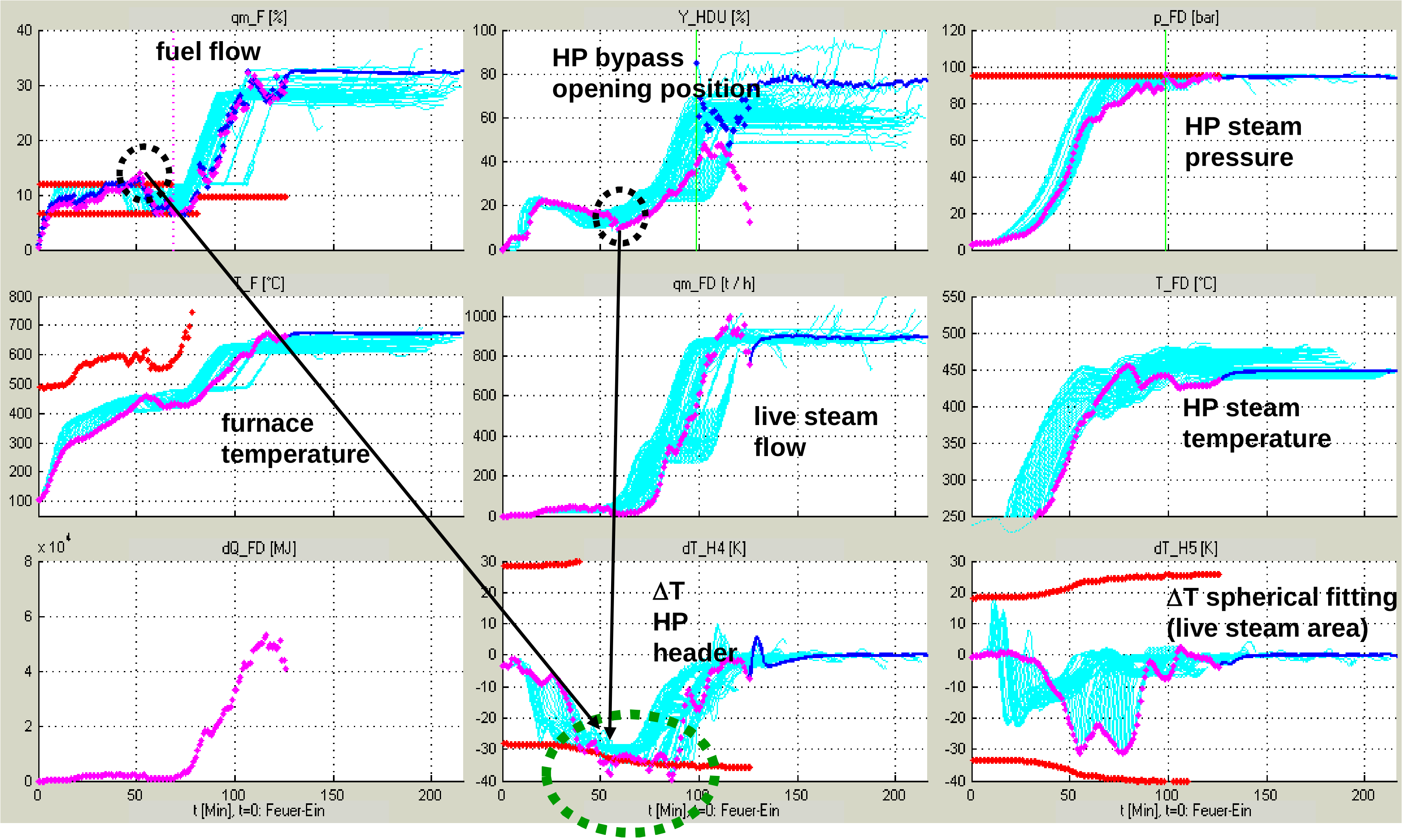
Task

- Decide on optimal sootblowing sequences maximizing efficiency

Solution

- Model of complete boiler (basing on Modelica.Fluid)
- Exploit overall mass and energy balance to replace missing measurements on flue gas side
- Online calculation of heat transfer coefficients between flue gas side and water/steam side

OPTIMAX® BoilerMax
Startup optimization boiler



Task

- Minimize fuel costs and startup time subject to thermal stress constraints

Solution

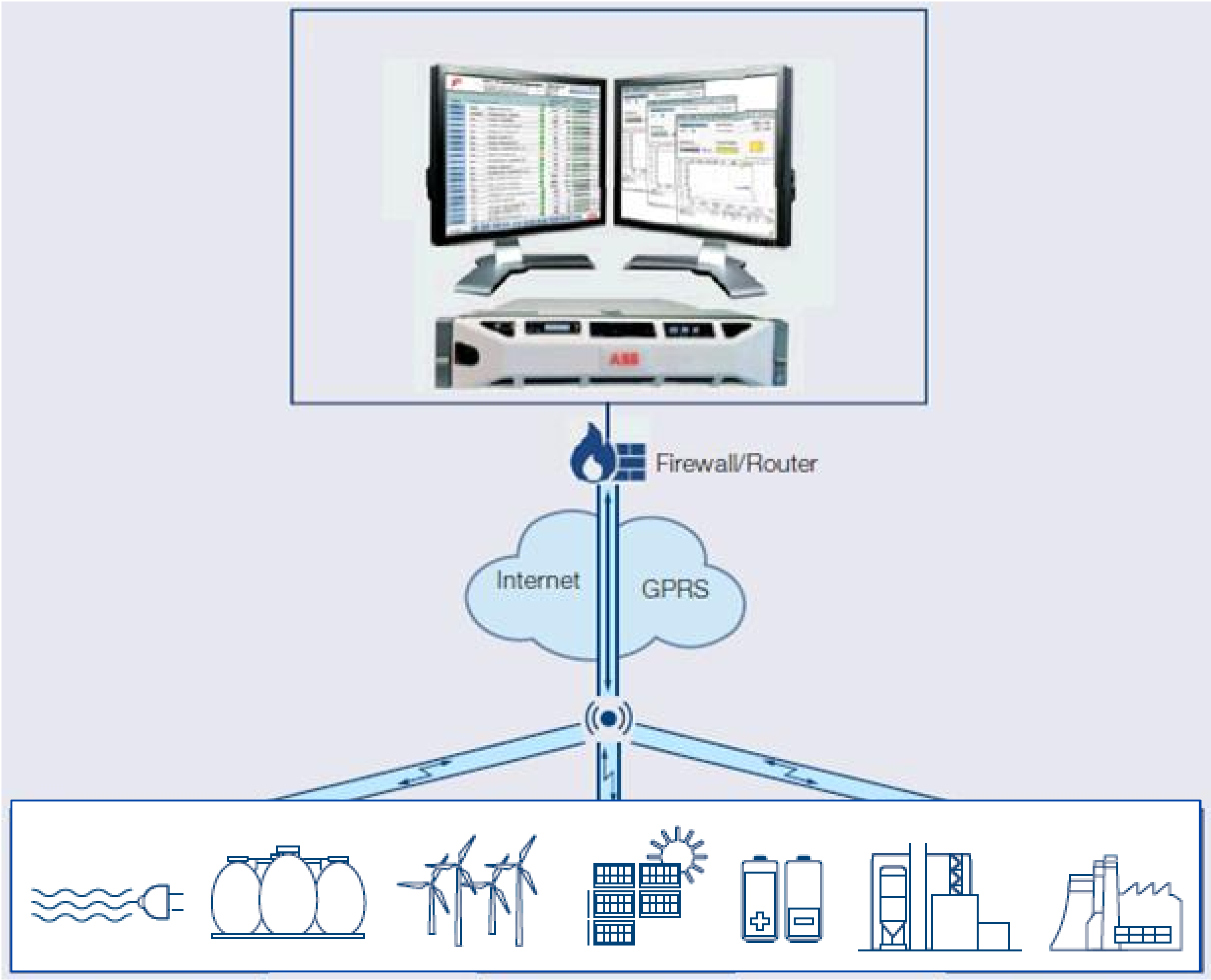
- NMPC for boiler startup

Graphics:

- Blue lines: optimized predictions, updated every minute
- Magenta points: actual operating points

See: https://www.modelica.org/events/modelica2006/Proceedings/sessions/Session3c2.pdf

OPTIMAX® PowerFit
Optimizing control of Virtual Power Plants and MicroGrids



Task

- Aggregate many small production units and treat them like one big power plant
- Exploit multiple forms of energy (e.g. el and heat) and storages

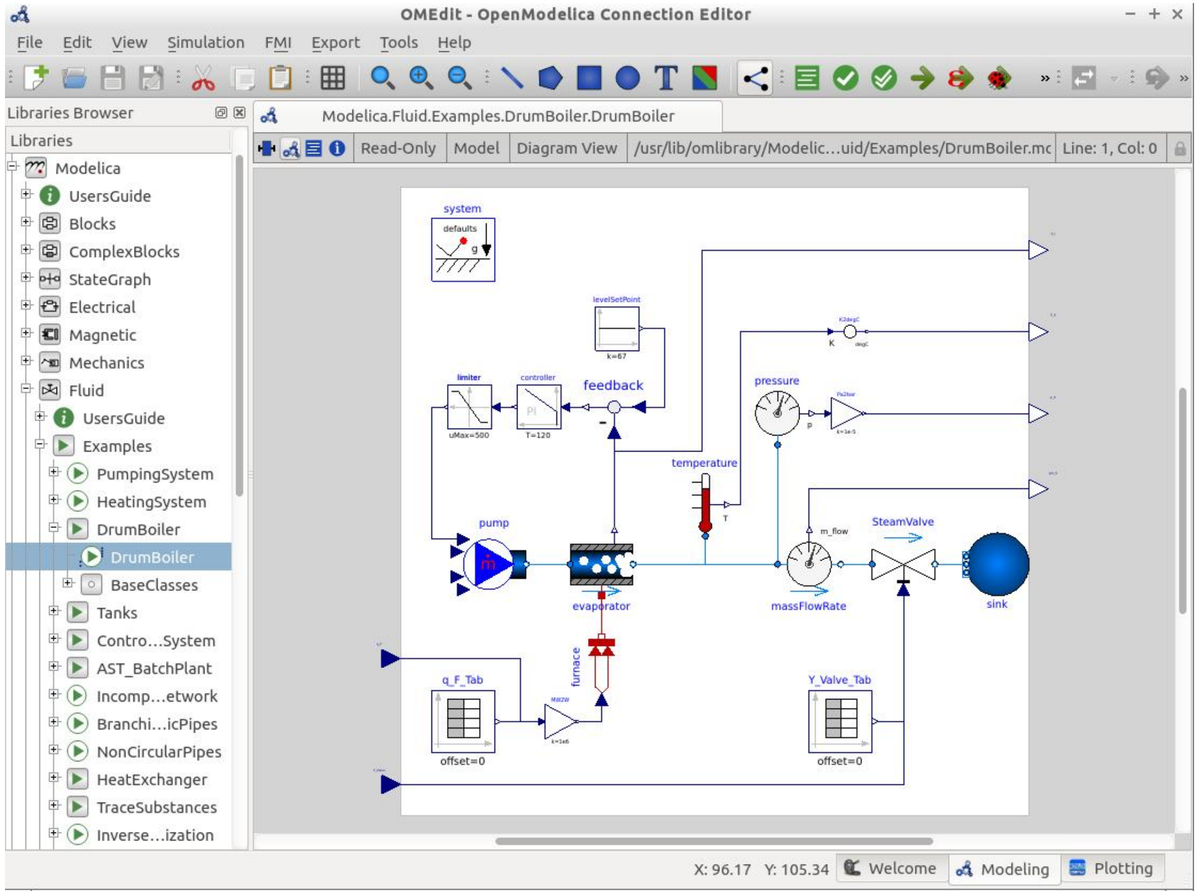
Solution

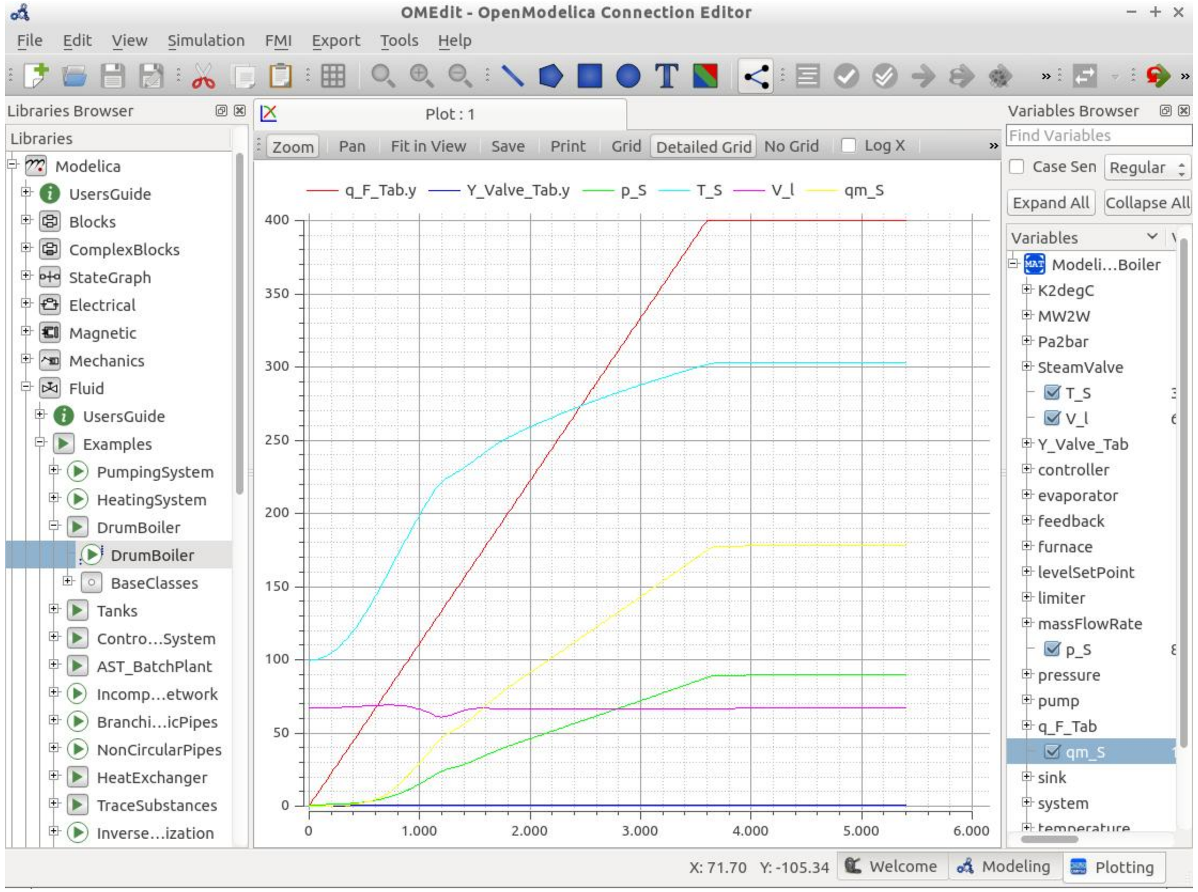
- Build overall plant model (exploiting Modelica multi-physics)
- Formulate optimizing control task as mathematical program
- Online optimization of set points and plant schedules

