# Meta-Modellierung mit OptiY®
## Gewinnung mathematischer Ersatzmodelle aus Messdaten oder komplexen FEA

Dr. The-Quan Pham
OptiY e.K.
Aschaffenburg

Dr. Alfred Kamusella
Institut für Feinwerktechnik und Elektronik-Design
TU Dresden

ITI Symposium
26.-27. November 2009 in Dresden

# Gliederung

1. Problemstellung
2. Theoretische Grundlagen
   - Gauß-Prozess
   - Adaptiver Gauß-Prozess (Visualisierung)
3. Praktisches FEA-Beispiel
   - Elektromagnetischer Antrieb
   - Kennfeld-Identifikation F=f(i,s) & Psi=f(i,s)
   - System-Simulation mit Kennfeldern

# Problemstellung

**Simulation eines realen Produktes oder Prozesses**

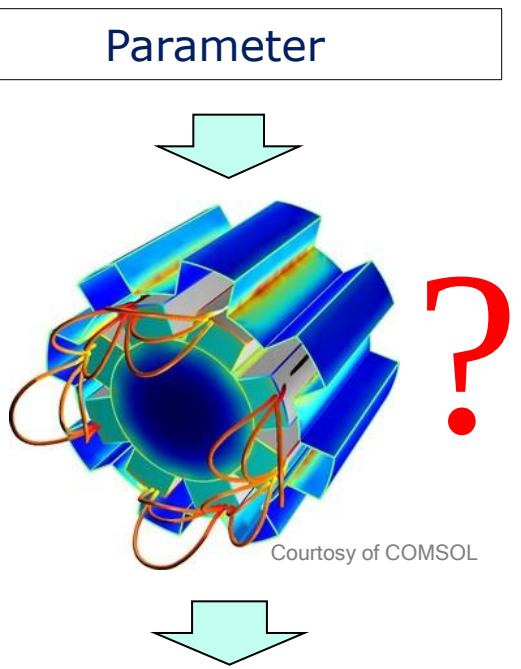
- Zusammenhänge oder Wirkprinzipe sind unbekannt
- Es existieren bisher nur Messdaten
- Was soll modelliert werden?

**Komplexes Finite Element Modell**

- Lange Rechenzeiten: Stunden oder Tage
- Ansteuerung/Regelkreis: Schleifen von Simulationen
- Test-Szenarien: erfordern ebenfalls viele Simulationen
- Technische Machbarkeit der Systemsimulation?

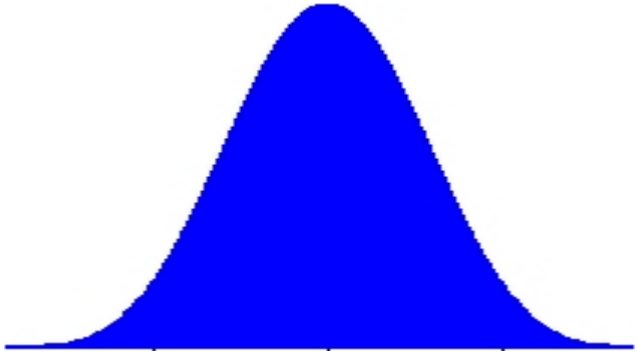
**Stand der Technik**

- Modell-Reduktion auf Netzwerk-Elemente
- Nur mathematische Beschreibung der Zusammenhänge
- Suche nach geeigneten Modellstrukturen und Parametern
- Parameter-Validierung ist erforderlich
- Zeit- und Kostenintensiv



# Gauß-Prozess

- Polynom f(**x**) mit *p*-ter Ordnung als globale Anpassung
- Stochastischer Prozess Z(**x**) als lokale Anpassung



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

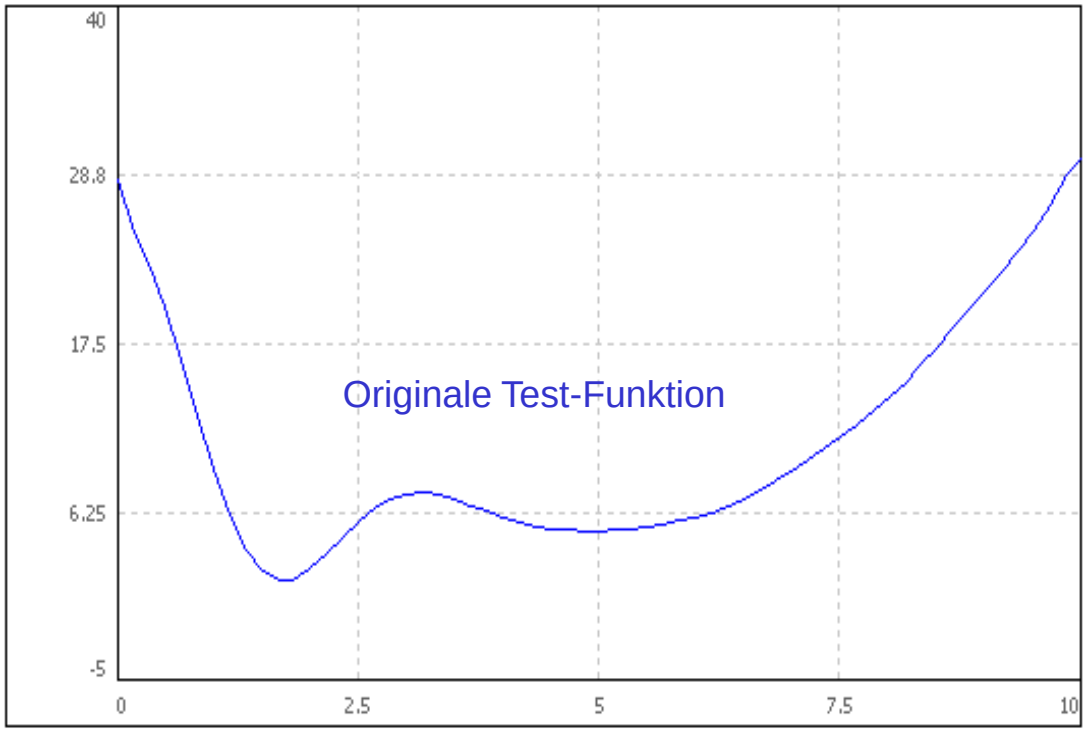
**Korrelation Funktion** R**(x)**

- Multivariate Gauß-Verteilung (Normal-Verteilung)
- Interpolation zwischen berechneten Punkten
- Wechselwirkung zwischen den einzelnen Parametern

