

Meta-Modeling With OptiY®

Winning Mathematical Surrogate Models from Measurement Data or Complex Finite Element Analysis

Dr. The-Quan Pham

OptiY e.K.

Aschaffenburg

Dr. Alfred Kamusella

Institute of Electromechanical and Electronic Design

Dresden University of Technology

ITI Symposium

26th-27th November 2009 in Dresden

Outline

1. Objective

2. Theoretical Basics

- Gaussian Process
- Visualization of the Adaptive Gaussian Process

3. Practical FEA Example

- Electromagnetic Actuator
- Characteristic Diagram $F=f(i,s)$ and $Psi=f(i,s)$
- System Simulation using Characteristic Diagrams

Objective

Simulation of a Real Product or Process

- Relationships or work principles are unknown
- Up to now only measurement data exist
- What should be modeled?

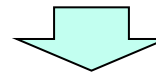
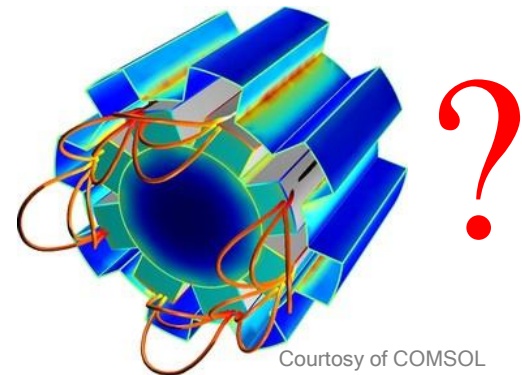
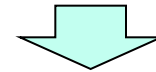
Complex Finite Element Model

- Long computing time: hours or days
- Control: loops of simulations
- Test scenarios: many simulations required
- Technical feasibility of system simulation?

Current Solutions

- Model reduction to network elements
- Mathematically describable relationships only
- Search for suitable model structures and parameters
- Parameter validation required
- Time-consuming and cost-intensive

Parameters

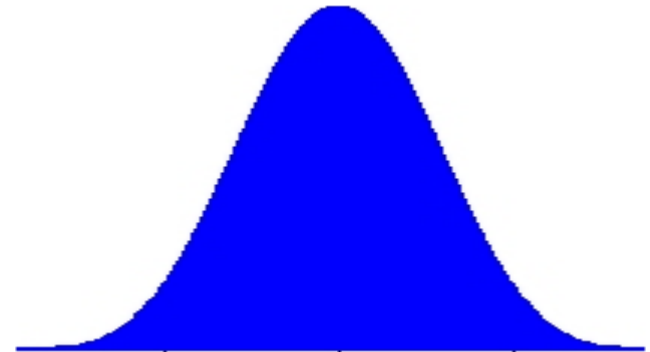


Measurement or
Simulation Data

Gaussian Process

- Polynomial $f(\mathbf{x})$ of p^{th} order for global adaptation
- Stochastic process $Z(\mathbf{x})$ for local adaptation

$$\begin{aligned}
 Y(\mathbf{x}) = & f_0 + b_{11}x_1 + b_{12}x_1^2 + \dots + b_{1p}x_1^p \\
 & + b_{21}x_2 + b_{22}x_2^2 + \dots + b_{2p}x_2^p \\
 & \dots \\
 & + b_{n1}x_n + b_{n2}x_n^2 + \dots + b_{np}x_n^p \\
 & + Z(\mathbf{x})
 \end{aligned}$$



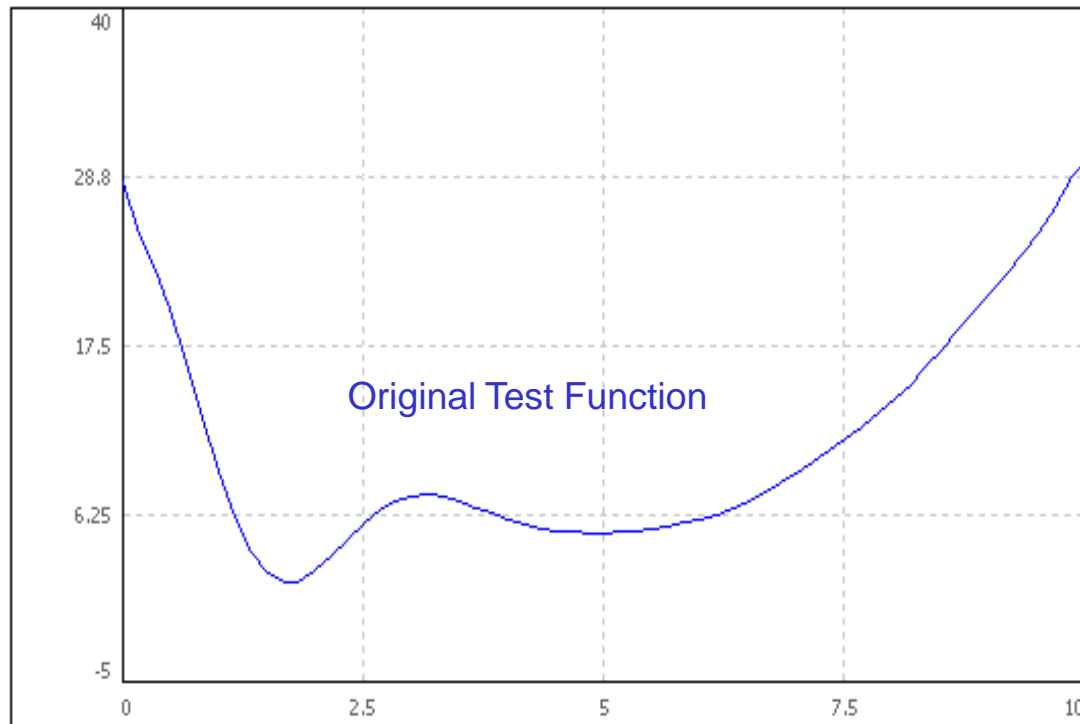
Correlation Function $R(\mathbf{x})$

- Multivariate Gaussian distribution (normal distribution)
- Interpolation between calculated points
- Interactions between individual parameters

