Generierung des Eingangssingals für Barrier Bucket RF Systeme and der GSI

Jonas Christ, Artem Moskalew, Maximilian Nolte
Jens Harzheim, M.Sc.

Projektseminar Beschleunigertechnik

TECHNISCHE UNIVERSITÄT DARMSTADT

T E M F
Institut für Theorie Elektromagnetischer Felder
Computational Electromagnetics Research Group at GSCE

Outline

1. Einführung
   - Problemstellung
   - Aufbau
2. Design
   - Blöcke
   - Test Driven Development
   - MockSystem
3. Optimierung
   - Optimierung der Übertragungsfunktion
   - Optimierung der Kennlinie

Problemstellung

- Barrier-Bucket System

Problemstellung

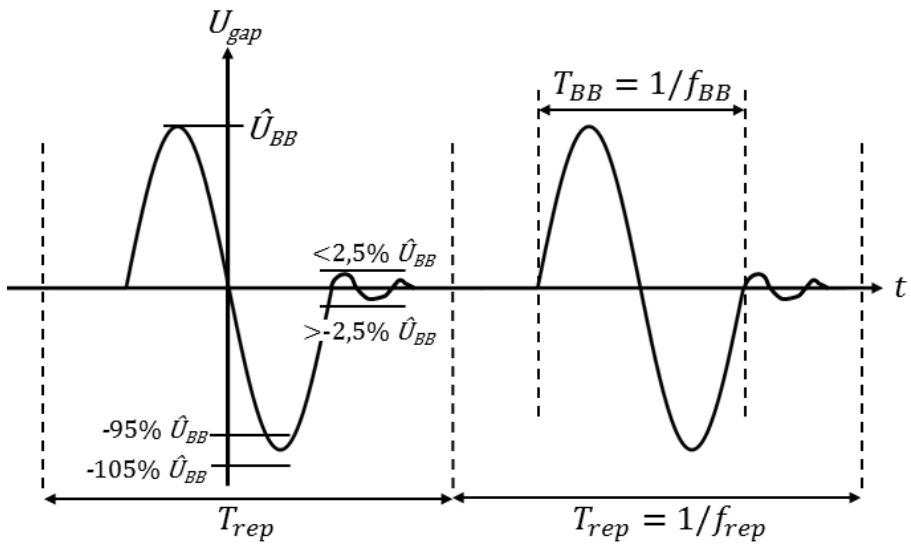
- Barrier-Bucket System :
  - Longitudinale Manipulation des Teilchenstrahls

Problemstellung

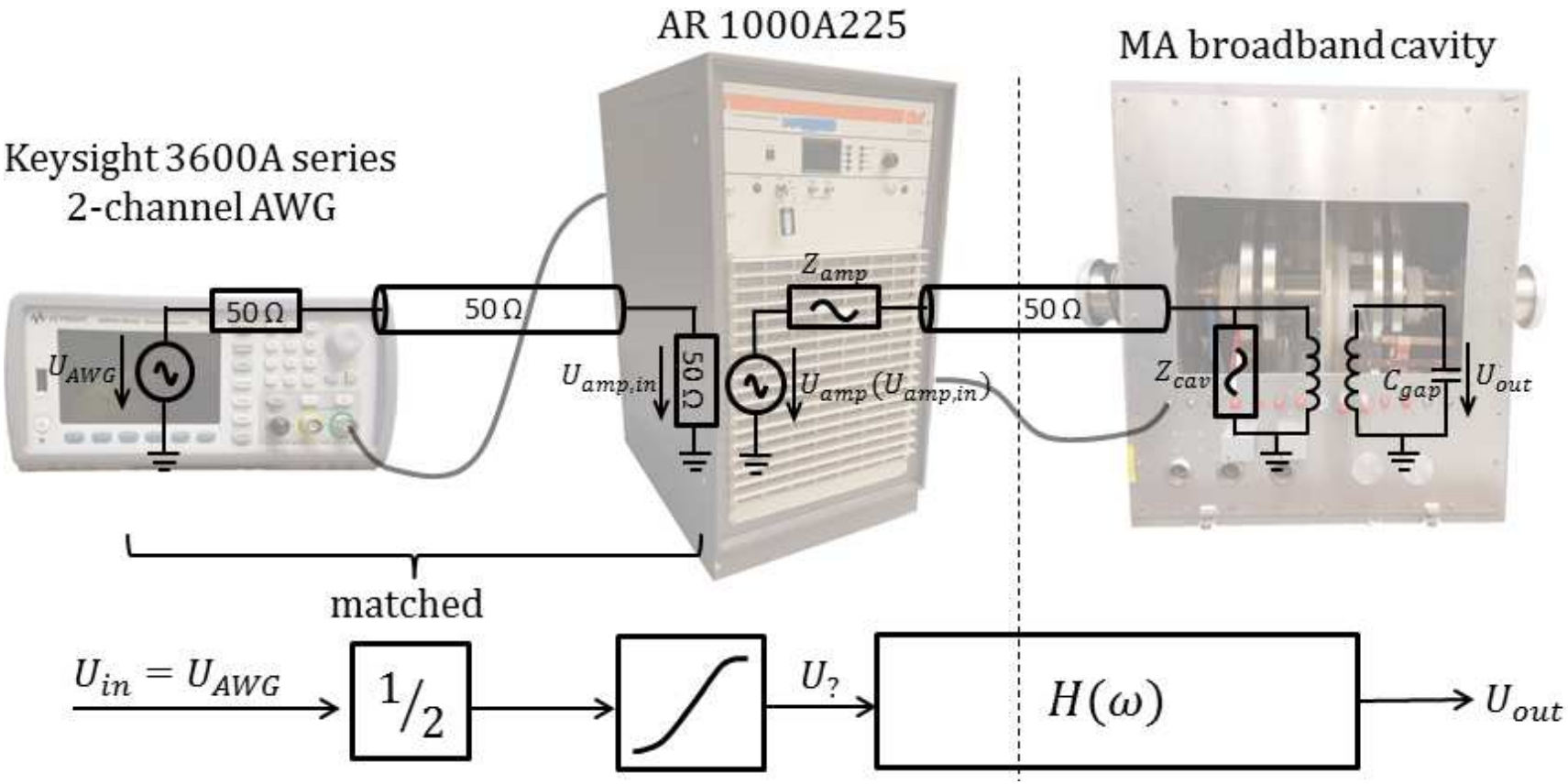
- Barrier-Bucket System :
  - Longitudinale Manipulation des Teilchenstrahls
- Ziel