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

# Adaptiver Gauß-Prozess (Visualisierung)

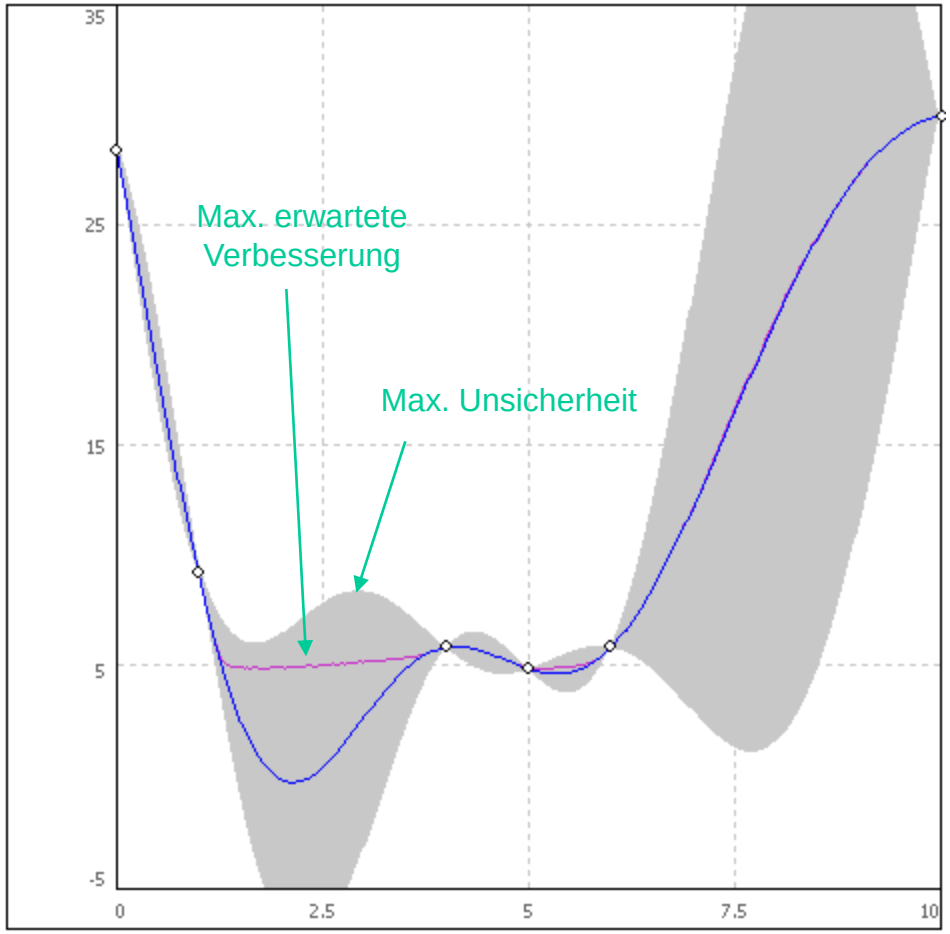
$$Y = (X-5)^2 - 15 \cdot e^{-(X-1.5)^2} + 5$$