$$R(x_i, x_j) = \sum_{k=1}^n w_k^2 (x_i - x_j)^2$$

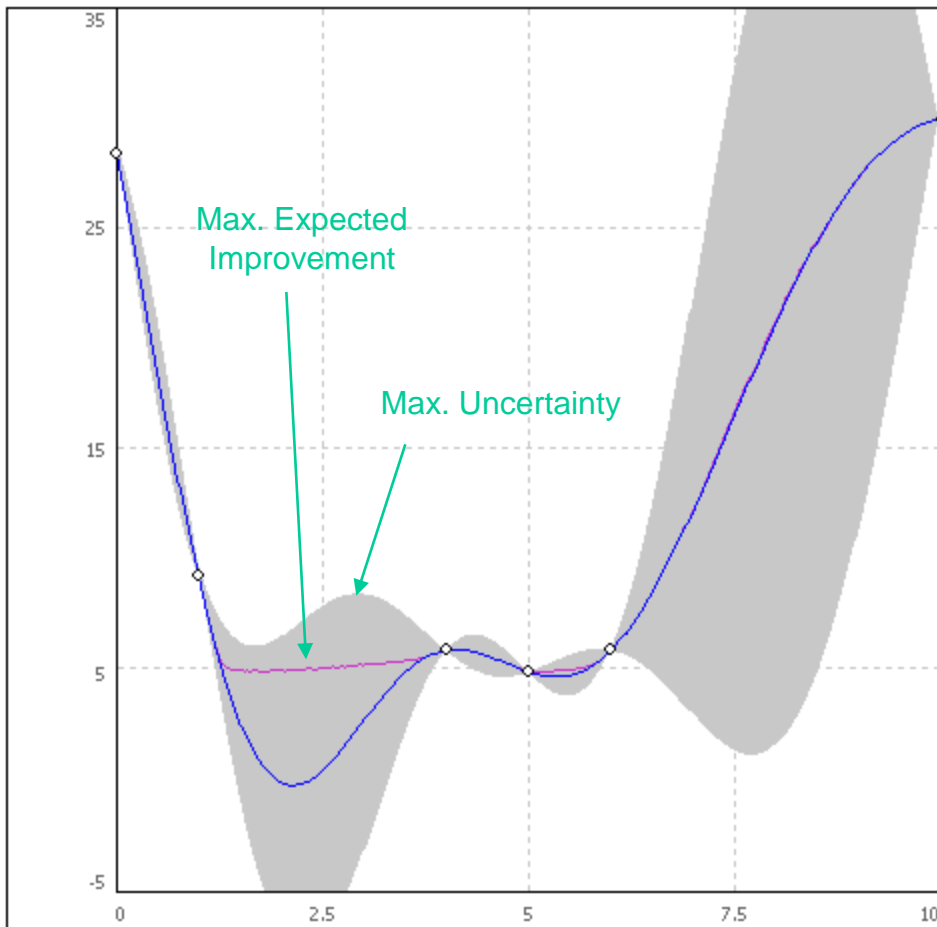
Visualization of the Adaptive Gaussian Process