Problemstellung

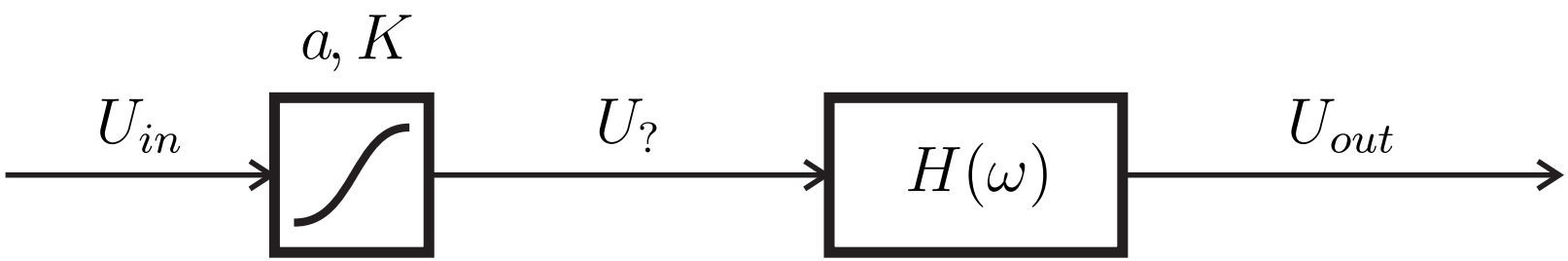
- Barrier-Bucket System :
  - Longitudinale Manipulation des Teilchenstrahls
- Ziel :
  - Gap Spannung in Form einer Ein-Sinus Periode
  - Qualität das Signals



Aufbau und Modell

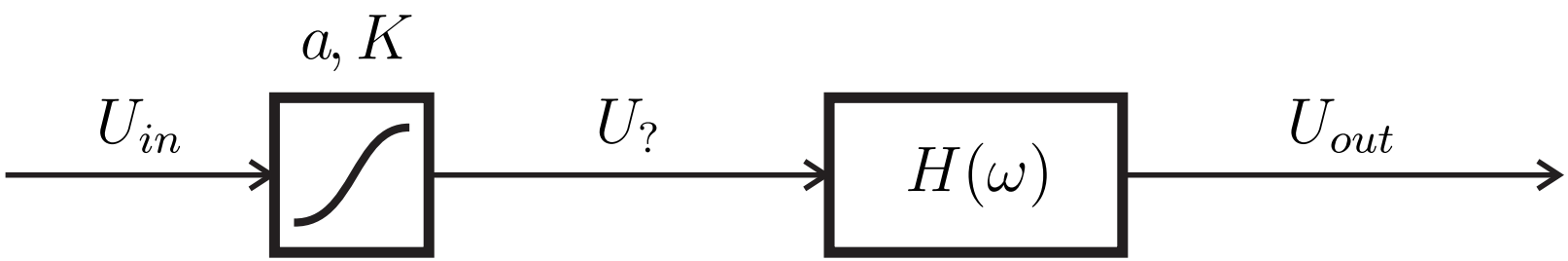


Aufbau und Modell



Aufbau und Modell

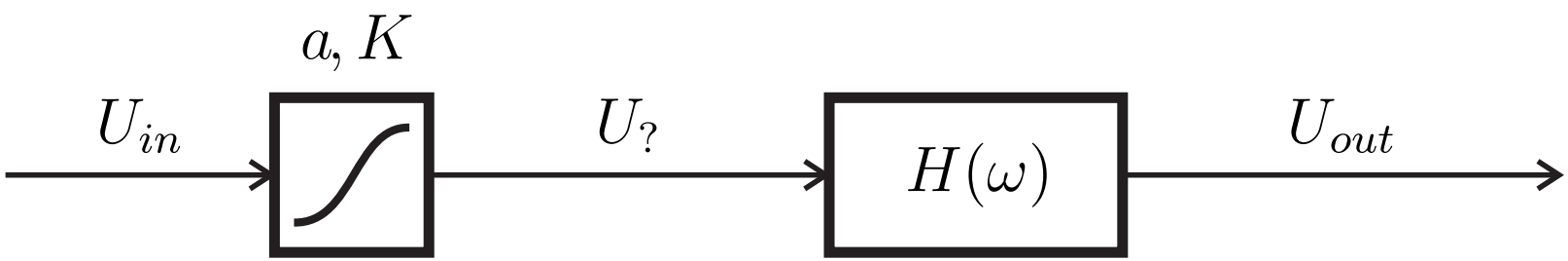
- Gegeben:
  - Lineare Übertragungsfunktion H bestimmt durch Pseudorauschen
  - System linear bis $\hat{U}_{BB} \approx 550\,V$ genähert