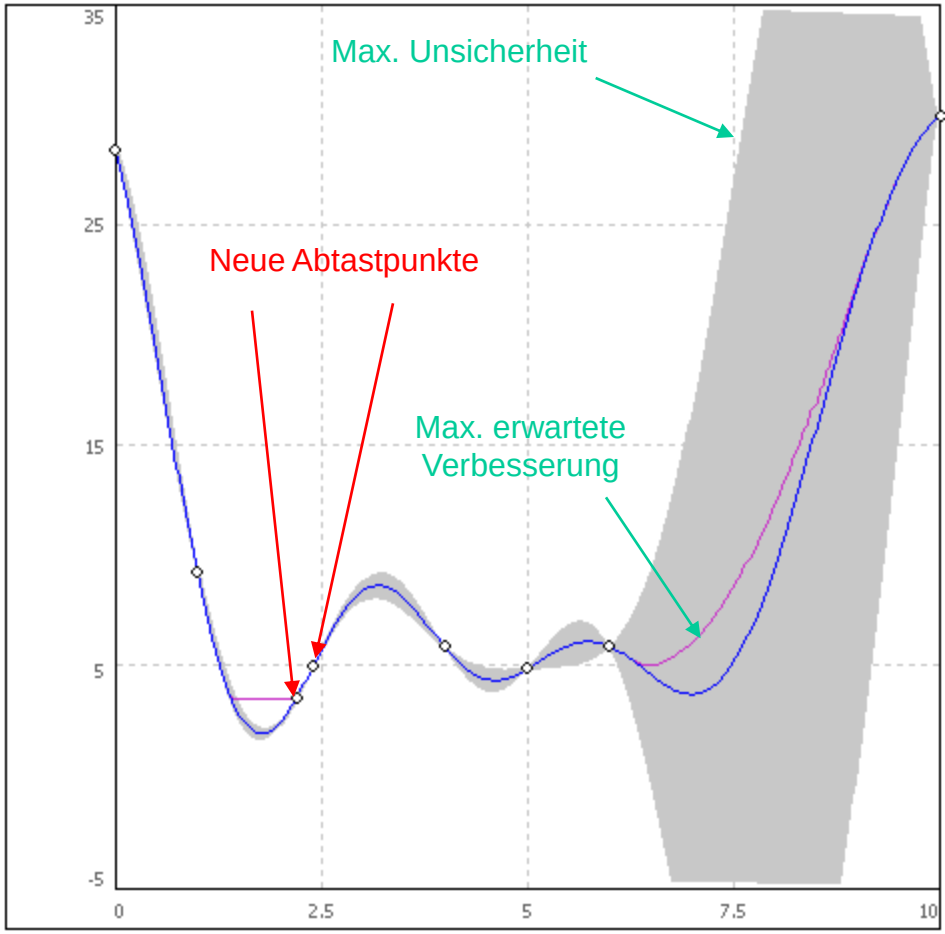
# Approximationsschleifen

Antwortfläche
Erwartete Verbesserung EI
95% Vertrauensintervall

**Start: 6 Abtastpunkte**



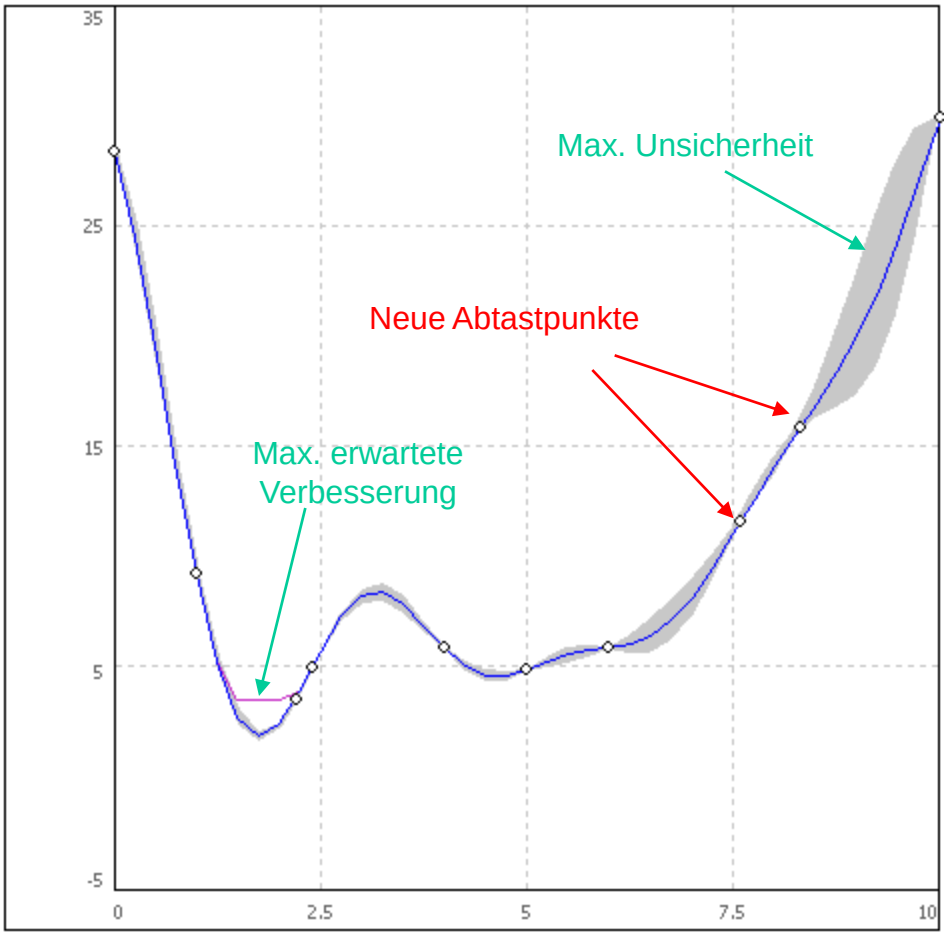

**Schleife 1**



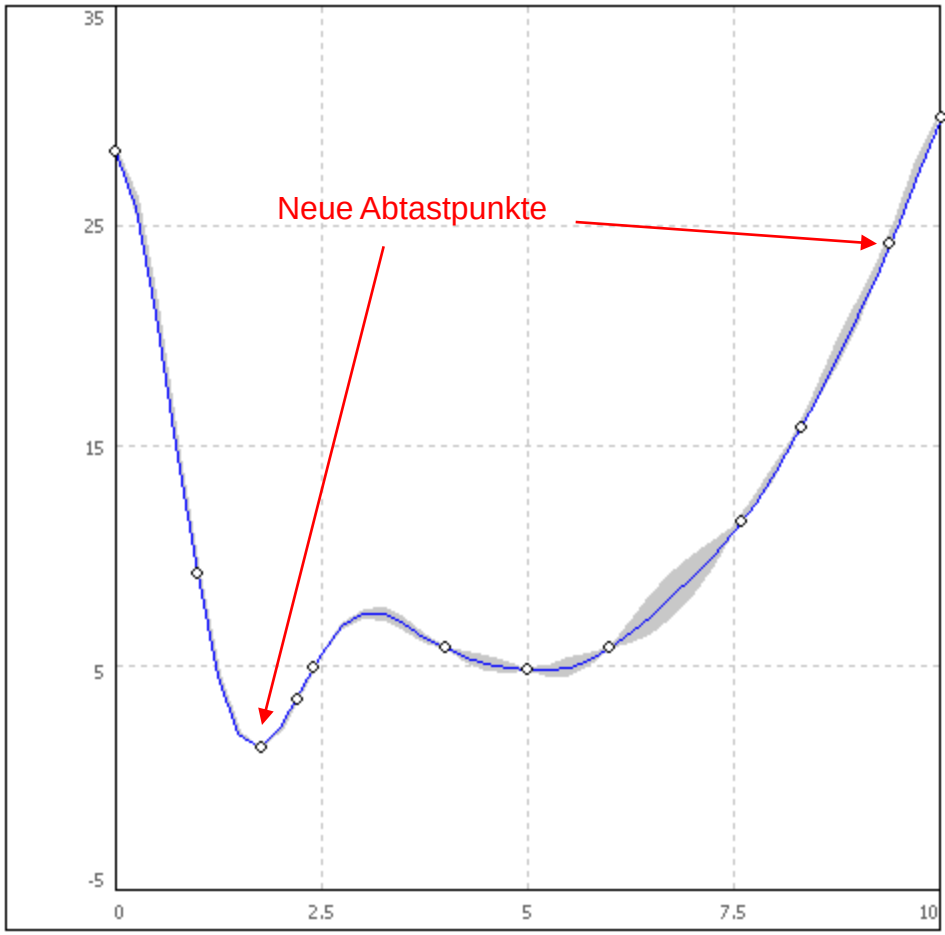
# Approximationsschleifen

**Stopp-Kriterium:**
**Erwartete Verbesserung < $(Y_{max}-Y_{min})/100$**

**Schleife 2**

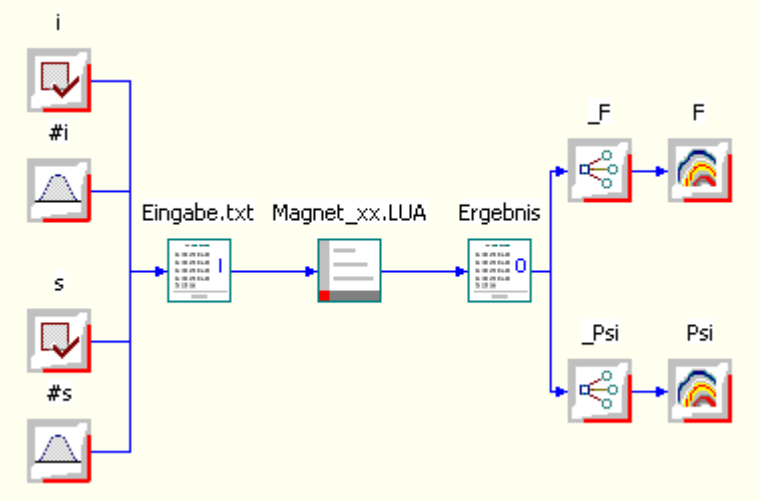


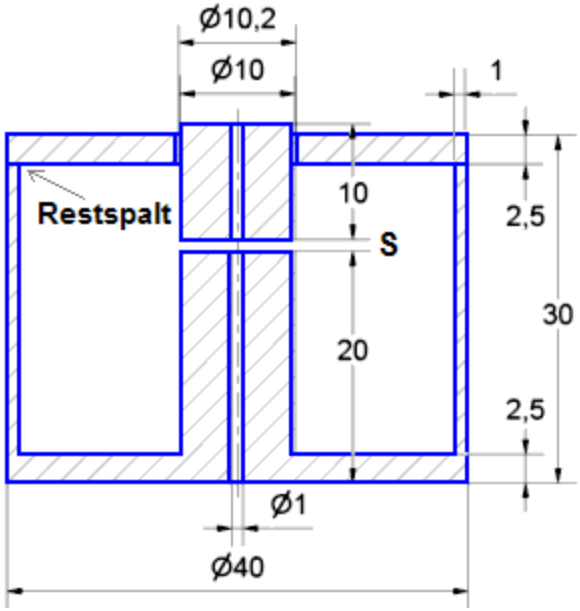

**Schleife 3: automatischer Abbruch**

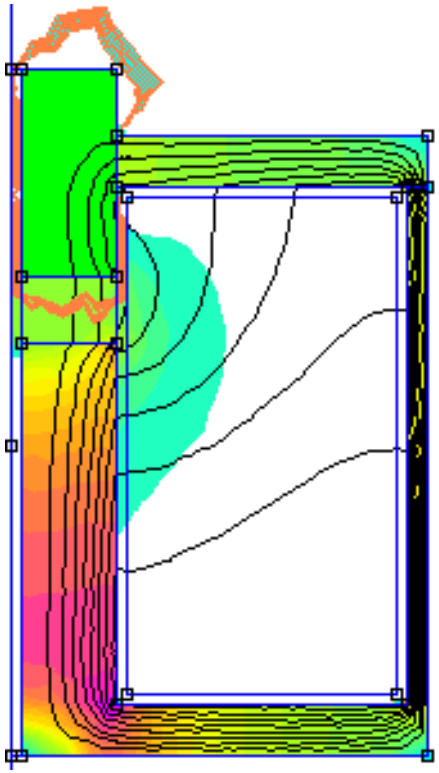


# Beispiel: Elektromagnetischer Antrieb

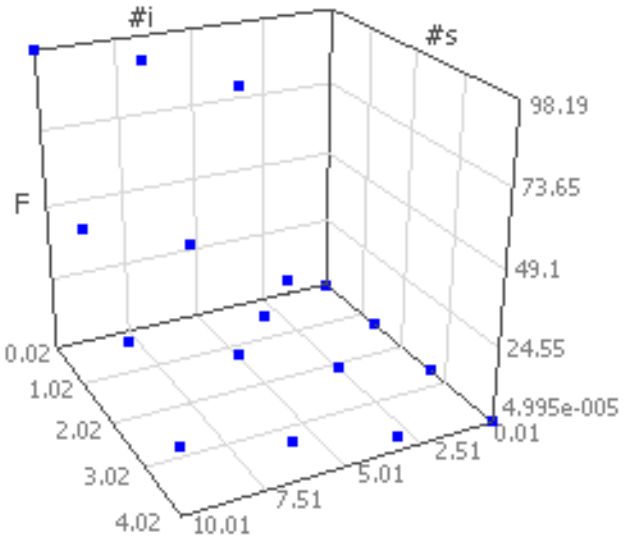
- Mit dem Programm **FEMM** wurde mittels LUA-Script das parametrisierte statische Modell eines Topf-Magneten erstellt (2D-Axialsymmetrisch).
- Mit **OptiY** kann man für **F** (Kraft) und **Psi** (Koppelfluss) die Kennfelder **F(i,s)** und **Psi(i,s)** identifizieren.
- Es wird gezeigt, wie der exportierte C-Code in einem SimulationX-Modell als elektro-mechanischer Wandler implementiert werden kann.