$$Y = (X - 5)^2 - 15 \cdot e^{-(X-1.5)^2} + 5$$

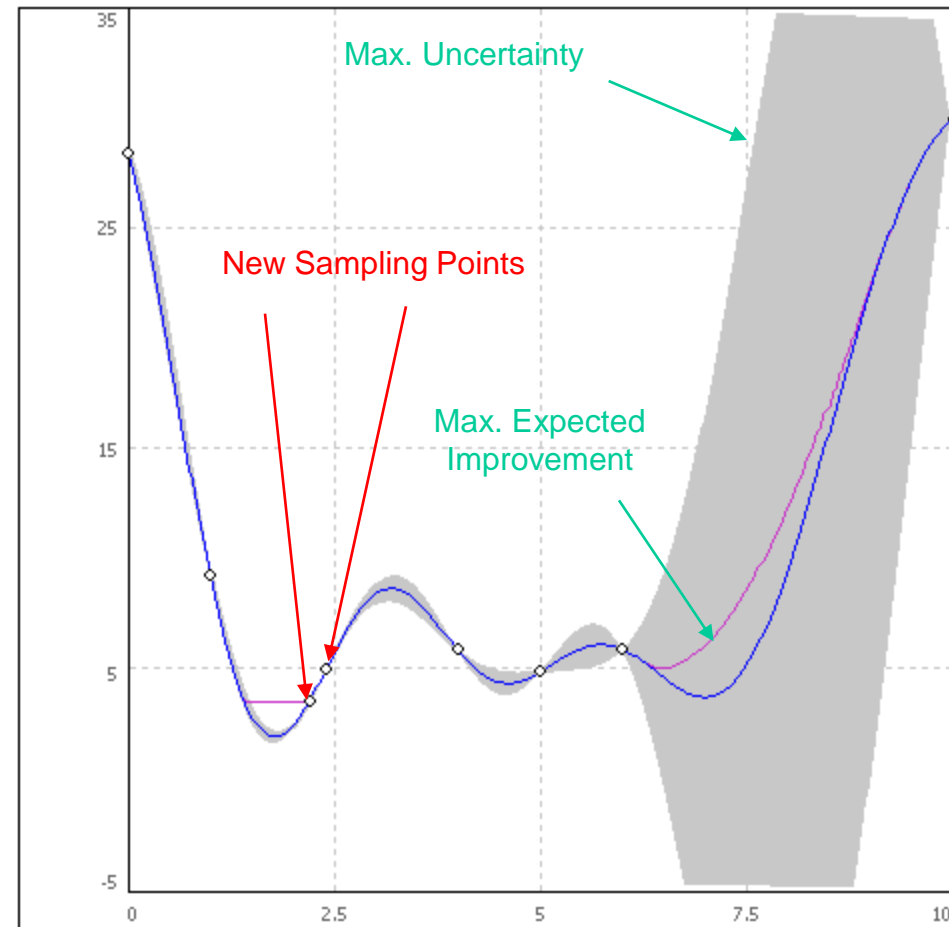


Approximation Loops

Start: 6 Sampling Points



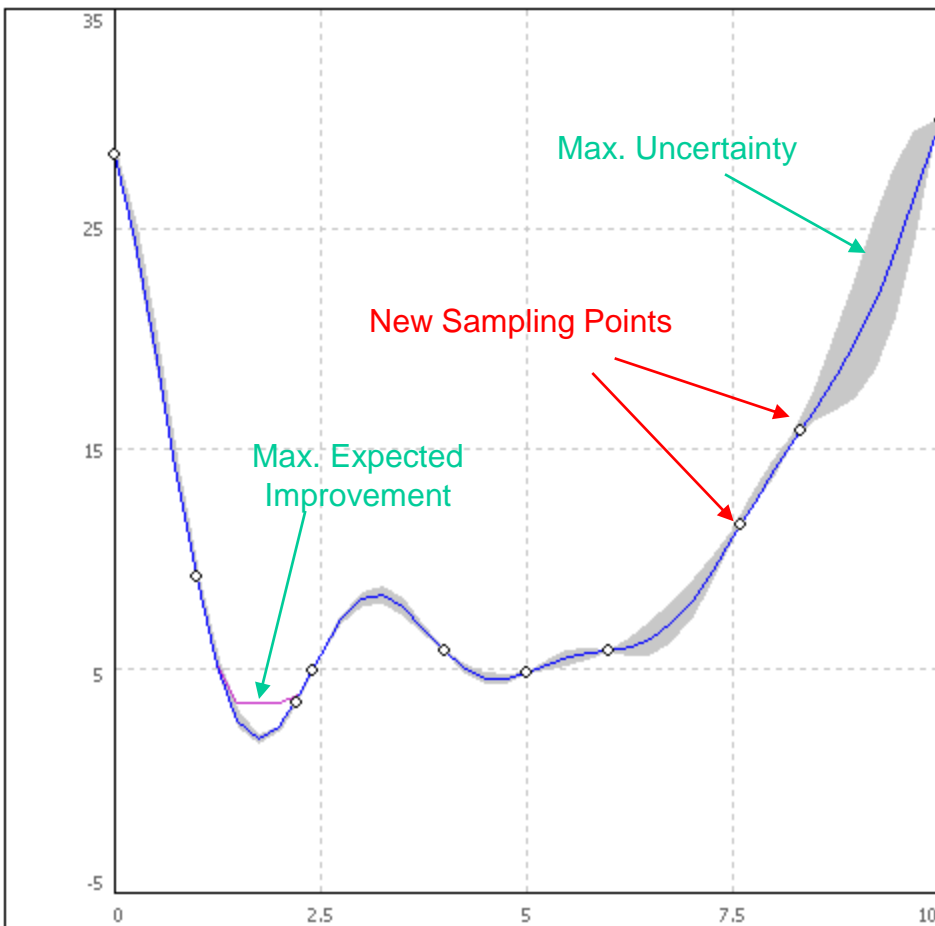
Loop 1



- Response Surface
- Expected Improvement EI
- 95% Confidence Interval

Approximation Loops

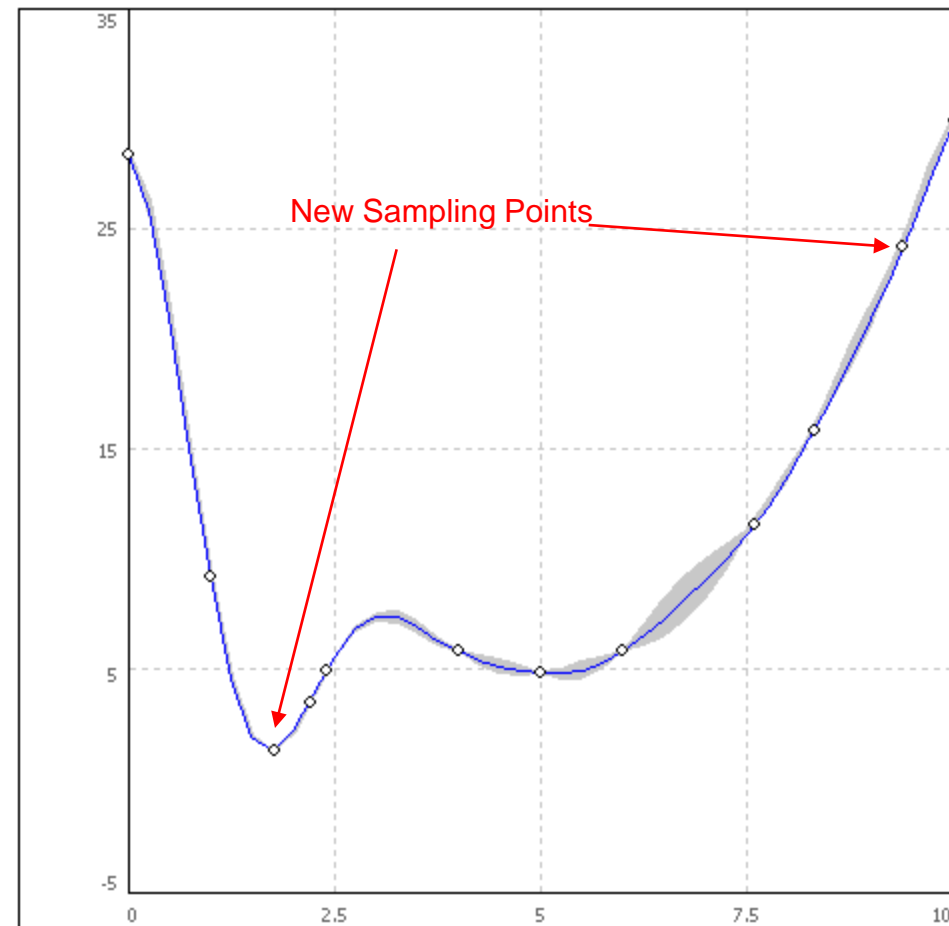
Loop 2



Stop-Criterion:

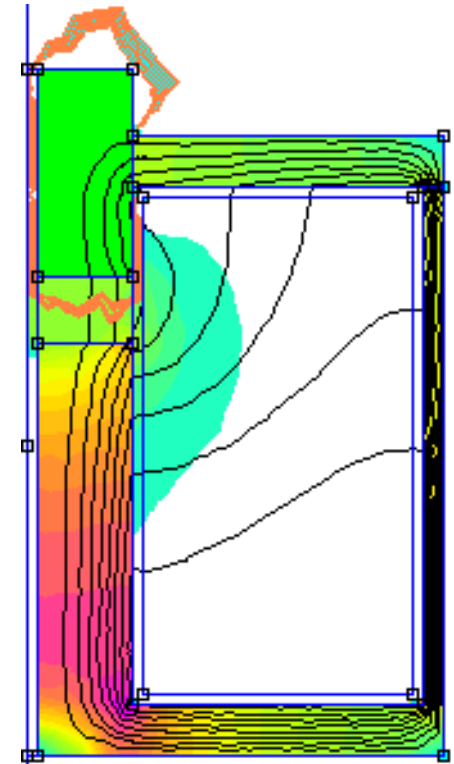
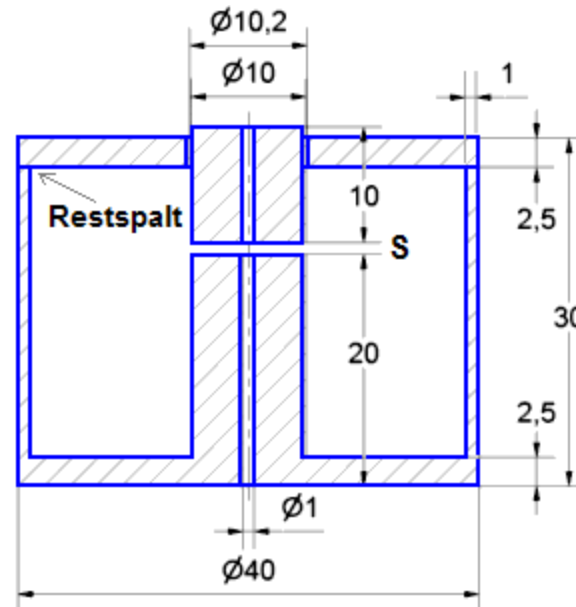
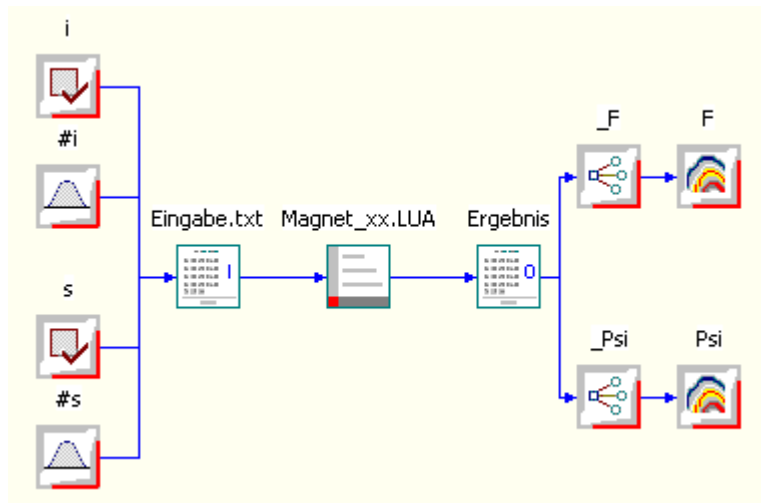
$$\text{Expected Improvement} < (Y_{\max} - Y_{\min}) / 100$$

Loop 3: Automatic Stop



Example: Electromagnetic Actuator

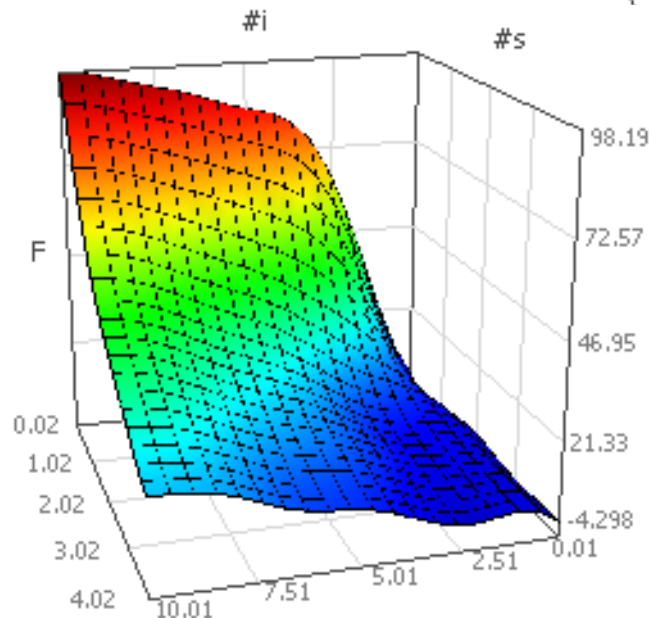
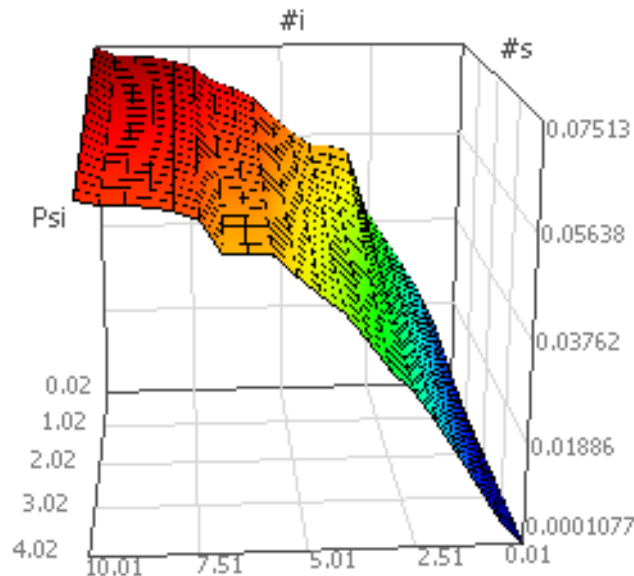
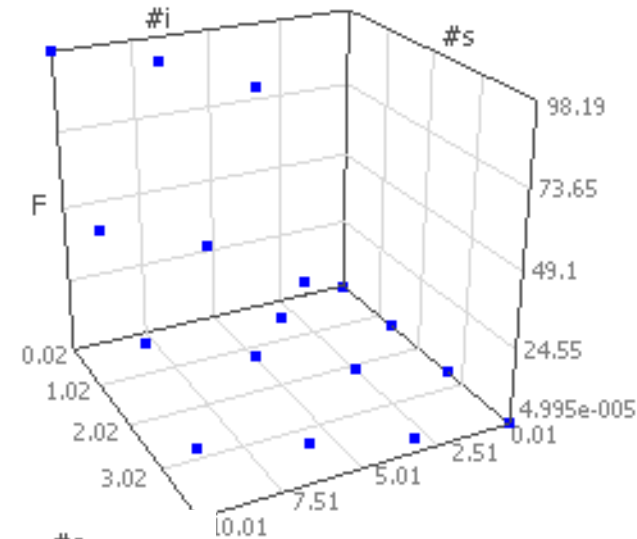
- Using the program **FEMM** and LUA script the parameterized model of a static pot magnet has been created (2D-axisymmetric).
- By means of OptiY the characteristic diagrams **$f(i,s)$** and **$\Psi(i,s)$** can be identified for **F** (force) and **Ψ** (coupling flux).
- It is shown how exported C code can be implemented in a SimulationX model as an electro-mechanical converter.