Aufbau und Modell

- Gegeben:
  - Lineare Übertragungsfunktion H bestimmt durch Pseudorauschen
  - System linear bis $\hat{U}_{BB} \approx 550\,V$ genähert
- Hammerstein Modell :
  - Ergänzung um eine nichtlineare Vorverzerrung mit einem Potenzreihenansatz

$$U_?(t) = \sum_{n=1}^{N} a_n \, [U_{in}(t)]^n \qquad \underline{U}_{out}(\omega) = H(\omega) \cdot \underline{U}_?(\omega)$$

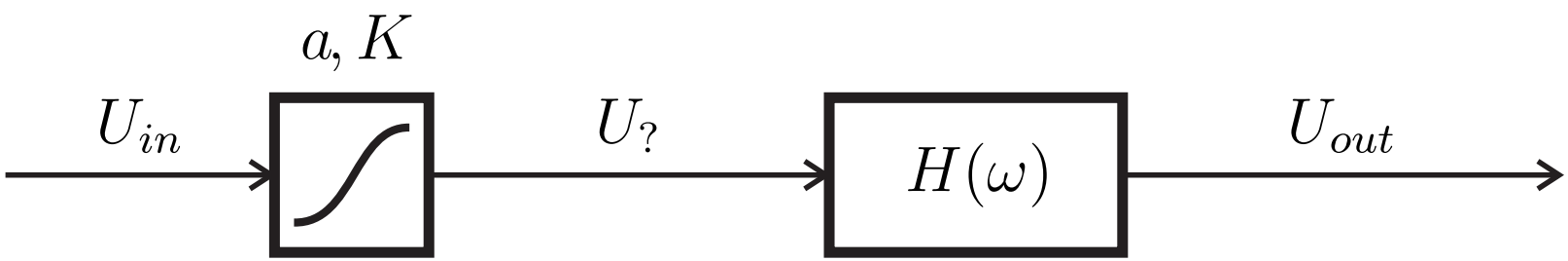


Aufbau und Modell

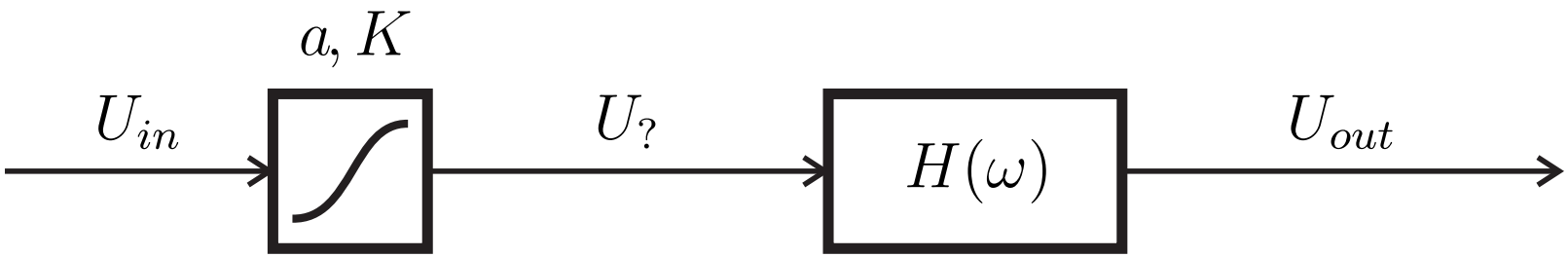
- Gegeben:
  - Lineare Übertragungsfunktion H bestimmt durch Pseudorauschen
  - System linear bis $\hat{U}_{BB} \approx 550\,V$ genähert
- Hammerstein Modell :
  - Ergänzung um eine nichtlineare Vorverzerrung mit einem Potenzreihenansatz

$$U_?(t) = \sum_{n=1}^{N} a_n \, [U_{in}(t)]^n \qquad \underline{U}_{out}(\omega) = H(\omega) \cdot \underline{U}_?(\omega)$$

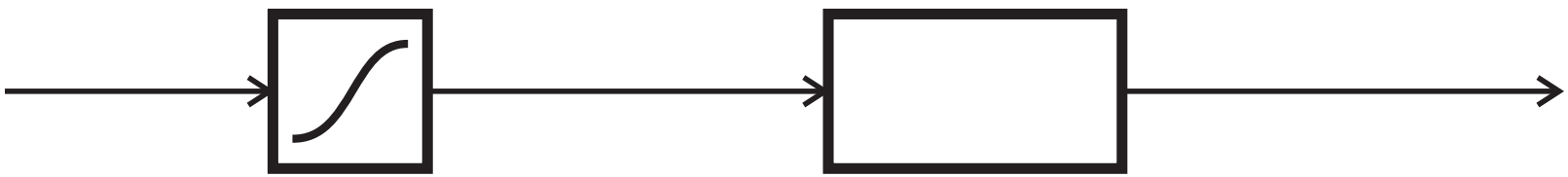
- Zielsetzung :
  - Parameter a_n der Kennlinie K zubestimmen
  - Ersten Optimierungs Ansatz implementieren