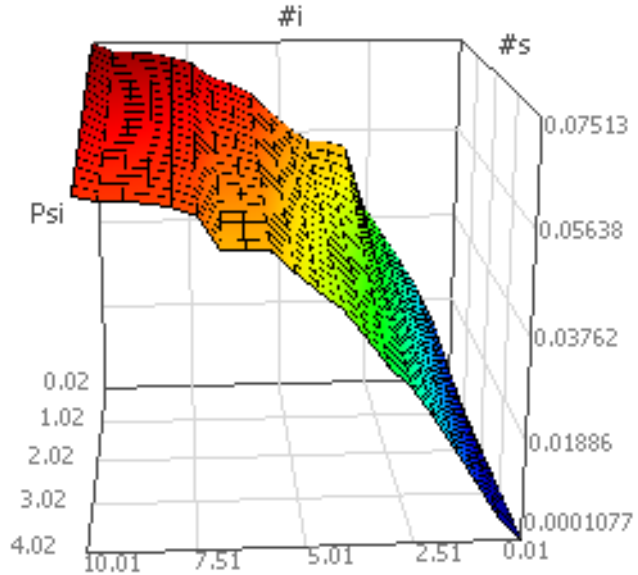



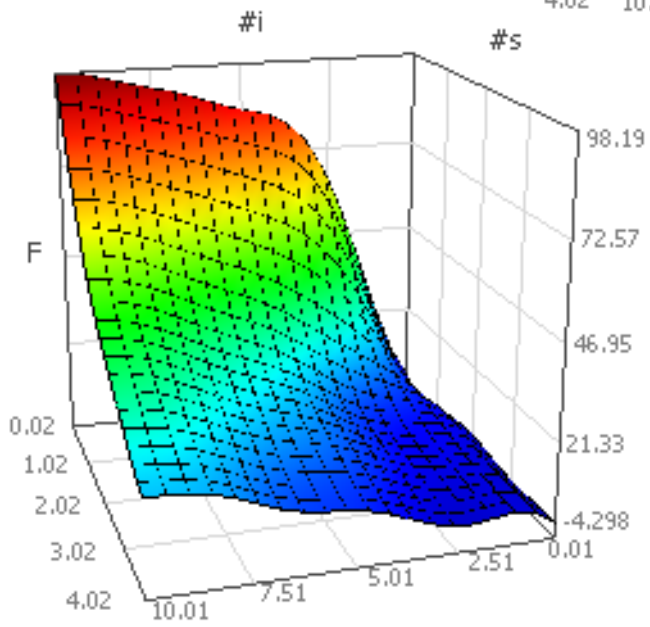
# Kennfeld-Identifikation F & Psi=f(i,s)



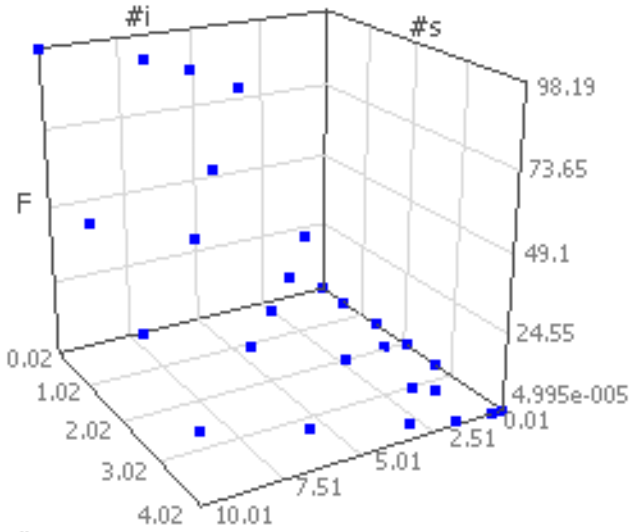

**1. Gauß-Prozess optimal konfigurieren:**

- „Full Factory Design“ = Grundraster → **(4x4)** +1
- Covarianz-Funktion → **Square Exponential**
- Approximationsordnung → **2**



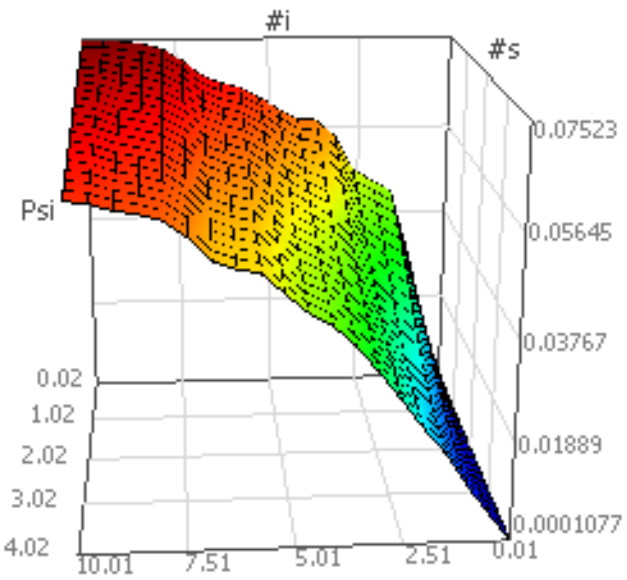



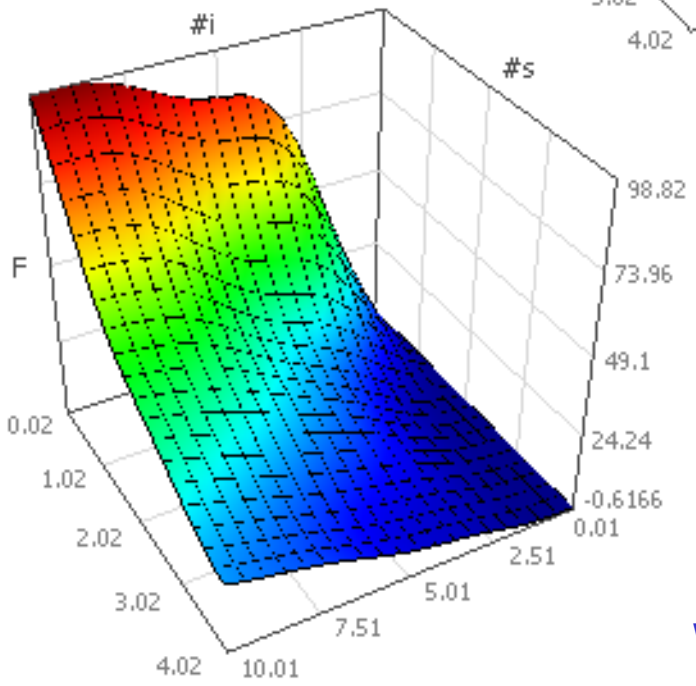
# Kennfeld-Identifikation F & Psi=f(i,s)