Identification of Characteristics

1. Optimal Setup of the Gaussian Process:

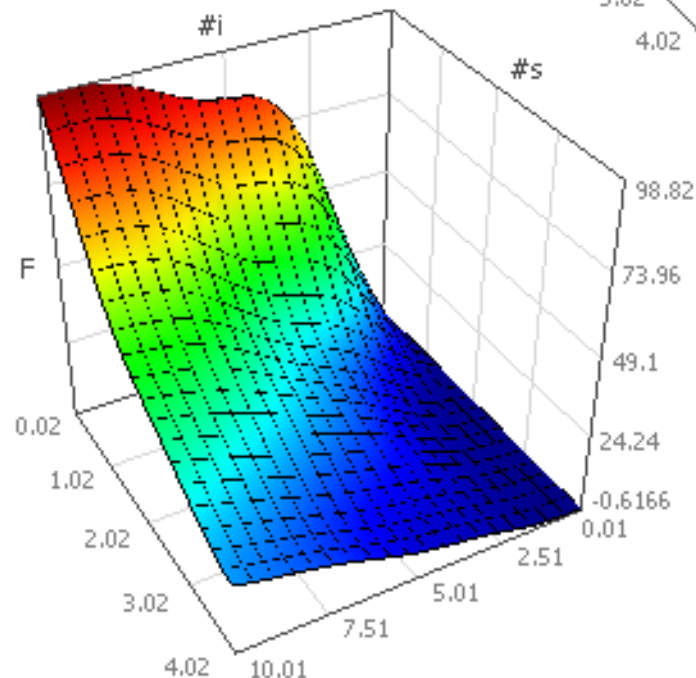
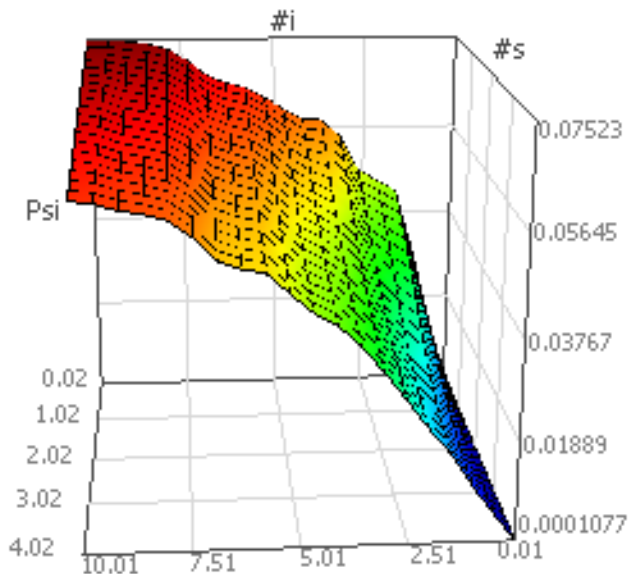
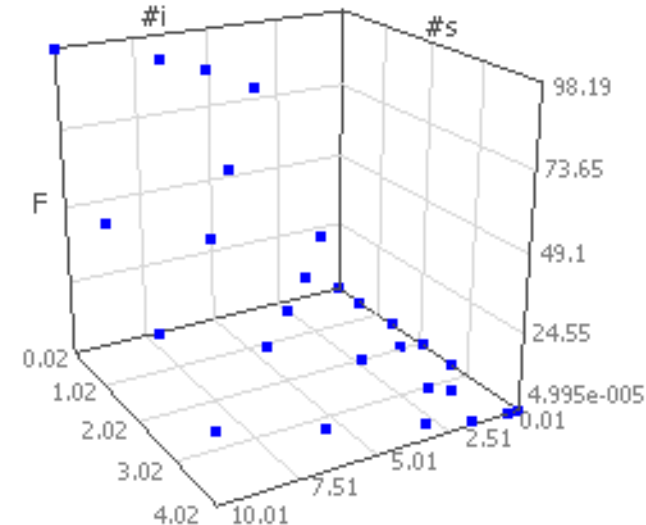
- Full Factory Design = Basic Grid \rightarrow **(4x4) +1**
- Covariance Function \rightarrow **Square Exponential**
- Approximation Order \rightarrow **2**



Identification of Characteristics

2. Adaptive Gaussian Process

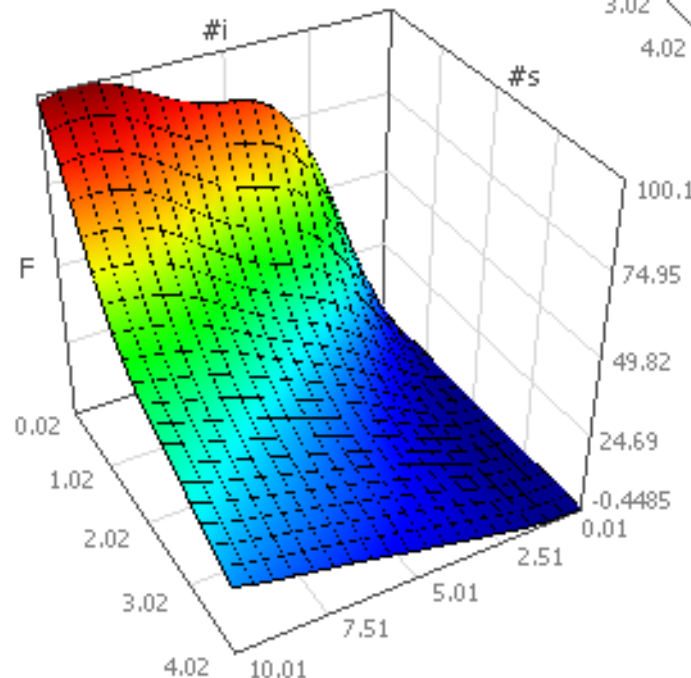
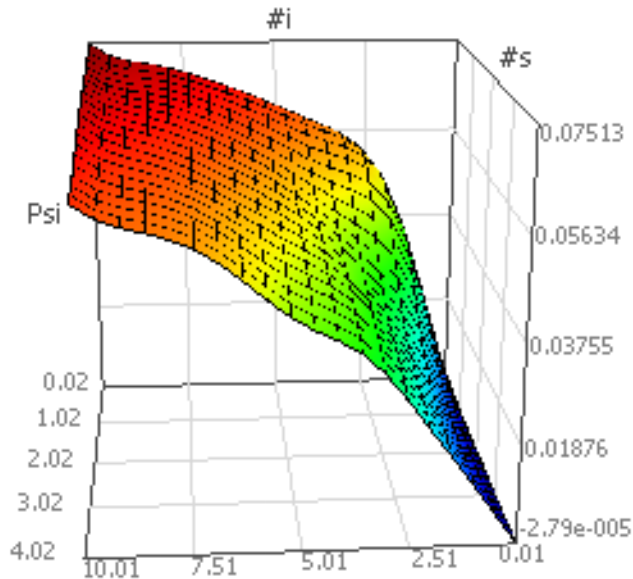
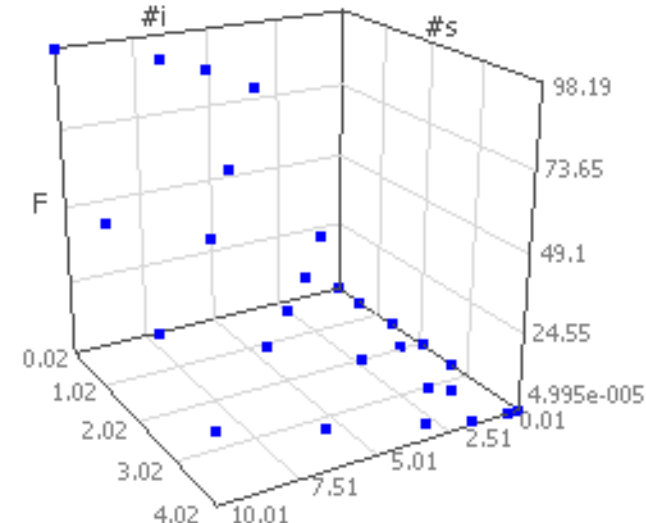
- In addition to the basic grid further scan points are calculated within larger confidence intervals.
- The previous optimal setup of the Gaussian process ensures a stable convergence of the adaptive process.