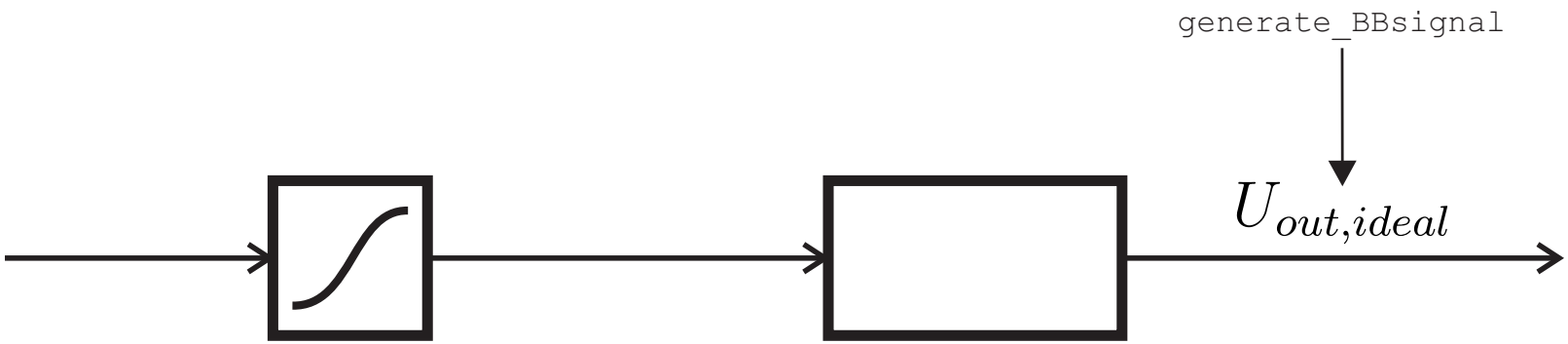
Code: Die Blöcke



Code: Die Blöcke

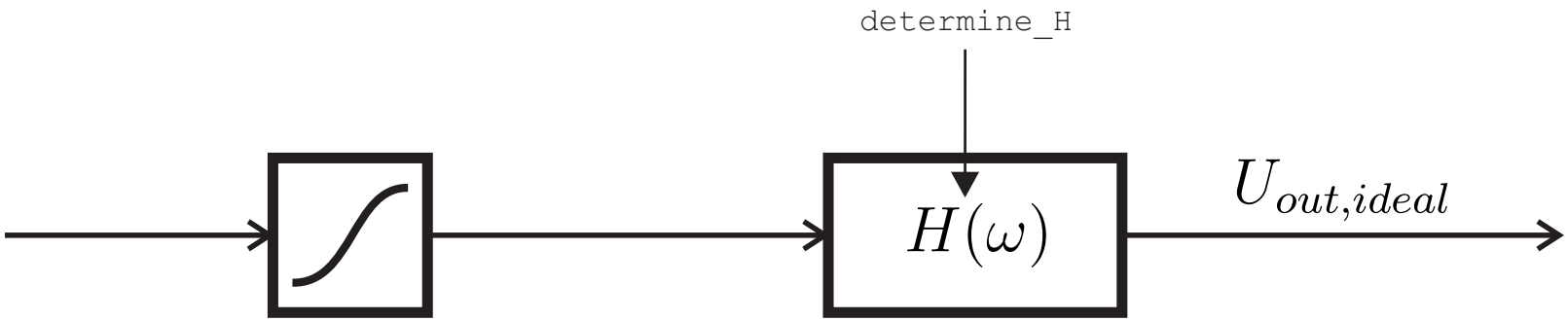


Code: Die Blöcke



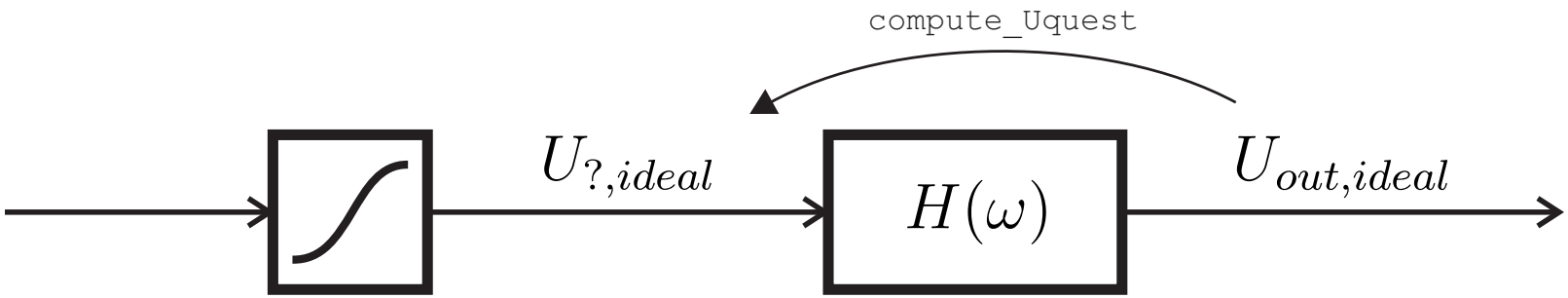

```
1 Uout_ideal = generate_BBsignal ( fq_rep , fq_bb , vpp )
```