Overview

- Introduction to ABB OPTIMAX
- DrumBoiler example
  1. Setpoint optimization
  2. Startup optimization
- Formulation of optimization programs in Modelica
- FMI
- Industrial applications
  - Industrial power plant exploiting FMI 2.0
  - Large-scale virtual power plant exploiting OpenModelica with FMI 2.0

DrumBoiler Example
Open in OMEdit (left) and simulte until steady-state reached (right)





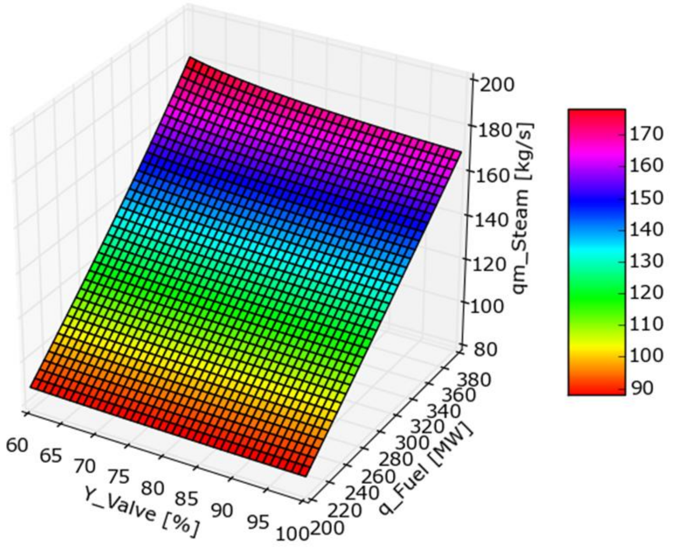
See: R. Franke, M. Rode, K. Krüger: On-line Optimization of Drum Boiler Startup, Modelica 2003, https://www.modelica.org/events/Conference2003/papers/h29_Franke.pdf

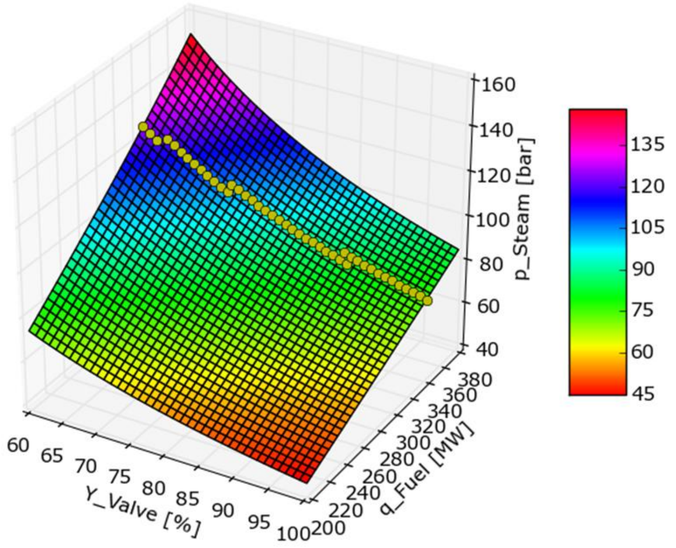
DrumBoiler setpoint optimization example
Simulation studies

Task: find steady-state operating point minimizing fuel q_F such that:
$qm_{Steam} = 150$ *kg/s, 100 bar* $\le p_{Steam} \le$ *120 bar*

Simulation aproach:

- obtain steady-state for inputs: $60 \le Y_{Valve} \le 100$, $200 \le q_F \le 380$
- plots show of 41 x 37 = 1517 steady-state values
- select best point





Nonlinear programming (NLP)

Nonlinear Programming (NLP)

Minimize optimization objective *J* subject to equality constraints *h* and inequality constraints *g*:

$$J(v) \underset{v}{\rightarrow} min \qquad J: R^n \rightarrow R^1$$

$$h(v) = 0 \qquad h: R^n \rightarrow R^{m_e}$$
$$g(v) \geq 0 \qquad g: R^n \rightarrow R^m$$

Setpoint optimization example

$$v = \begin{pmatrix} x \\ u \end{pmatrix}$$
$$x = (x_{PI}, p_{Steam}, V_{Liquid})$$
$$u = (q_F, Y_{Valve})$$

$$J(v) = q_F \underset{x,u}{\rightarrow} min \qquad J: R^5 \rightarrow R^1$$

$$h(v) = \begin{pmatrix} f(x,u) \\ qm_S - 150 \end{pmatrix} = 0 \qquad h: R^5 \rightarrow R^4$$

$$g(v) = \begin{pmatrix} 120 - p_S \\ p_S - 100 \\ 500 - q_F \\ q_F \\ 1 - Y_{Valve} \\ Y_{Valve} \end{pmatrix} \geq 0 \qquad g: R^5 \rightarrow R^6$$

Treatment of nonlinear programs (NLP)
Lagrangian, KKT conditions, Newton method

Lagrange function

$$L(v,\lambda,\mu) = J(v) - \lambda^T h(v) - \mu^T g(v)$$
$$L: R^n \times R^{m_e} \times R^m \rightarrow R^1$$

Karush Kuhn Tucker conditions

Stationarity

$$\nabla_v L(v,\lambda,\mu) = \nabla J(v) - \nabla h(v)^T \lambda - \nabla g(v)^T \mu = 0$$

Feasibility

$$\nabla_\lambda L(v,\lambda,\mu) = -h(v) = 0$$
$$g(v) \geq 0$$
$$\mu \geq 0$$

Complementarity

$$g(v)^T \mu = 0$$

Lagrange Newton iteration (*m=0*)

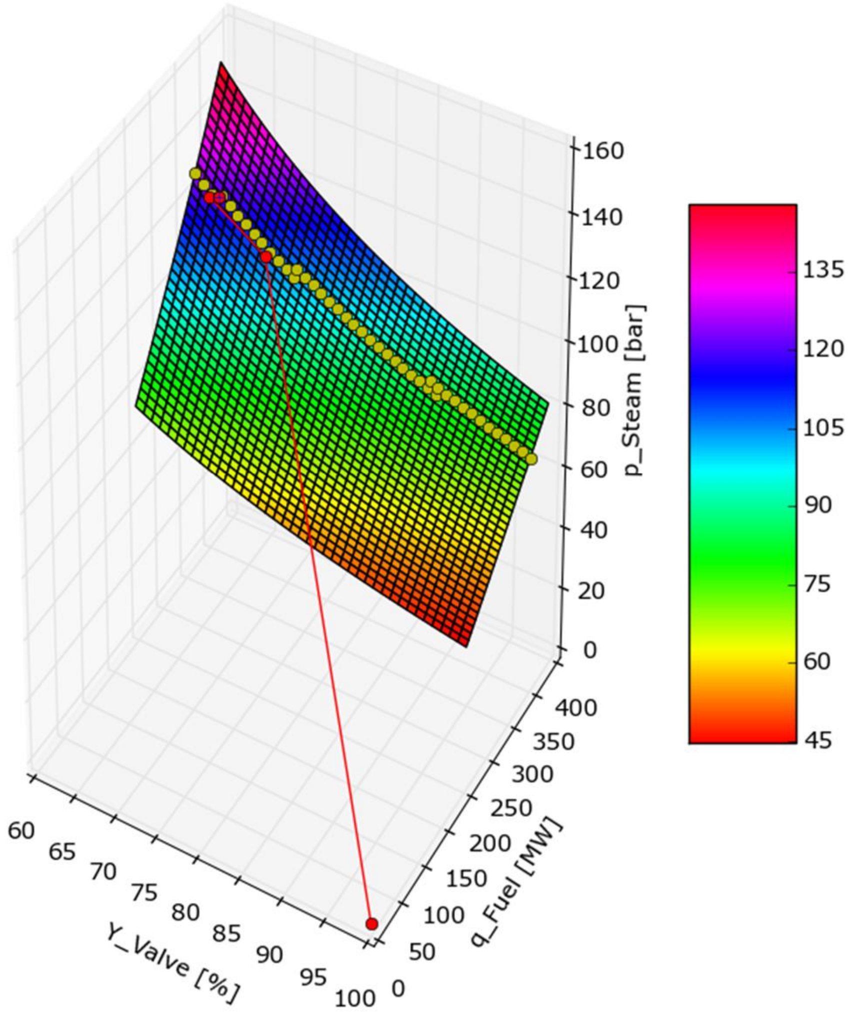
$$\nabla^2 L(v,\lambda) \begin{pmatrix} \Delta v \\ \Delta\lambda \end{pmatrix} = -\nabla L(v,\lambda)$$