Identification of Characteristics

3. Fine Tuning of the Approximation Order (AO):

- AO = 2 ensures a stable convergence of the adaptive Gaussian process.
- Caused ripples in the identified response surfaces.
- The reduction to AO = 1 proved to be advantageous.



System Simulation based on the Characteristics

```
double Covariance(double x1[],double x2[],double p[])
{
    double Co, W;
    W = 0;
    for(int i = 0; i<2; i++) {
        W = W + (x1[i]-x2[i])*(x1[i]-x2[i])*p[i]*p[i];
    }
    Co = exp(-W);
    return Co;
}
```

```
double F(double i, double s)
{
    double p[2];
    double x1[2];
    double x2[2];
    double y = -46.7372056;
    y = y+10.5264863*pow(i,1);
    y = y+4.62081477*pow(s,1);
    p[0] = 0.151298213;
    p[1] = 0.928373134;
    x1[0] = i;
    x1[1] = s;
    x2[0] = 5.01;
    x2[1] = 2.02;
    y = y-183.986679*Covariance(x1,x2,p);
    x2[0] = 0.01;
    x2[1] = 0.02;
    y = y-8624.5598*Covariance(x1,x2,p);
    x2[0] = 2.01;
    x2[1] = 0.02;
    y = y+27677.7263*Covariance(x1,x2,p);
    :
    :
    x2[0] = 10.01;
    x2[1] = 4.02;
    y = y-1042.30105*Covariance(x1,x2,p);
    return y;
}
```

1. Code-Export (C-Code) and Creation of a DLL-File:

- GNU C compiler **gcc** in **cygwin**
- Only marginal changes of the C source code needed (Currently still **float** → **double**)
- Process can be automated using DOS batch commands:

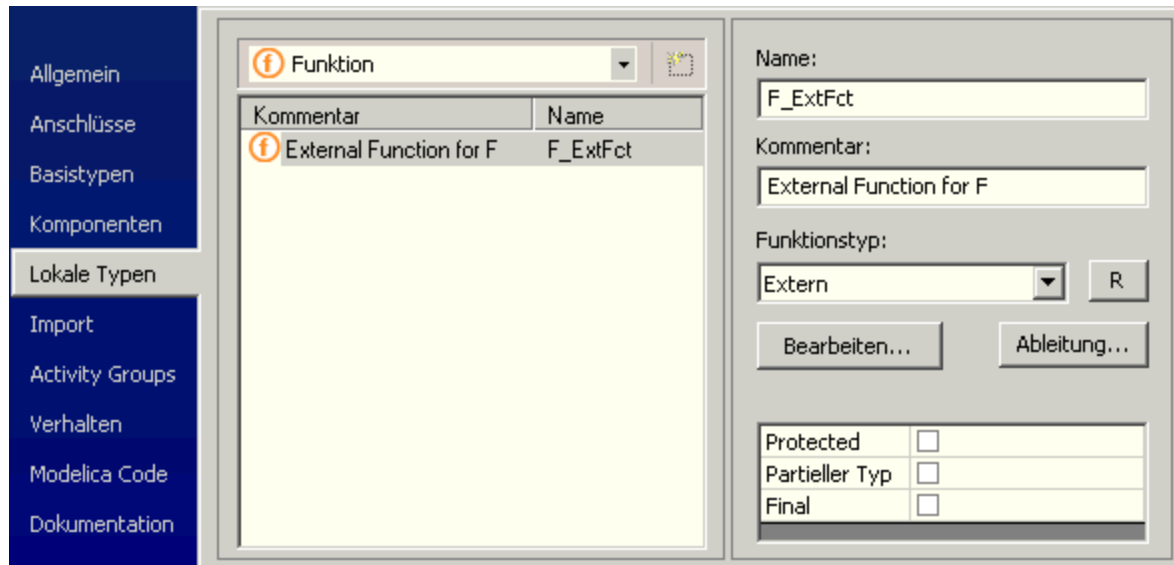
```
gcc-3 -mno-cygwin -shared -o Magnet_RSM.dll Magnet_RSM.c
```

- The DLL file has to be saved in a folder for SimulationX External functions:
Extras->Options->Directories->External Functions

System Simulation based on the Characteristics

2.1. External Function in User-Defined Element Type:

- Use of SimulationX *TypeDesigner* to define a custom converter type.
- Integration of external functions as local types in the user-defined element type.
- Definition of a function of type 'External' with name and comment.



- "Edit" opens a special SimulationX *TypeDesigner* for functions.

System Simulation based on the Characteristics

2.2. Definition of an interface to the external function:

The screenshot shows the 'SimulationX TypeDesigner (Funktion) - Modell2.F_RSM.F_ExtFct' window. It is divided into several sections:

- Argumente:** A table listing output parameters:

Kommentar	Name
Strom	i
Luftspalt [mm]	s
Koppelfluss	F_Output
- Properties Panel (Right):**
 - Name: s
 - Kommentar: Luftspalt [mm]
 - Typ: Real (R)
 - Physikalische Größe: Abmessungen
 - Dimension: Skalar
 - Deklarationsgleichung: mm
 - Variabilität: Variable
 - Diskret:
 - In-/Output?: Input
- Configuration Panel (Bottom Left):**
 - Externe Bibliothek: Magnet_RSM.dll
 - Aufrufkonvention: C / C++
 - Funktionsaufruf: F_Output=F(i, s)

System Simulation based on the Characteristics

3.1. Electromechanical Converter (mechanical side):

The screenshot shows a simulation software interface with a hierarchical tree of components on the left and a code editor on the right.