Code: Die Blöcke



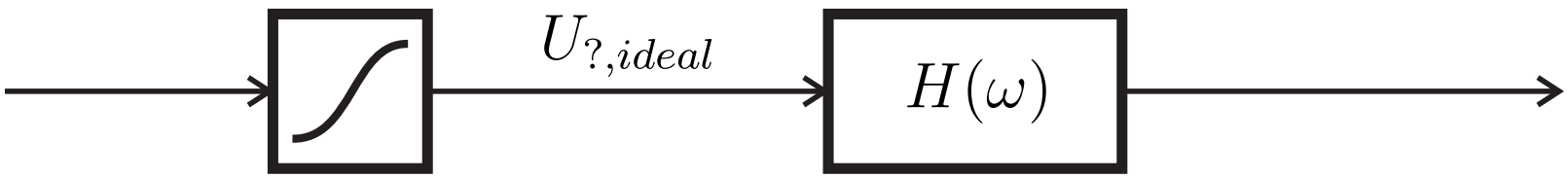

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = determine_H ( )
```

Code: Die Blöcke



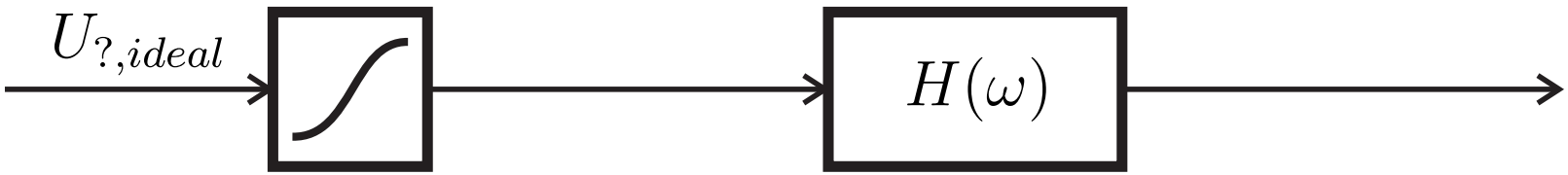

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = determine_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```

Code: Die Blöcke



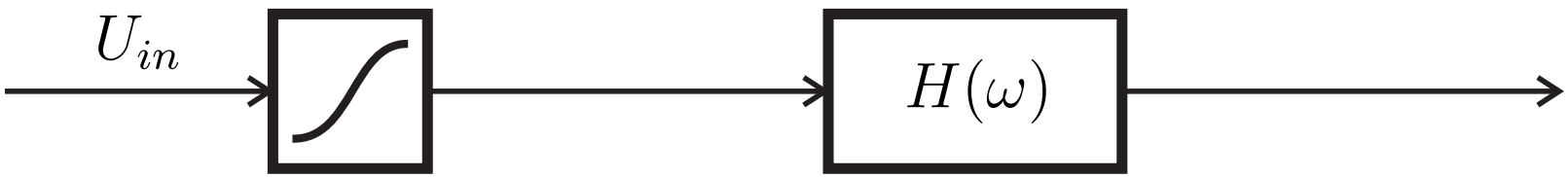

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = determine_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```

Code: Die Blöcke



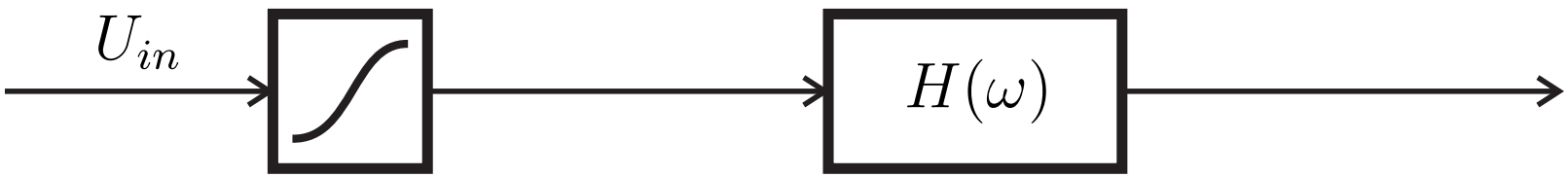

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = determine_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```

Code: Die Blöcke



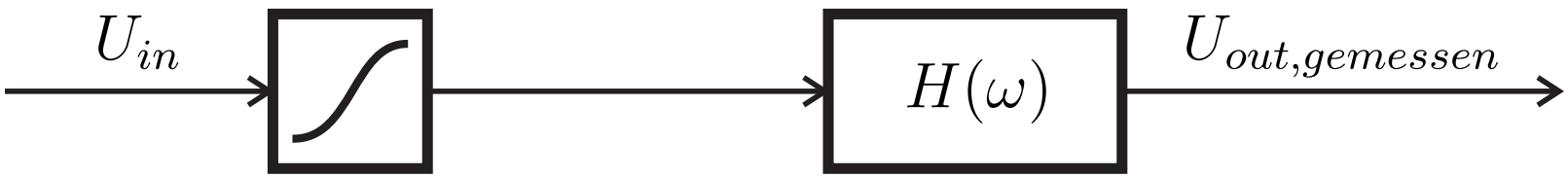

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
```