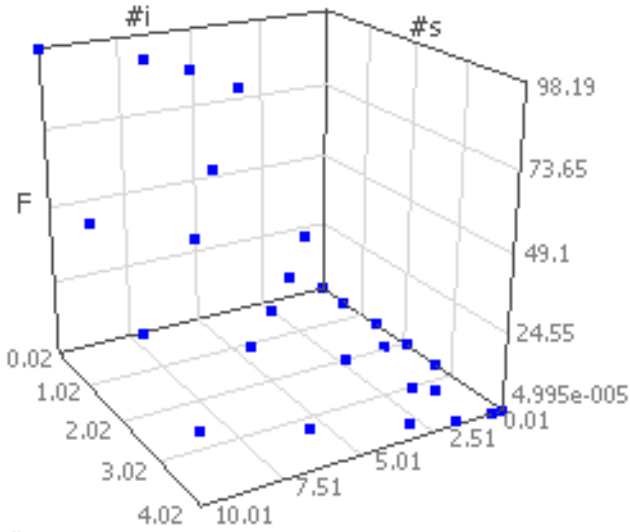

**2. Adaptiver Gauß-Prozess**

- Ergänzend zum groben Grundraster werden zusätzliche Abtastpunkte in „unsicheren“ Bereichen berechnet.
- Die vorherige optimale Konfiguration ermöglicht eine sichere Konvergenz.



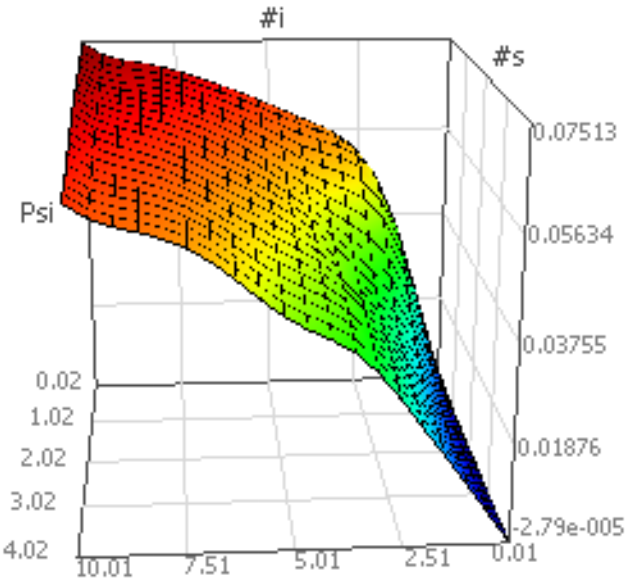



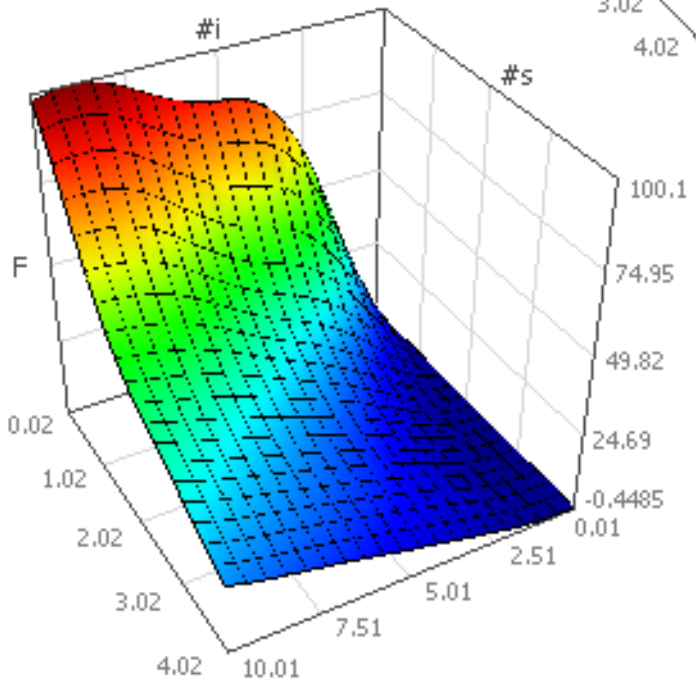
# Kennfeld-Identifikation F & Psi=f(i,s)




**3. Feinabgleich der Approximationsordnung (AO):**

- AO=2 bewirkte eine sichere Konvergenz des adaptiven Gauß-Prozesses.
- Führte zu Welligkeiten in identifizierten Kennfeldern.
- Reduktion auf AO=1 erwies sich als günstig.



# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

```
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;
}
```

**1. Modell-Export (C-Code) und DLL-Erzeugung:**

- Freier GNU-C-Compiler **gcc** in **Cygwin**
- Nur geringfügige Anpassungen der C-Quelle erforderlich (Zur Zeit noch float → double)
- Prozess automatisierbar durch Kommandozeile in BAT-File:

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

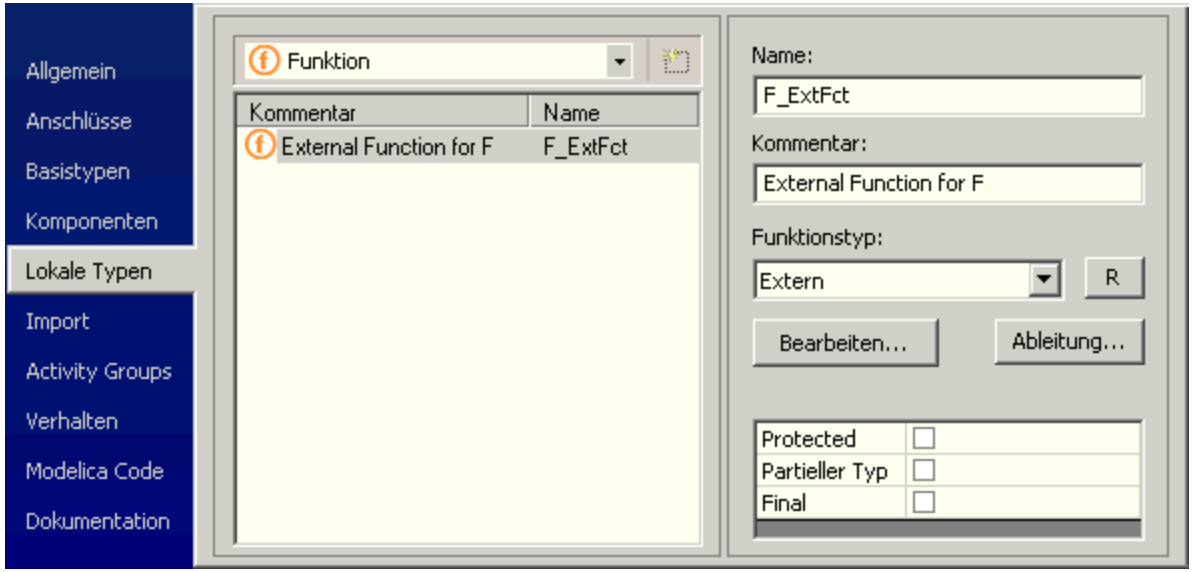
- Die .DLL-Datei muss in einen SimulationX-Ordner für Externe Funktionen abgelegt werden:
  *Extras->Optionen->Verzeichnisse->Externe Funktionen*

```
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] = 0.161298213;
    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;
}
```

# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

**2.1. „External Function“ in eigenem Element-Typ:**

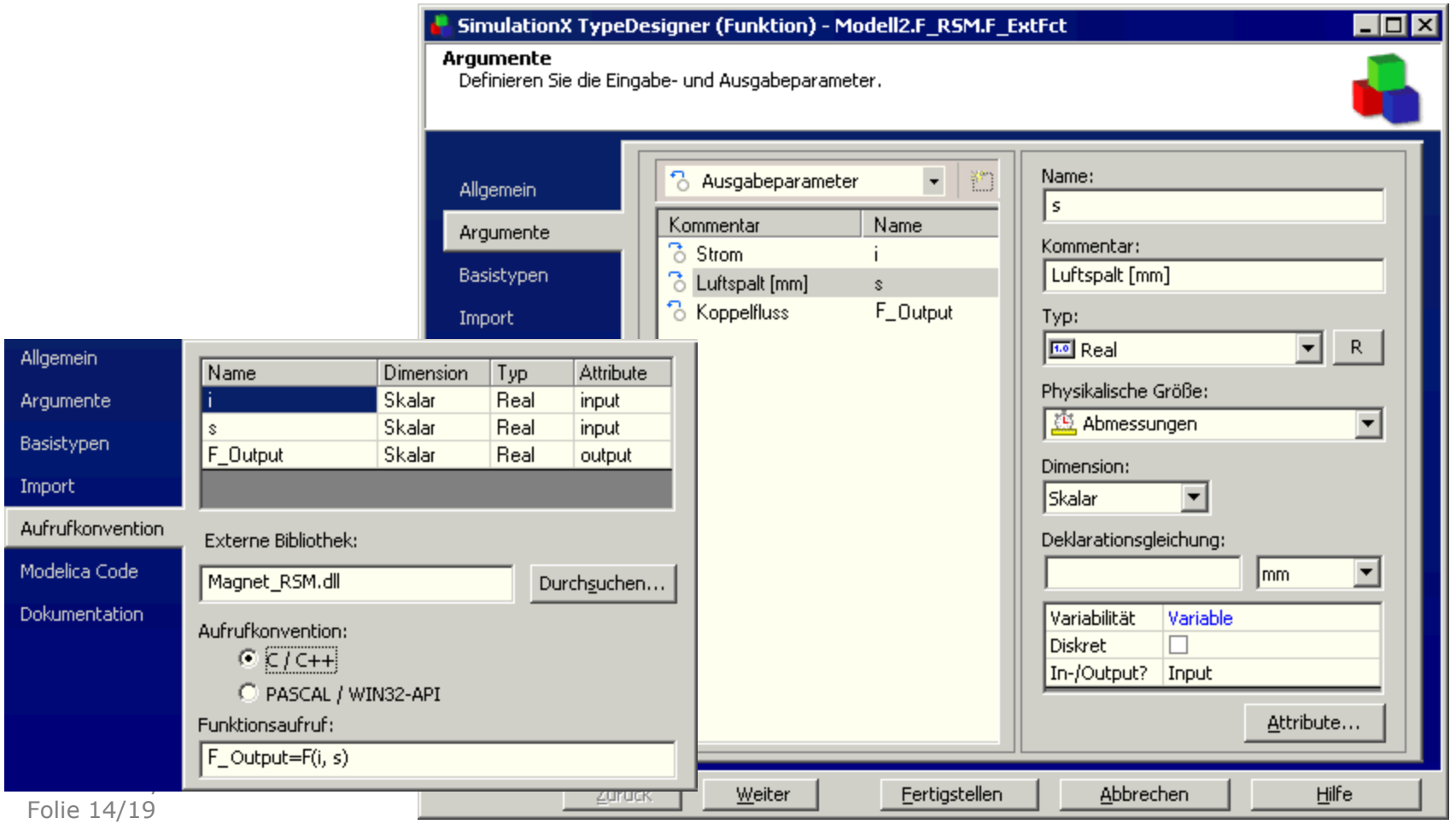
- Nutzung des SimulationX *TypeDesigner* zur Definition eines eigenen Wandler-Types.
- Einbindung einer externen Funktion als *Lokalen Typen* im eigenen „Kennfeld“-Wandler.
- Definition einer Funktion vom Typ „Extern“ mit Name und Kommentar.




- „Bearbeiten“ öffnet einen speziellen *TypeDesigner* für Funktionen.

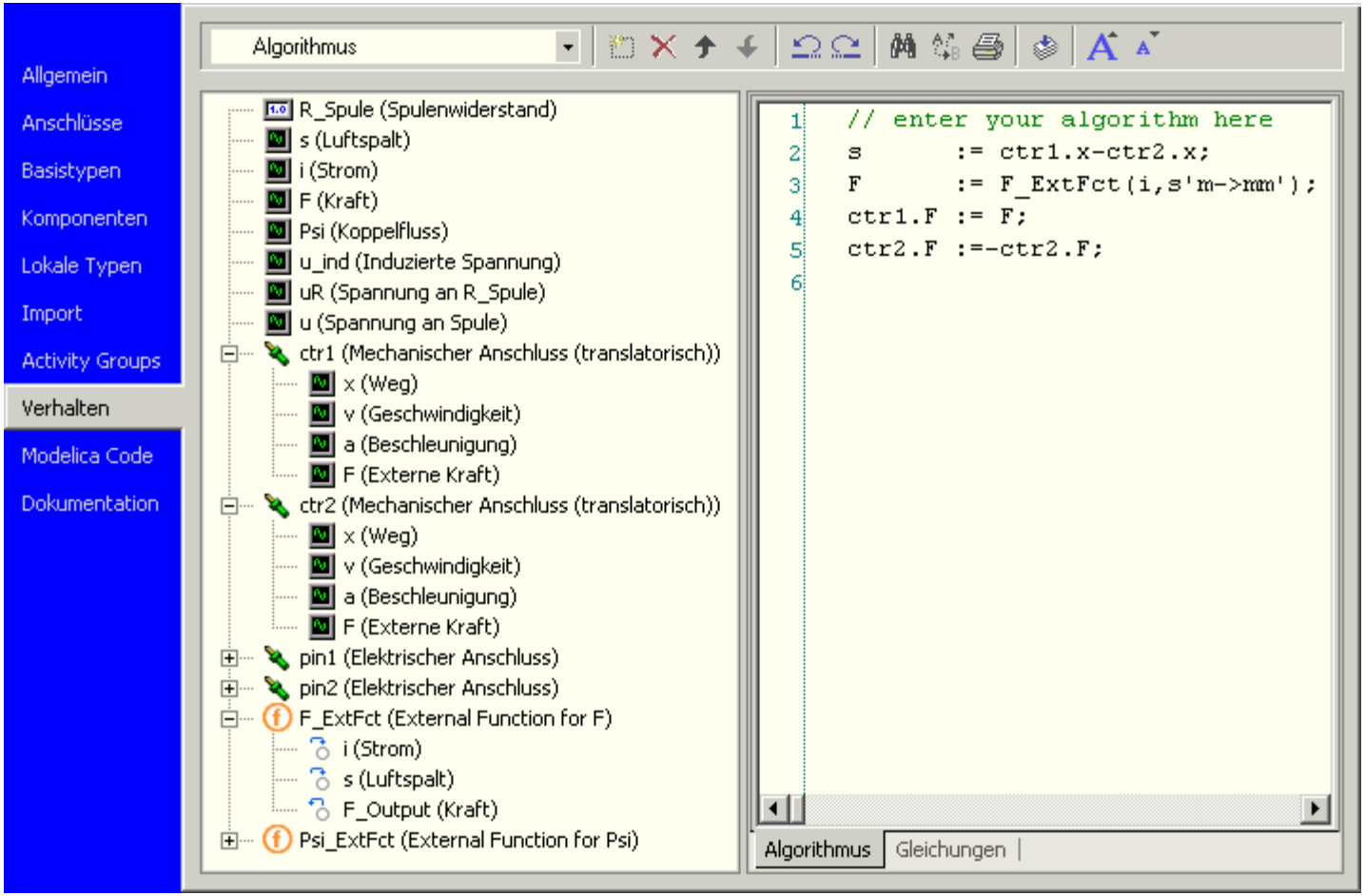
# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