Component Tree:

- R_Spule (Spulenwiderstand)
 - s (Luftspalt)
 - i (Strom)
 - F (Kraft)
 - Psi (Koppelfluss)
 - u_ind (Induzierte Spannung)
 - uR (Spannung an R_Spule)
 - u (Spannung an Spule)
- ctr1 (Mechanischer Anschluss (translatorisch))
 - x (Weg)
 - v (Geschwindigkeit)
 - a (Beschleunigung)
 - F (Externe Kraft)
- ctr2 (Mechanischer Anschluss (translatorisch))
 - x (Weg)
 - v (Geschwindigkeit)
 - a (Beschleunigung)
 - F (Externe Kraft)
- pin1 (Elektrischer Anschluss)
- pin2 (Elektrischer Anschluss)
- F_ExtFct (External Function for F)
 - i (Strom)
 - s (Luftspalt)
 - F_Output (Kraft)
- Psi_ExtFct (External Function for Psi)

Code Editor:

```

1 // enter your algorithm here
2 s := ctr1.x-ctr2.x;
3 F := F_ExtFct(i,s'm->mm');
4 ctr1.F := F;
5 ctr2.F :=-ctr2.F;
6
    
```

The interface includes a menu on the left with options like 'Allgemein', 'Anschlüsse', 'Basistypen', 'Komponenten', 'Lokale Typen', 'Import', 'Activity Groups', 'Verhalten', 'Modelica Code', and 'Dokumentation'. The top toolbar contains various icons for navigation and editing. The bottom of the code editor has tabs for 'Algorithmus' and 'Gleichungen'.

System Simulation based on the Characteristics

3.1. Electromechanical Converter (electrical side):

The screenshot shows a simulation software interface with a hierarchical tree of components on the left and a code editor on the right. The tree includes electrical and mechanical components, and the code editor contains MATLAB-style equations for an electromechanical converter.

Component Tree:

- R_Spule (Spulenwiderstand)
- s (Luftspalt)
- i (Strom)
- F (Kraft)
- Psi (Kopplfluss)
- u_ind (Induzierte Spannung)
- uR (Spannung an R_Spule)
- u (Spannung an Spule)
- ctr1 (Mechanischer Anschluss (translatorisch))
 - x (Weg)
 - v (Geschwindigkeit)
 - a (Beschleunigung)
 - F (Externe Kraft)
- ctr2 (Mechanischer Anschluss (translatorisch))
 - x (Weg)
 - v (Geschwindigkeit)
 - a (Beschleunigung)
 - F (Externe Kraft)
- pin1 (Elektrischer Anschluss)
- pin2 (Elektrischer Anschluss)
- F_ExtFct (External Function for F)
- Psi_ExtFct (External Function for Psi)
 - i (Strom)
 - s (Luftspalt)
 - Psi_Output (Kopplfluss)

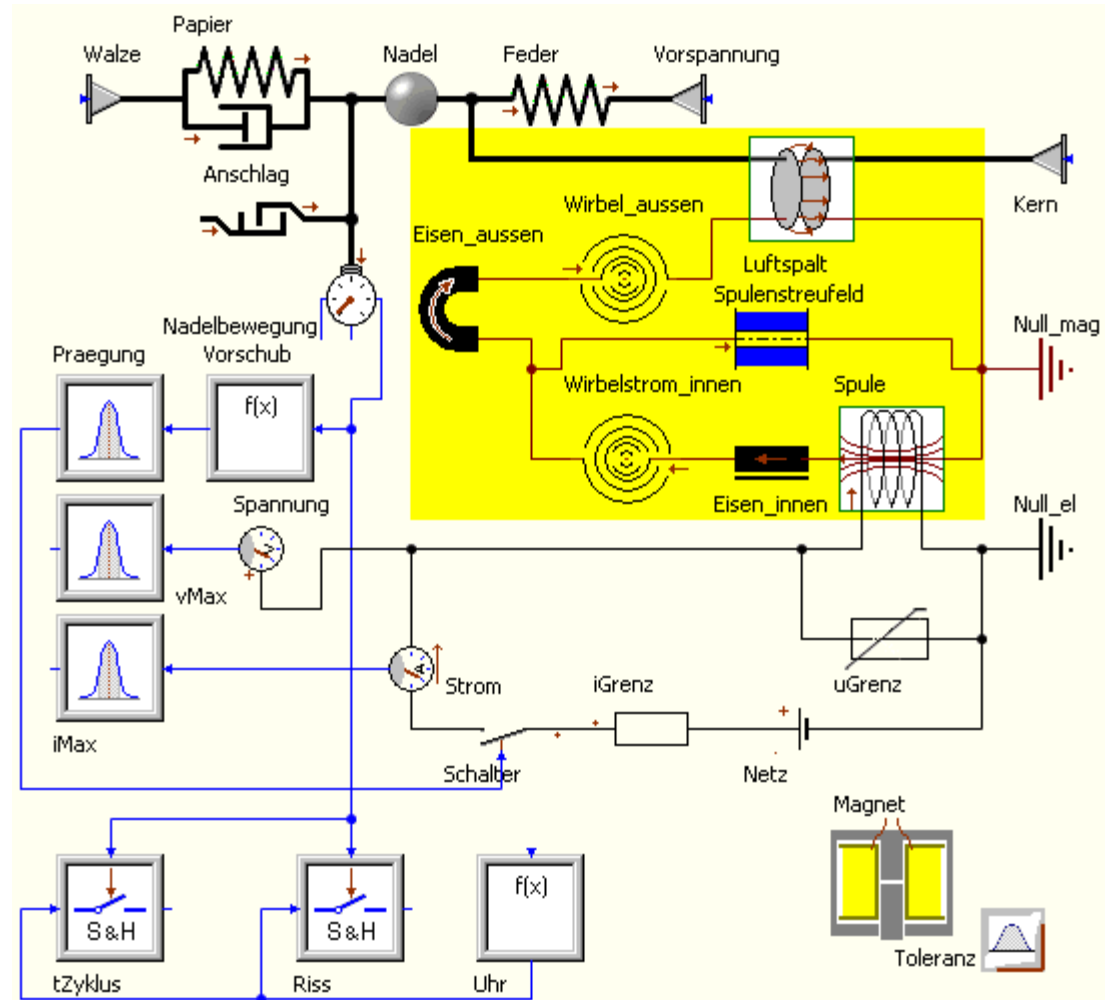
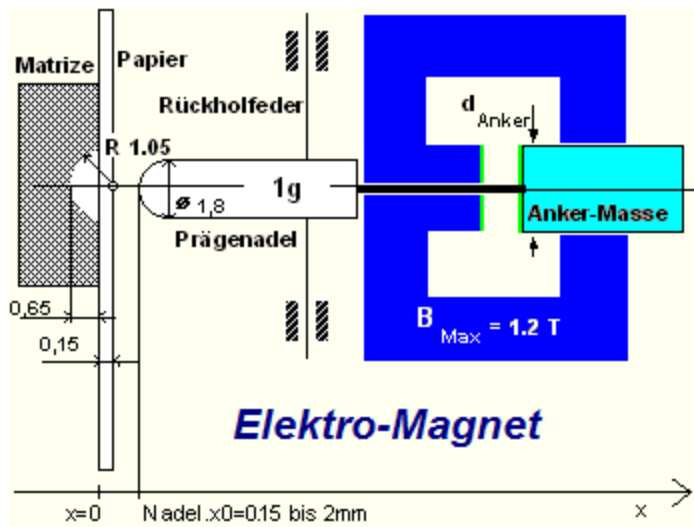
Code Editor (Equations):

```

1 // enter your equations here
2 u      = pin1.v - pin2.v;
3 Psi    = Psi_ExtFct(i, s'm->mm');
4 u_ind  = der(Psi);
5 uR     = R_Spule * i;
6 u      = u_ind + uR;
7 pin1.i = i;
8 pin2.i = -pin1.i;
9
    
```