Code: Die Blöcke



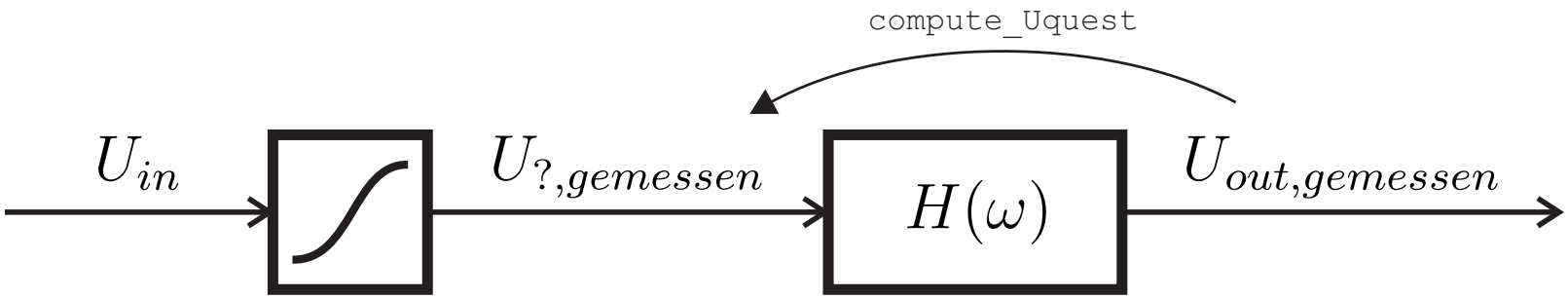

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
```

Code: Die Blöcke



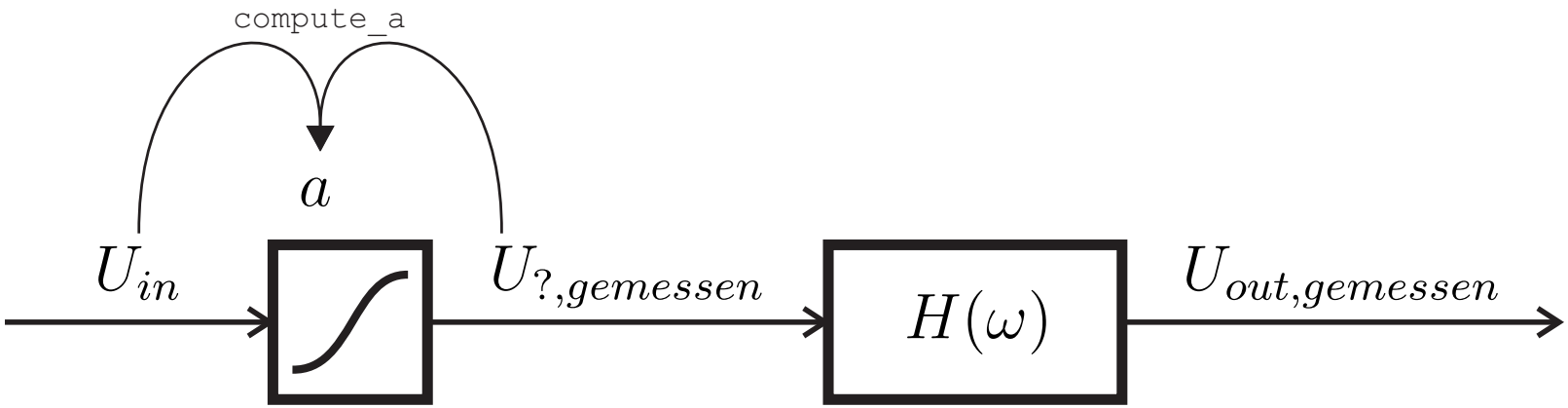

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
```