## 2.2. Definition der Schnittstellen zur externen Funktion:



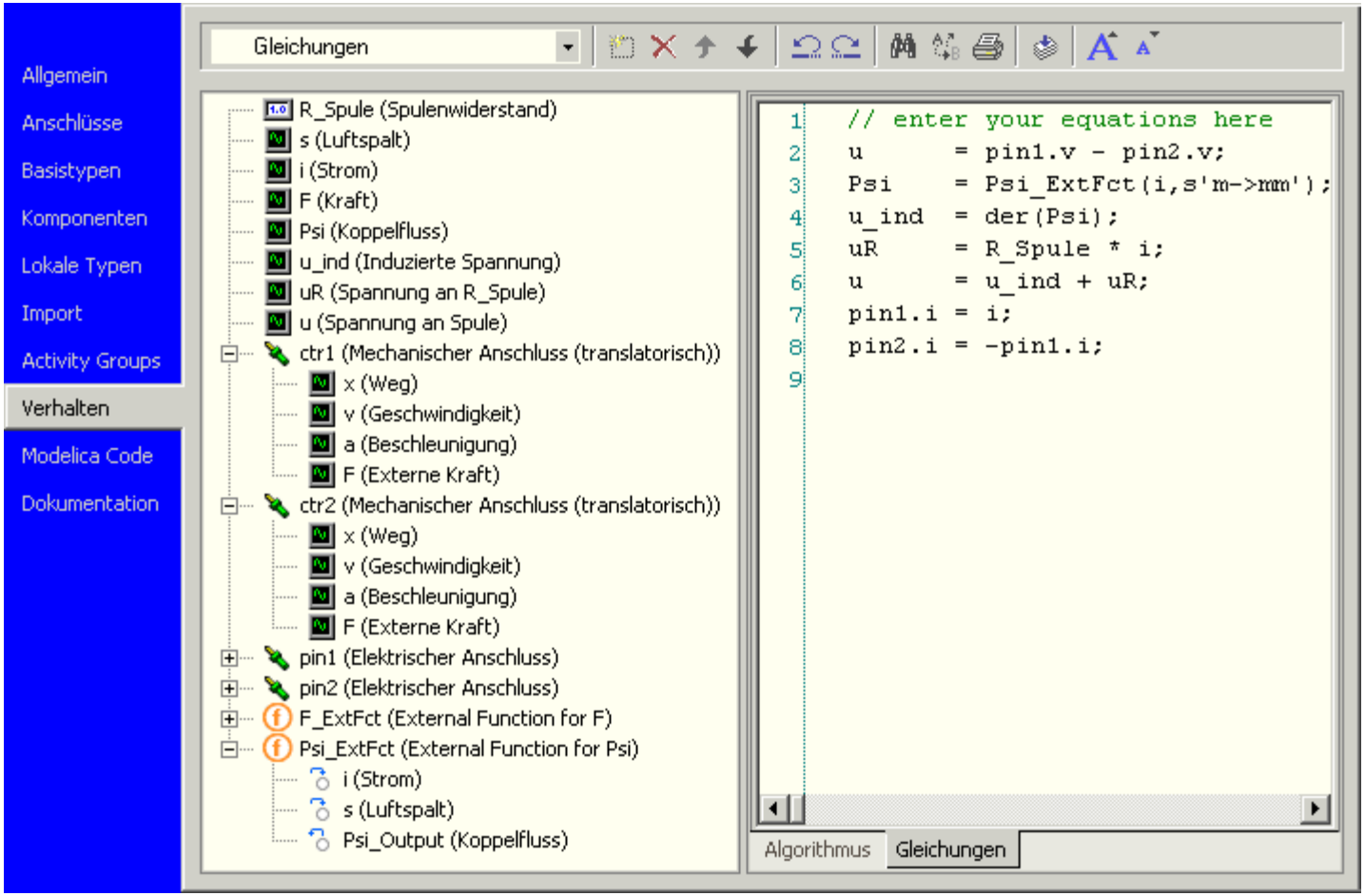
# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

## 3.1. Elektro-mechanischer Wandler (Mechanische Seite):



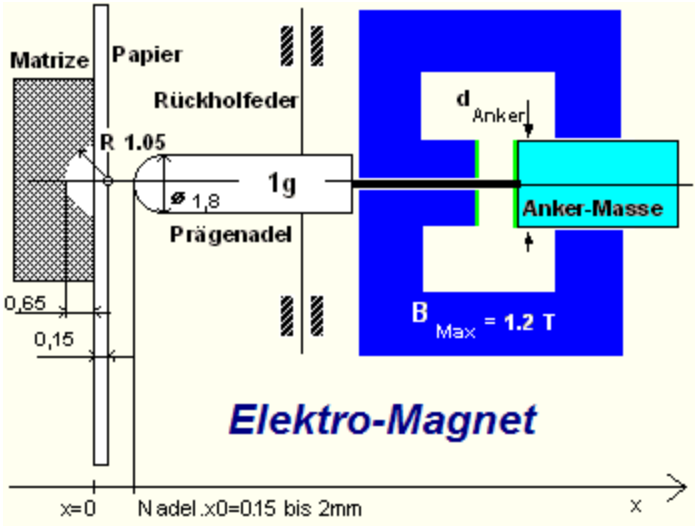
# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