System Simulation based on the Characteristics

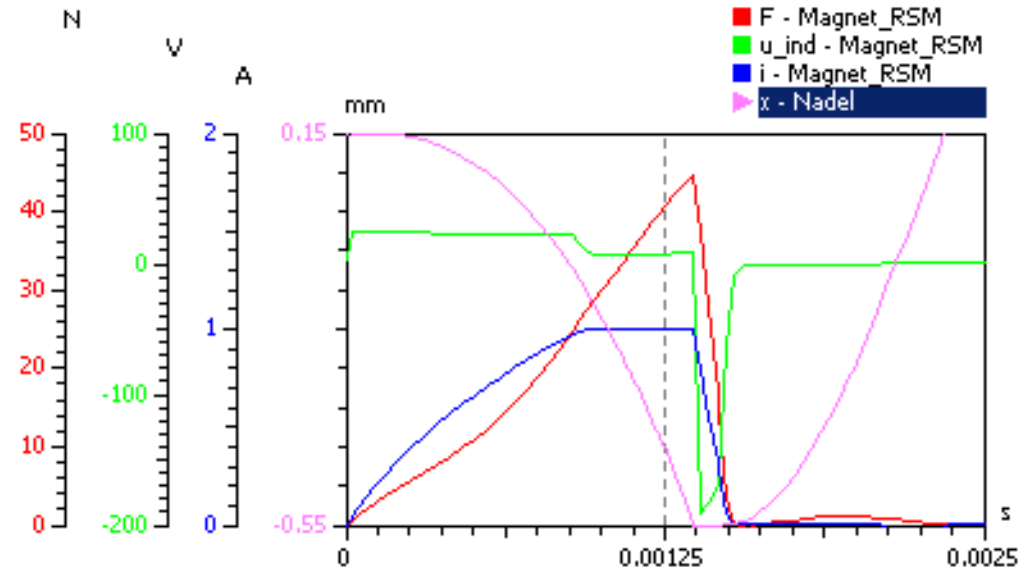
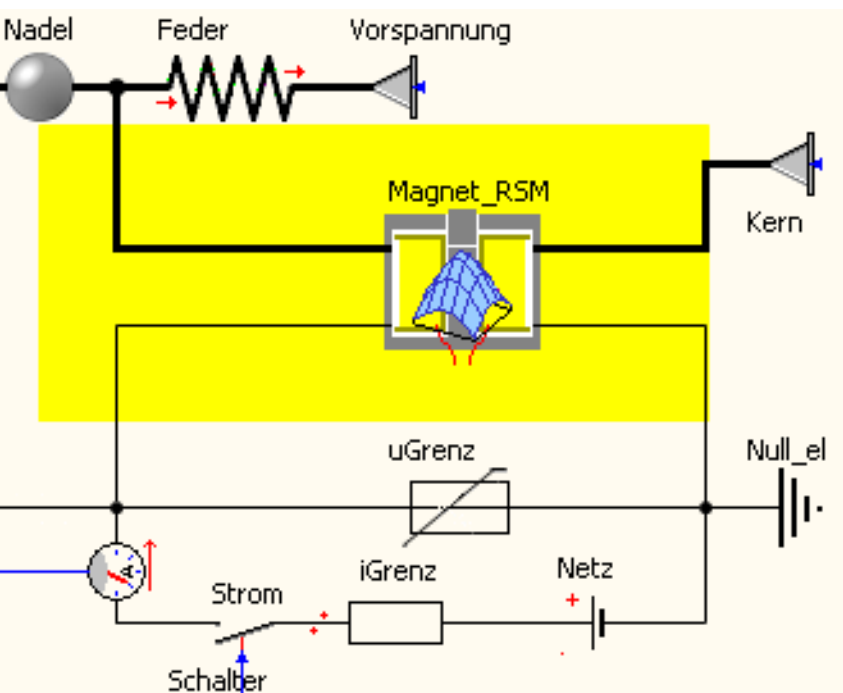
4.1. Network model of an electromagnetic actuator for the Braille blind embossing



System Simulation based on the Characteristics

4.2. Network model of an electromagnetic actuator based on the characteristic diagrams:

- The characteristic diagrams using response surfaces of a FE calculation vary accurately reflect the transmission behavior even for a complicated converter geometry.
- All model parameters except the converter geometry can be varied in simulations.



Conclusion

- Modeling and simulation of a real product or process is infeasible because of unknown relationships or time-consuming computations. Current solutions for solving this problem apply model reduction to network elements, which is very time- and cost-intensive
- The new approach in OptiY® allows a fast and simple extraction of a mathematical surrogate model from measurement data or complex FEA. The adaptive process can suggest new sampling points for the measurement or model calculation to build the surrogate model more accurately.
- The exported C code can be included as an external function in appropriately designed element types of SimulationX models.
- The surrogate model of complex components can be used as part of a complex system model.
- The advantages of network-based system simulation can be combined with the advantages of the FEA and the measurement of real objects throughout the design process.