Code: Die Blöcke



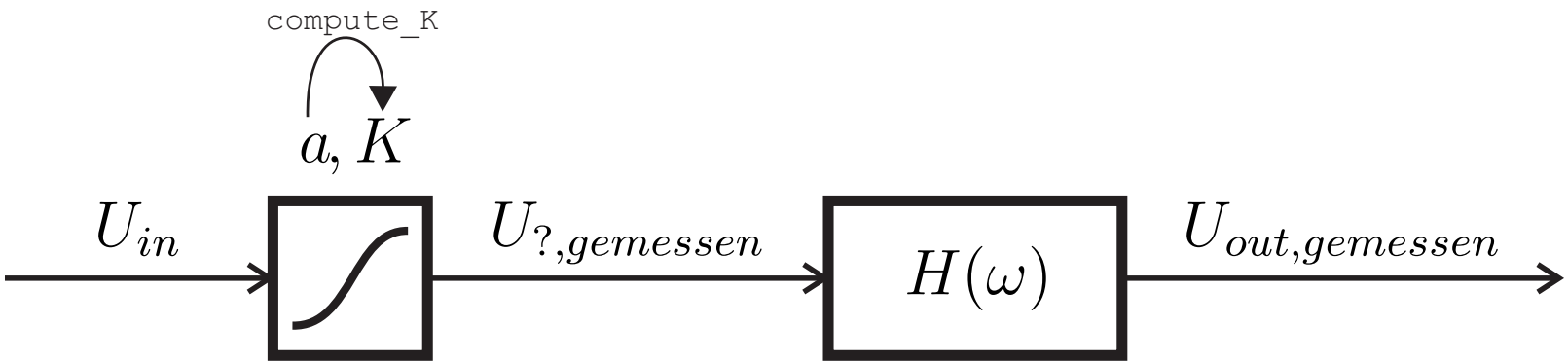

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = determine_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
```

Code: Die Blöcke



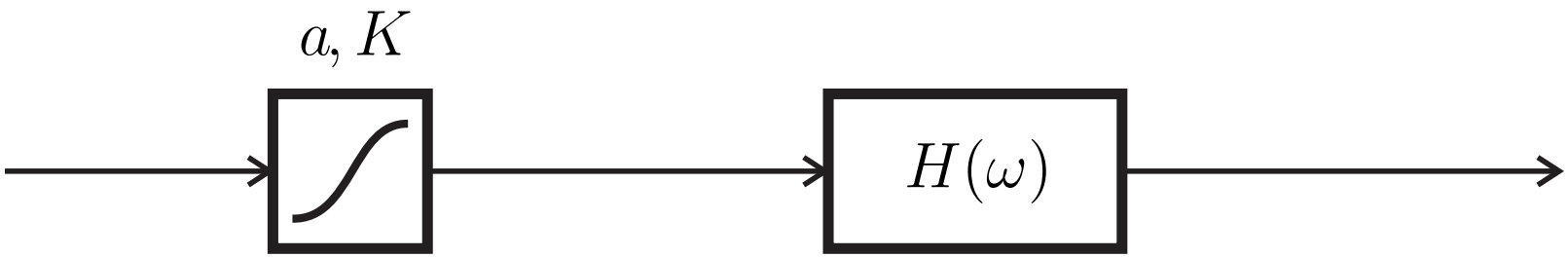

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = determine_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
7 a = compute_a ( Uin, Uquest_measured, N )
```