$$\begin{pmatrix} v^+ \\ \lambda^+ \end{pmatrix} := \begin{pmatrix} v + \Delta v \\ \lambda + \Delta\lambda \end{pmatrix}$$

Specific Lagrange Newton methods

- Active Set Method:
  convert to equality constrained problems
- Interior Point Method:
  augment Lagrange function for inequalities
- Sequential Quadratic Programming (SQP)
  fully solve constraints in each iteration
- Quasi Newton Method:
  numerical update/approximation of Hessian

DrumBoiler setpoint optimization
Application of mathematical programming



Starting point:

- $q_F = 0$ *MW*
- $Y_{Valve} = 100\%$
- *x = (0, 1 bar, 65 m3)*

Solution after 5 iterations:

- $q_F = 328$ *MW*
- $Y_{Valve} = 62.8\ \%$
- *x = (15, 120 bar, 67 m3)*

See red solution path in figure left

Dynamic Optimization
Numerical treatment

$$J = f_0(x^K) + \sum_{k=0}^{K-1} f_0(x^k, u^k) \longrightarrow \min_{x^0,\, u^k}$$

$$s.t.$$

$$c^k(x^k, u^k) \geq 0, \quad k = 0, \dots, K-1$$

$$c^K(x^K) \geq 0$$

$$0 = \begin{cases} \text{Collocation:} \\ f^k(x^k, u^k, x^{k+1}) \\ \text{Multistage/shooting:} \\ f^k(x^k, u^k) - x^{k+1} \end{cases}, \quad k = 0, \dots, K-1$$

- Optimize over time horizon $t \in [t_0, t_f]$
- Degrees of freedom: optimal control *u(t)* and possibly initial states x(t_0)
- Numerical treatment:
  - parameterize control trajectories: $u(t) = u(u^k)$, $k=0,1,\dots,K-1$
  - Treat constraints and state equations in discrete time
    - Collocation: implicit state equations
    - Multistage: explicit state equations
  - Collect all discrete-time control and state variables in one large vector of optimization variables
    $$v = (x^0, u^0, x^1, u^1, \dots, x^{K-1}, u^{K-1}, x^K)$$
- Alt. simple treatment: $\breve{v} = (u^0, u^1, \dots, u^{K-1})$

DrumBoiler startup optimization

Minimize

- quadratic deviation from set point

$$J = \int_{t=t_0}^{t_f} w^T \begin{Bmatrix} \left[p_S(t) - p_{ref}\right]^2 \\ \left[q_{m,S}(t) - q_{m,ref}\right]^2 \\ \left[\frac{dq_F(t)}{dt}\right]^2 \end{Bmatrix} dt \xrightarrow[u(t)]{} min$$

subject to

- control bounds

$$0 \le \quad q_F \quad \le 500\ MW$$
$$0 \le Y_{Valve} \le 1$$

- rate of change bounds

$$-24\ MW/min \le \quad \frac{dq_F}{dt} \quad \le 24\ MW/min$$

- state/output constraints
(thermal and membrane stress)

$$-150 \frac{N}{mm^2} \le \ \sigma_{Drum} \ \le 150 \frac{N}{mm^2}$$
$$\sigma_{Drum} \quad = 10^{-3} \frac{dT_{Drum}}{dt} + 10^{-5} p_{Drum}$$

Overview

- Introduction to ABB OPTIMAX
- DrumBoiler example
  1. Setpoint optimization
  2. Startup optimization
- Formulation of optimization programs in Modelica
- FMI
- Industrial applications
  - Industrial power plant exploiting FMI 2.0
  - Large-scale virtual power plant exploiting OpenModelica with FMI 2.0

Formulation in Optimica – see JModelica.org
Extending Modelica syntax (see red text)

```
optimization DrumBoilerStartup(
  objective = 1e-7 * (p_S - 110)^2 + 1e-8 * (qm_S - 180)^2 + 1e-4 * dq_F^2,
  startTime = 0,
  finalTime = 3600)
  extends DrumBoiler(
                     q_F = integrated_q_F,
                     Y_Valve(free = true, initialGuess = 1),
                     use_inputs = true
                     );
  Real sigma_D = (-1.0e3 * der(evaporator.T_D)) + 1.0e-05 * evaporator.p;
  input Real dq_F(free = true, initialGuess = 400/3600);
  Real integrated_q_F;
equation
  der(integrated_q_F) = dq_F;
constraint
  0 <= q_F; q_F <= 500;
  -24/60 <= dq_F; dq_F <= 24/60;
  0 <= Y_Valve; Y_Valve <= 1;
  -150 <= sigma_D; sigma_D <= 150;
end DrumBoilerStartup;
```

Formulation in OpenModelica (provided by Vitalij Ruge)
Inside Modelica model using standard attributes and new annotations

```
model optDrumBoiler
  extends DrumBoiler(use_inputs = true,
                     Y_Valve(min = 0, max = 1, nominal = 1, start = 0.5),
                     q_F(min = 0, max = 500, start = 0, fixed = true, nominal = 400),
                     controller.x(nominal = 10));
  Real cost_p_S(nominal = 1e7) annotation(isLagrange = true);
  Real cost_qm_S(nominal = 1e8) annotation(isLagrange = true);
  Real cost_dq_F(nominal = 1e4) annotation(isLagrange = true);
  Real sigma_D(min = -150, max = 300) = (-1.0e3 * der(evaporator.T_D)) + 1.0e-05 * evaporator.p;
  Real conSigma(min = -150, max = 150) annotation(isConstraint = true);
  input Real dq_F(min = -24/60, max = 24/60, start = 400/3600);
equation
  cost_p_S = (p_S - 110)^2;
  cost_qm_S = (qm_S - 180)^2;
  cost_dq_F = dq_F^2;
  conSigma = sigma_D;
  der(q_F) = dq_F;
end optDrumBoiler;
```