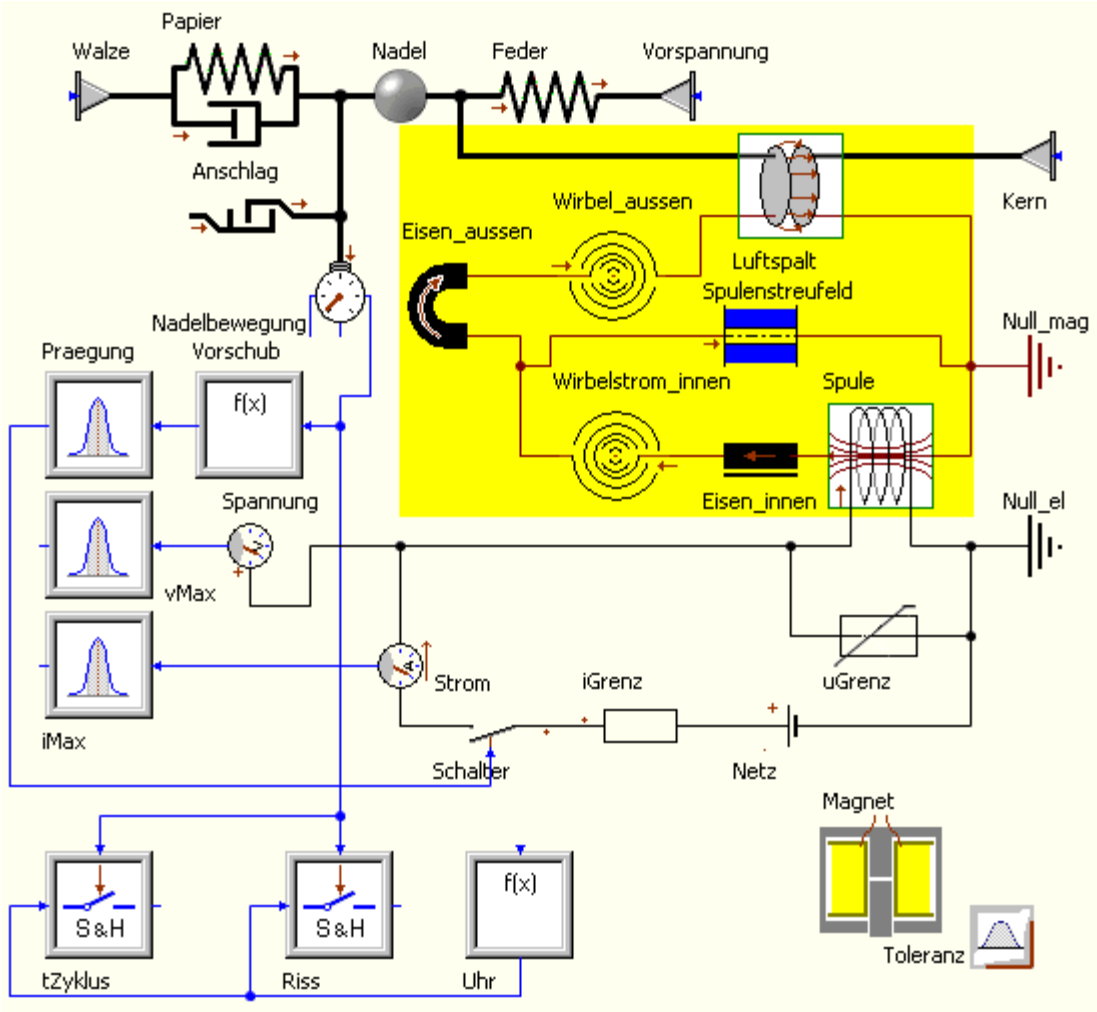
## 3.2. Elektro-mechanischer Wandler (Elektrische Seite):



# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

## 4.1. Netzwerk-Modell eines Magnet-Antriebes (Nadel-Antrieb „Blindenschrift-Präger“):

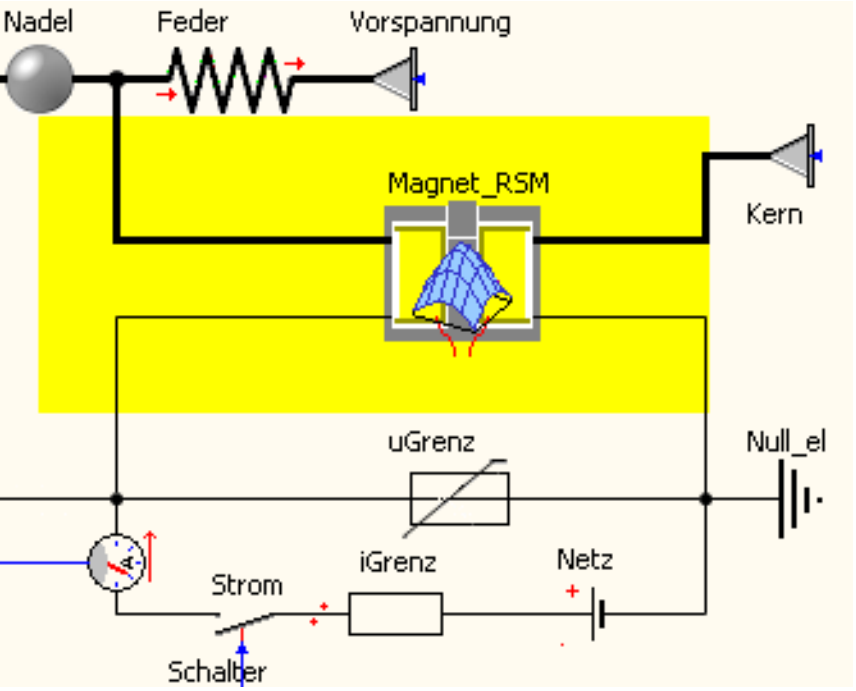


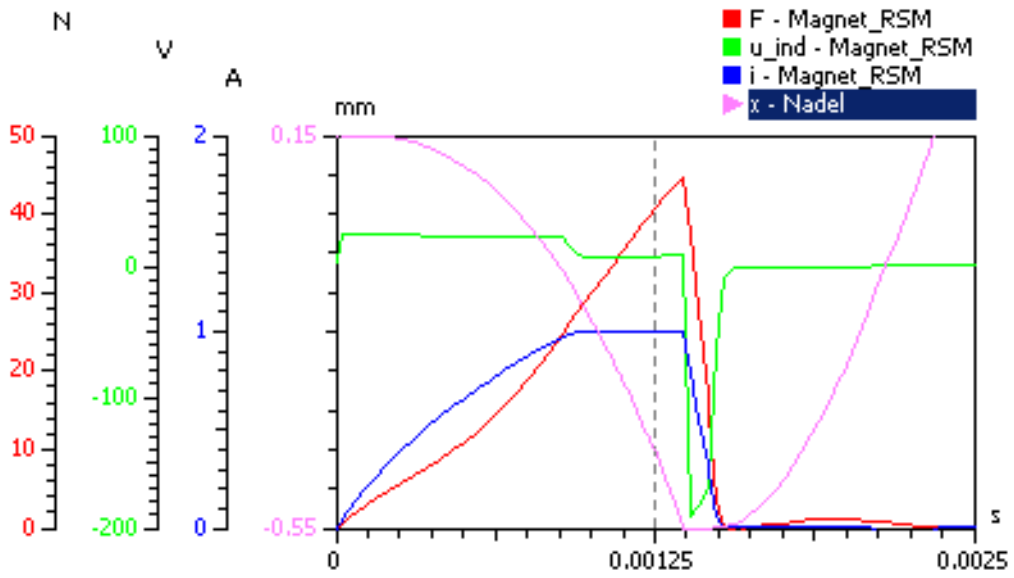



# Systemsimulation mit Kennfeldern F=f(i,s) & Psi=f(i,s)

**4.2. Netzwerk-Modell des Magnet-Antriebs mit „Kennfeld-Wandler“:**

- Kennfelder aus FE-Berechnung für einen konkreten Magnetkreis bilden auch für eine komplizierte Wandler-Geometrie das Übertragungsverhalten gut nach.
- Außer der Wandler-Geometrie kann man in der System-Simulation alles variieren.



# Zusammenfassung

- Die Modellierung und Simulation eines Produktes oder Verfahrens ist oft nicht durchführbar, weil:

  1. die Berechnungen zu zeitaufwändig sind,
  2. die zu modellierenden Zusammenhänge unbekannt sind,
  3. die Modell-Reduktion auf Netzwerk-Elemente sehr Zeit- und Kostenintensiv würde.

- Die Identifikation von Antwortflächen in OptiY ® ermöglicht eine schnelle und einfache Gewinnung eines mathematischen Ersatz-Modells aus Messdaten oder komplexer FEA. Der adaptive Gauß-Prozess reduziert dabei die Anzahl der Messpunkte  oder Modellberechnungen auf das unbedingt erforderliche Maß.

- Die als C-Code exportierten Funktionen können automatisiert in entsprechend aufgebaute Element-Typen von SimulationX-Modellen eingebunden werden (identifizierte Ersatzmodelle komplexer Komponenten als Bestandteil komplexer System-Modelle).

- Die Vorteile der Netzwerk-basierten System-Simulation lassen sich damit innerhalb von Entwurfsprozessen mit den Vorteilen der FEA bzw. der Messung an realen Objekten zusammenführen.