Formulation with parameter records and custom annotations
Generic approach that nicely applies to optimization

```
import DO = DynamicOptimization.Types;  // parameter records for custom annotations

model DrumBoiler_StartupOptimization
  extends DrumBoiler(
    use_inputs = true,
    q_F
      annotation(DO.Input(u_active = true, u_start = 0, der_u_start = 400/3600,
        u0_max = 0, u_min = 0, u_max = 500,
        der_u_min = -24/60, der_u_max = 24/60, der_u_weight2 = 1e-4)),
    Y_Valve
      annotation(DO.Input(u_active = true, u_start = 1,
        u0_max = 0, u_min = 0, u_max = 1)),
    p_S
      annotation(DO.Output(y_ref = 110, y_nominal = 100, y_weight2 = 1e-3)),
    qm_S
      annotation(DO.Output(y_ref = 180, y_nominal = 100, y_weight2 = 1e-4)));

  output Real sigma_D = -1e3 * der(evaporator.T_D) + 1e-5 * evaporator.p
    annotation(DO.Output(y_min = -150, y_max = 150));
end DrumBoiler_StartupOptimization;
```

ABB Dynamic Optimization

Use Excel GUI automating import of model and upload to runtime

Microsoft Excel - Drum Boiler_Startup Optimization.xls

File Edit View Insert Format Tools Data Window Help Dynamic Optimization

C7 = DrumBoiler

	A	B	C
1			
2	Object	String	Drum Boiler
3	Aspect	String	Startup Optimization
4			
5			
6	SOH	Type	Value
7	ModelName	String	DrumBoiler
8	SolverName	String	HQP
9	SolverVersion	String	1.9.8
10	SolverState	String	Ready
11	SolverIteration	Integer	0
12	SolverMessage	String	.
13	Service.Enable	Boolean	FALSE
14	Service.Trigger	Integer	0
15	Service.Quit	Integer	0
16	Data.LogName	String	
17	Data.StartTime	DateTime	+00:00:00
18	Data.EndTime	DateTime	+01:00:00
19	Data.SampleTime	TimeSpan	0:01:00
20	Results.LogName	String	ResultsLog

Microsoft Excel - Drum Boiler_Startup Optimization.xls

File Edit View Insert Format Tools Data Window Help Dynamic Optimization

A1 = SOH

	A	B	C	D	E	F	G	H	I
1	SOH	varname	y	y_nominal	y_ref	y_min	y_max	y_weight1	y_weight2
2	Type	String	Real	Real	Real	Real	Real	Real	Real
3		T_S	0	100					
4		p_S	0	100	110				1.00E-03
5		qm_S	0	100	180				1.00E-04
6		sigma_D	0	100		-150	150		
7		V_l	0	100					

meta / input / output / parameter / state / Solver / DataLogs

Ready

Microsoft Excel - Drum Boiler_Startup Optimization.xls

File Edit View Insert Format Tools Data Window Help Dynamic Optimization

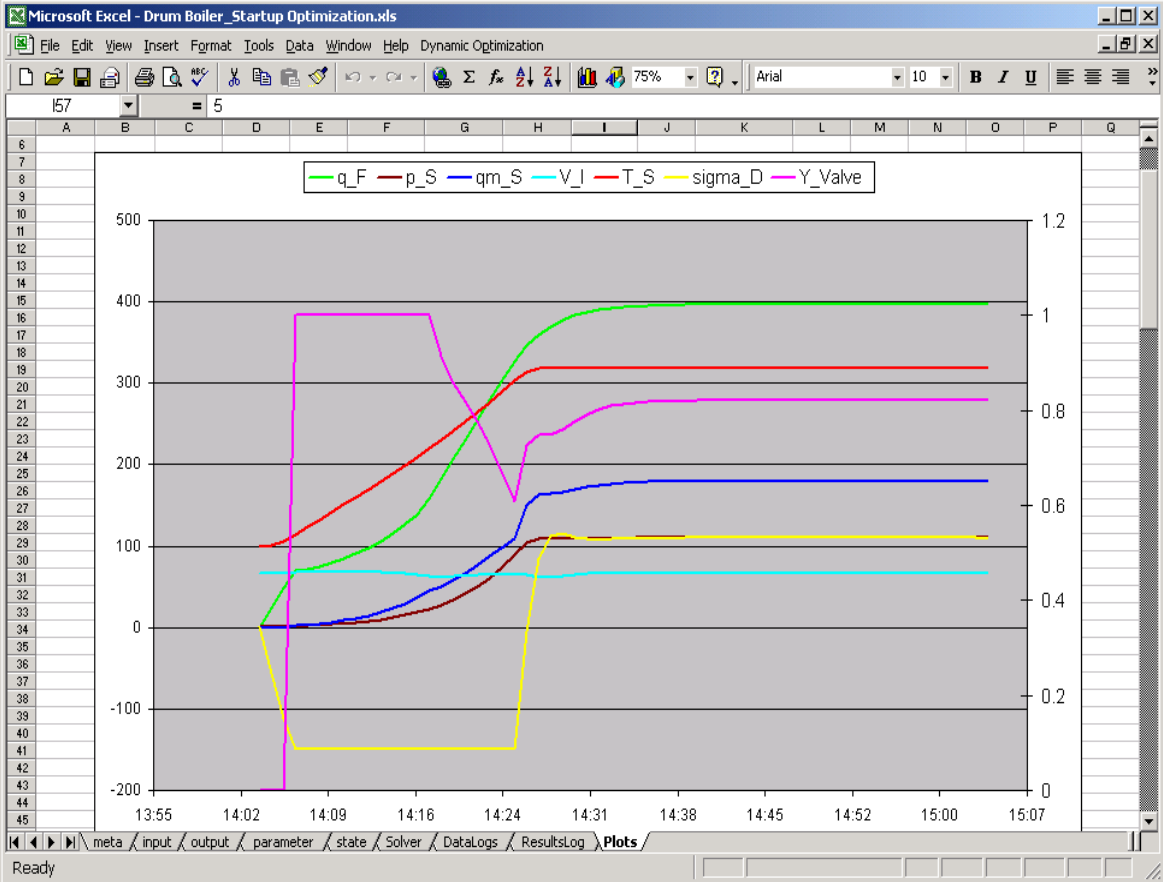
A1 = SOH

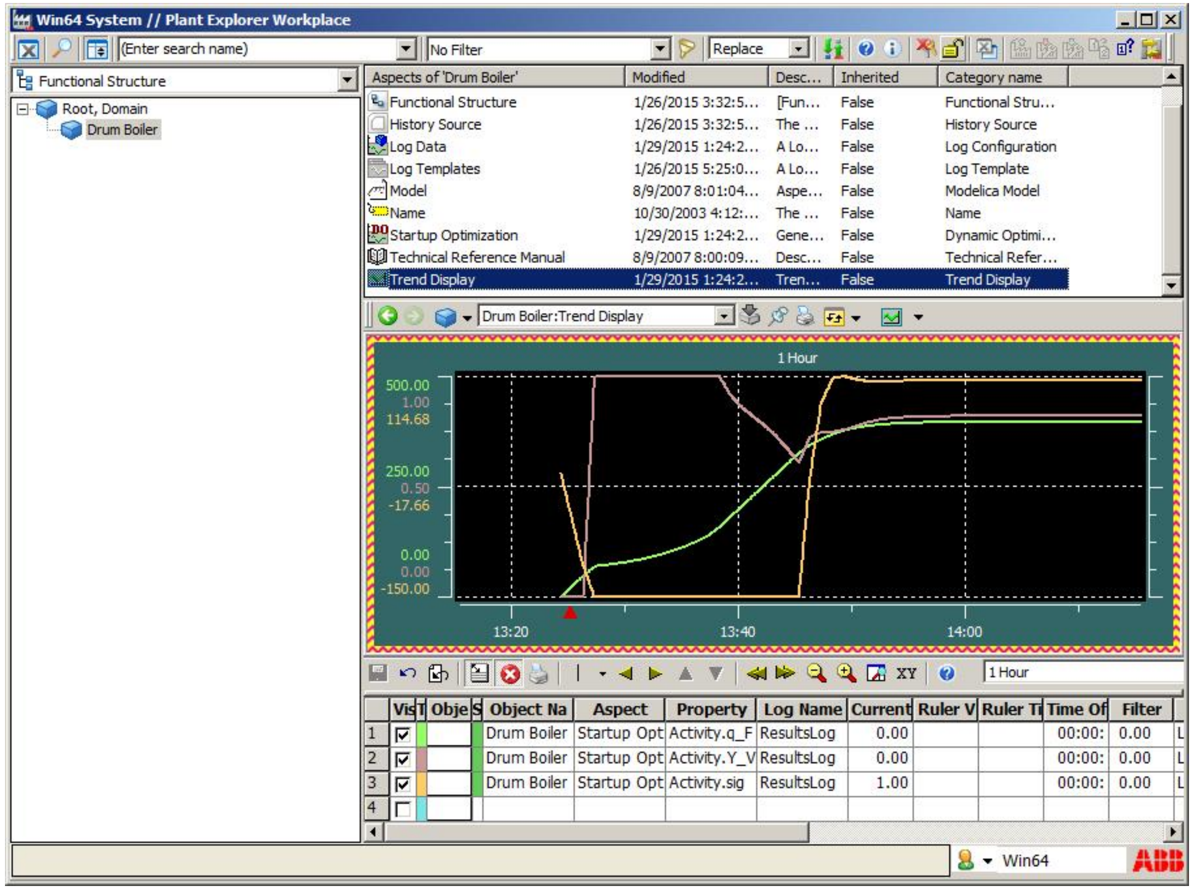
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	SOH	varname	u	u_start	u_order	u_active	u0_min	u0_max	u_min	u_max	der_u	der_u_start	der_u_min	der_u_max	der_u_weight2
2		String	Real	Real	Integer	Boolean	Real	Real	Real	Real	Real	Real	Real	Real	Real
3		q_F	0	0	1	TRUE	0	0	0	500	0	1.11E-01	-0.4	0.4	1.00E-04
4		Y_Valve	0	1	1	TRUE	0	1	0	1					

meta / input / output / parameter / state / Solver / DataLogs

Ready

ABB Dynamic Optimization
Offline engineering in Excel / Online optimization in control system



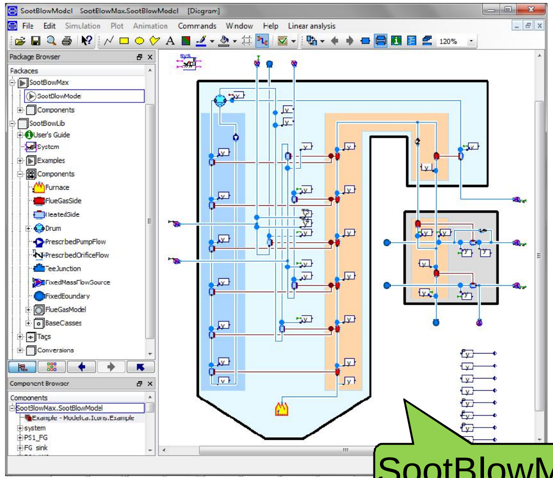


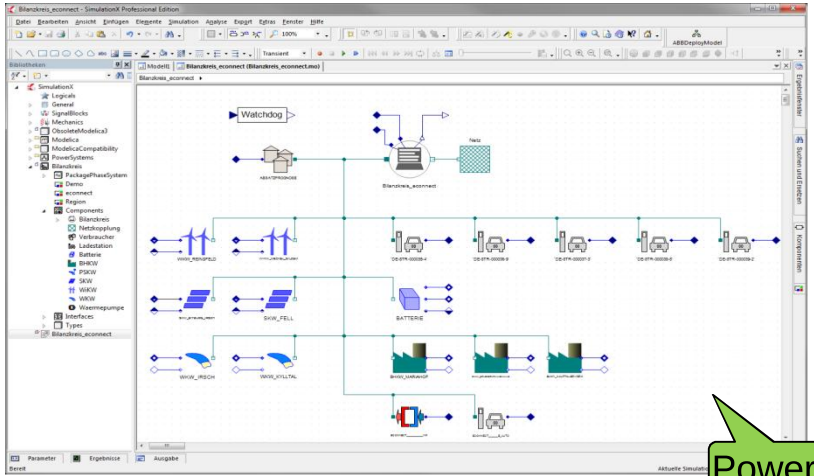
Overview

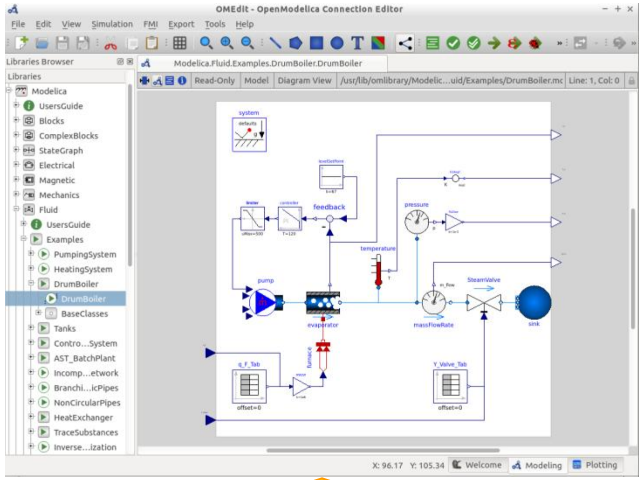
- Introduction to ABB OPTIMAX
- DrumBoiler example
  1. Setpoint optimization
  2. Startup optimization
- Formulation of optimization programs in Modelica
- FMI
- Industrial applications
  - Industrial power plant exploiting FMI 2.0
  - Large-scale virtual power plant exploiting OpenModelica with FMI 2.0

ABB Dynamic Optimization
Standardized engineering and deployment of control&optimization apps

Modelica for application engineering (www.modelica.org)



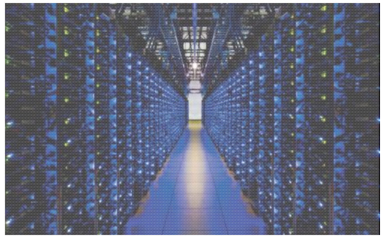




FMI (Functional Model Interface) for deployment (www.fmi-standard.org)

Embedded

Server



Cloud

Exploitation of sparse structures with FMI 2.0
Jacobian for constraints (state equations and general constraints)

$$\frac{\partial h(v)}{\partial v} = \begin{pmatrix} \frac{\partial f^0}{\partial x^0} & \frac{\partial f^0}{\partial u^0} & -I & & & & \\ & & \ddots & & & & \\ & & & \frac{\partial f^{K-1}}{\partial x^{K-1}} & \frac{\partial f^{K-1}}{\partial u^{K-1}} & -I \\ \frac{\partial \bar{c}^0}{\partial x^0} & \frac{\partial \bar{c}^0}{\partial u^0} & & & & \\ & & \ddots & & & \\ & & & \frac{\partial \bar{c}^{K-1}}{\partial x^{K-1}} & \frac{\partial \bar{c}^{K-1}}{\partial u^{K-1}} & \\ & & & & & \frac{\partial \bar{c}^K}{\partial x^K} \end{pmatrix}$$

Note:

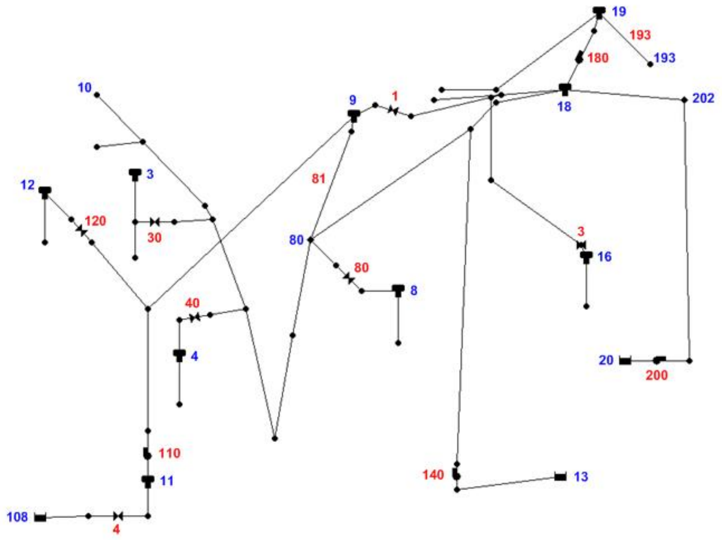
$$\left(\frac{\partial f^k}{\partial x^k} \; \frac{\partial f^k}{\partial u^k} \right), \quad k = 0, \ldots, K-1$$

are the sensitivity matrices obtained for each stage by integrating sensitivity equations together with state equations

Exploitation of sparse structures
Example: water distribution network

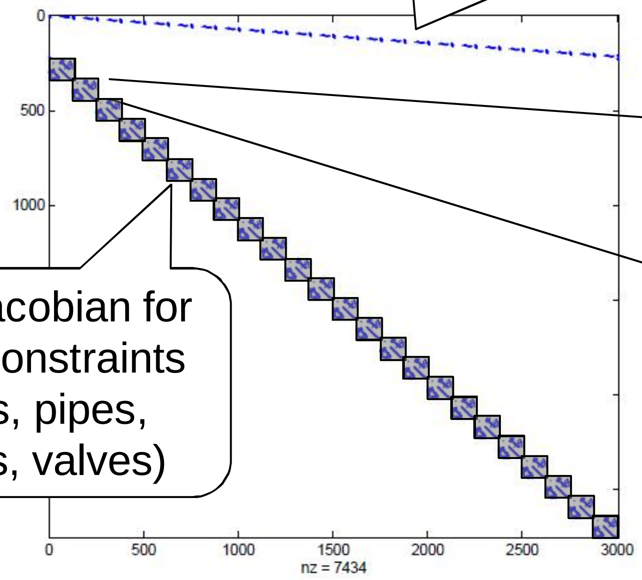
S-function, FMI 1.0: can exploit sparsity resulting from time staging (left spy plot)

FMI 2.0: additionally exploit sparse models (right spy plot)

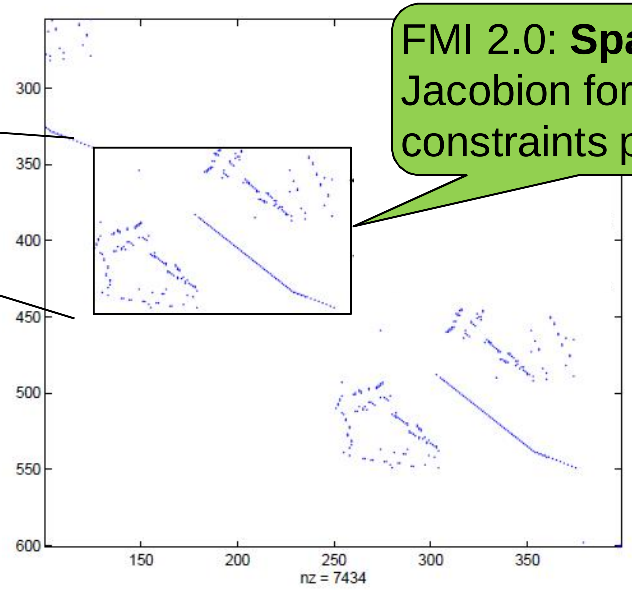


Jacobian for state equations (tanks) over 24 stages

24x Jacobian for DAE constraints (nodes, pipes, pumps, valves)



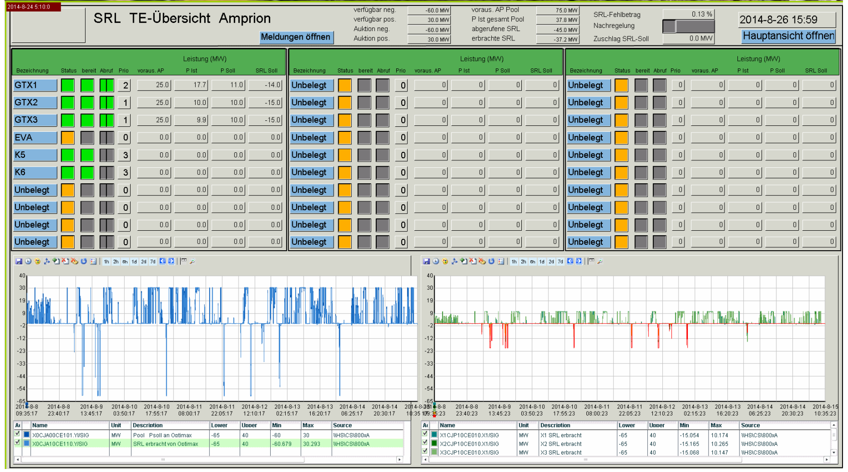
FMI 2.0: **Sparse** Jacobion for DAE constraints per stage



Overview

- Introduction to ABB OPTIMAX
- DrumBoiler example
  1. Setpoint optimization
  2. Startup optimization
- Formulation of optimization programs in Modelica
- FMI
- Industrial applications
  - Industrial power plant exploiting FMI 2.0
  - Large-scale virtual power plant exploiting OpenModelica with FMI 2.0

Reference OPTIMAX PowerFit
Industrial power plant / active site



Site

- 800 buildings, 120 production facilities
- 90 companies (chemical, pharmaceutical)
- 22.000 employees

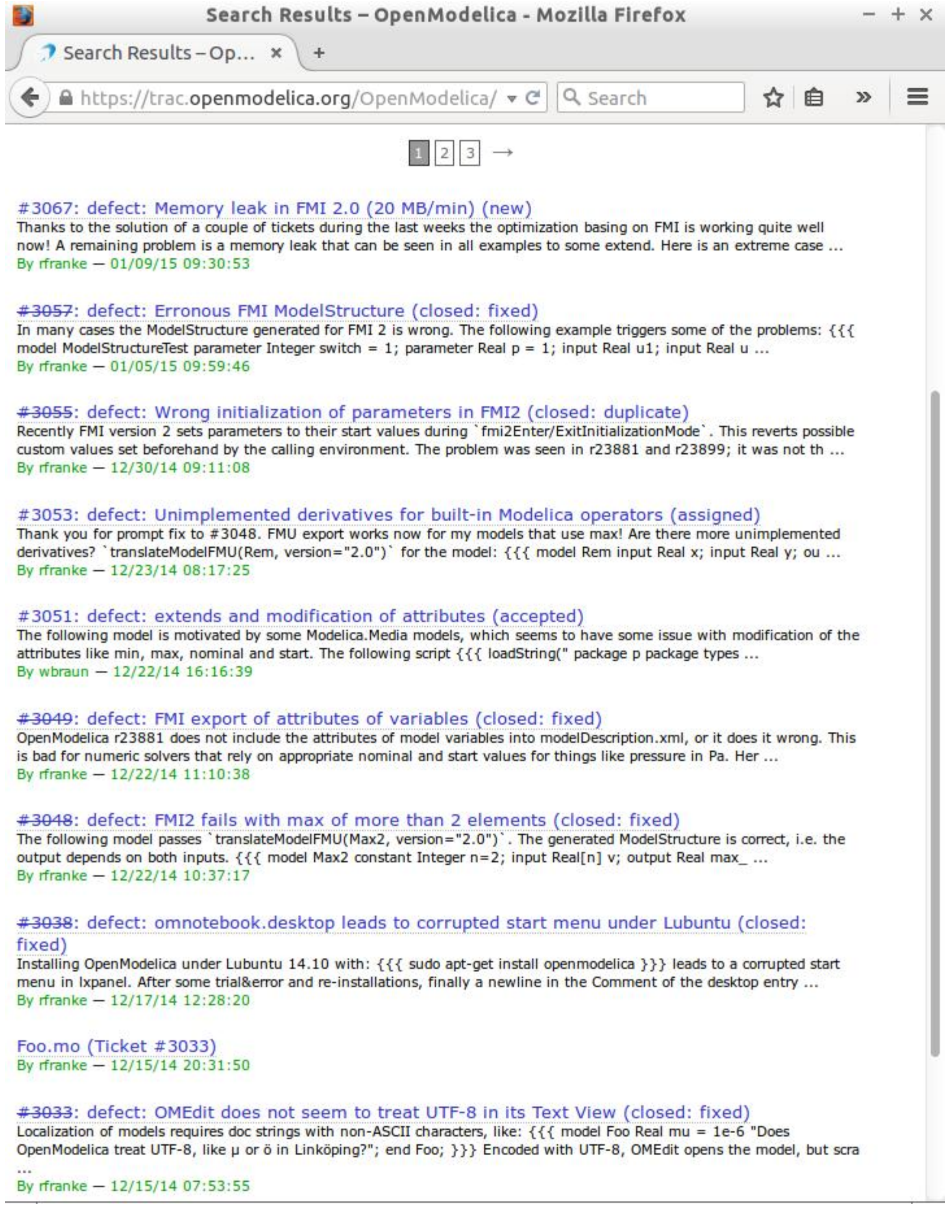
OPTIMAX PowerFit

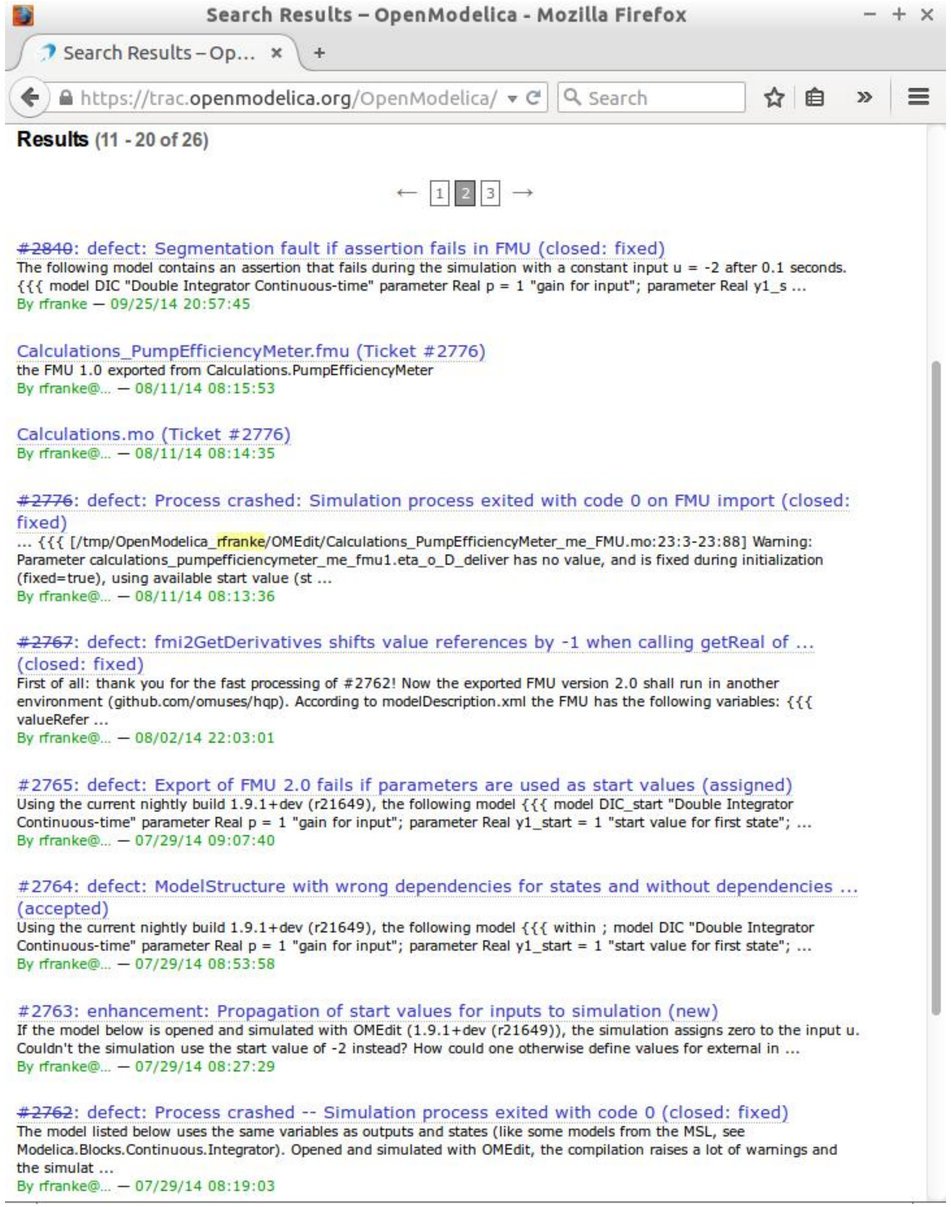
- Software extension of regular plant control system
- Providing secondary frequency control with 3 GTs and 2 E-Boilers
- To be extended
- Connect plant control to energy management

First industrial application of FMI 2.0

- Running 7/24 since June 9, 2014 00:00:00

Thank‘s to Your support … 26 tickets later …





Reference OPTIMAX PowerFit
Large-scale Virtual Power Plant

Customer

- Fast growing start-up company
- Pooling more than 1000 renewable plants in all 4 German grid areas

OPTIMAX PowerFit

- Stand-alone supervisory control system
- Geographically redundant with multiple high-performance virtual servers
- Providing secondary frequency control with large-scale Virtual Power Plant

Benefit of FMI 2.0

- Exploit sparse model structures to reduce required server capacity by 50%

Conclusions

Modelica

- unique technology for the implementation of technical process knowledge
- Modelica models provide constraints for optimization programs
- Additional specifications, like bounds and weights, can be treated with custom annotations – exploiting parameter records and Modelica parameter GUIs

FMI

- offers a standard interface for the deployment of executable models
- unique features: XML declaration file, sparse model structures, …
- missing in FMI 2.0: DAE constraints, clocked equations for optimization

OpenModelica

- unique open development process; great progress during the last years
- some pioneering features – e.g. MetaModelica for library upgrades
- missing for commercial applications: encryption of Modelica libraries

This work was supported in parts by BMBF within the ITEA2 project MODRIO (Model Driven Physical Systems Operation), Project Id: ITEA 2 – 11004.

Power and productivity
for a better world™

ABB