Code: Die Blöcke



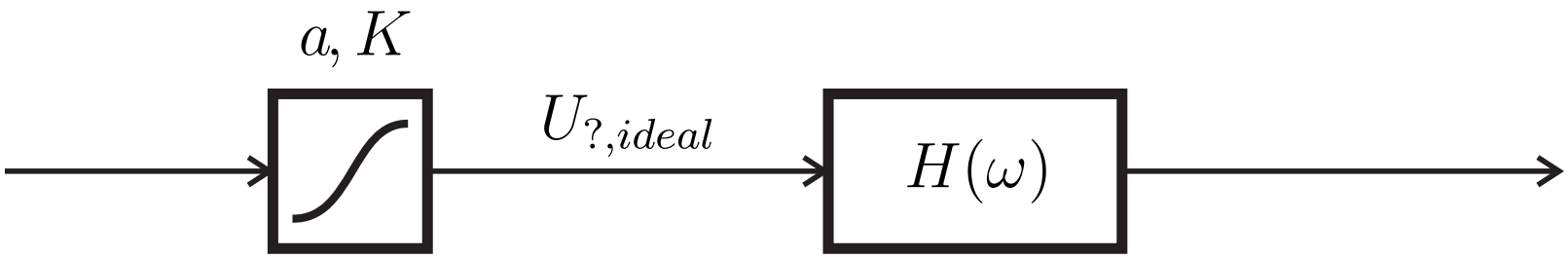

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Die Blöcke



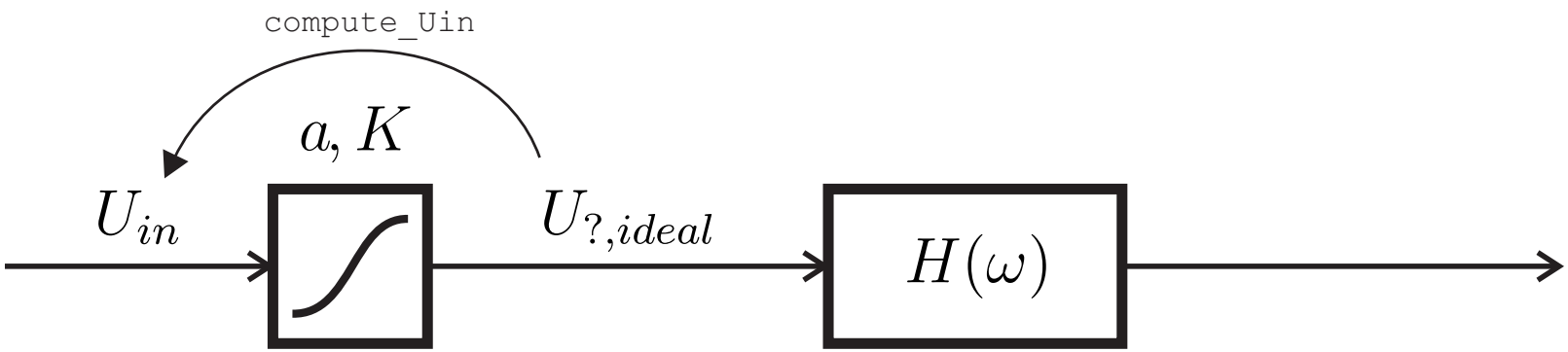

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Die Blöcke



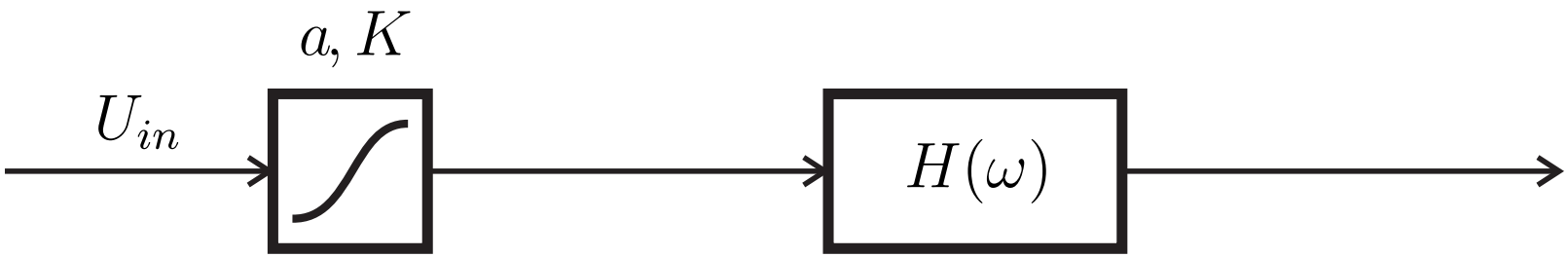

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Die Blöcke



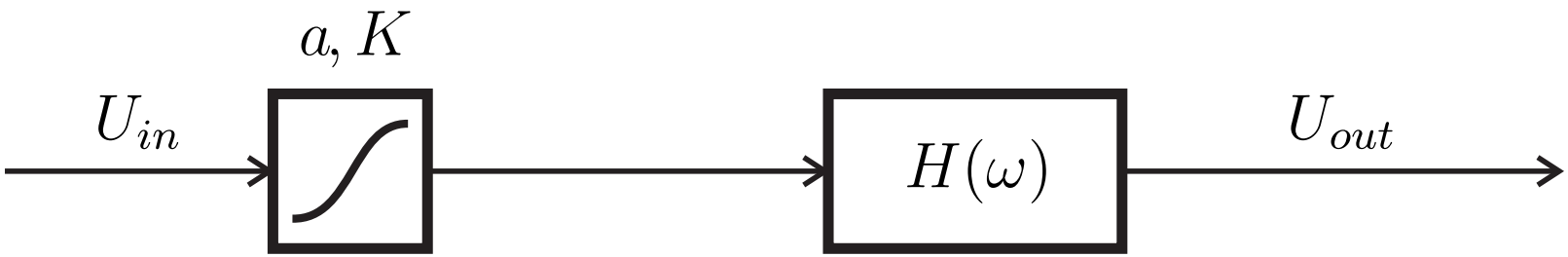

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
Uin = compute_Uin ( Uquest_ideal, K )
```

Code: Die Blöcke




```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = determine_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
Uin = compute_Uin ( Uquest_ideal, K )
```

Code: Die Blöcke




```
1  Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2  H = determine_H ( )
3  Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4  Uin = Uquest_ideal
5  Uout_measured = measure_Uout ( Uin )
6  Uquest_measured = compute_Uquest ( Uout_measured, H )
7  a = compute_a ( Uin, Uquest_measured, N )
8  K = compute_K ( a )
9  Uin = compute_Uin ( Uquest_ideal, K )
10 Uout = measure_Uout ( Uin )
```

Test Driven Development

Test Driven Development

- 27 Unit Tests

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

- Ermöglichen:
  - inkrementierende Code-Anpassungen

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

- Ermöglichen:
  - inkrementierende Code-Anpassungen
  - verteiltes Debuggen ohne den Messaufbau

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

- Ermöglichen:
  - inkrementierende Code-Anpassungen
  - verteiltes Debuggen ohne den Messaufbau
- Zwingen zum modularen Code-Design

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

- Ermöglichen:
  - inkrementierende Code-Anpassungen
  - verteiltes Debuggen ohne den Messaufbau
- Zwingen zum modularen Code-Design
- Erleichtern das Migrieren der Funktionen aus anderen Sprachen

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

- Ermöglichen:
  - inkrementierende Code-Anpassungen
  - verteiltes Debuggen ohne den Messaufbau
- Zwingen zum modularen Code-Design
- Erleichtern das Migrieren der Funktionen aus anderen Sprachen

Nachteile:

Test Driven Development

- 27 Unit Tests
- 4 System Tests

Vorteile:

- Ermöglichen:
  - inkrementierende Code-Anpassungen
  - verteiltes Debuggen ohne den Messaufbau
- Zwingen zum modularen Code-Design
- Erleichtern das Migrieren der Funktionen aus anderen Sprachen

Nachteile:

- Extra Aufwand: mehr Code zu debuggen

Das Mock-System

Das Mock-System

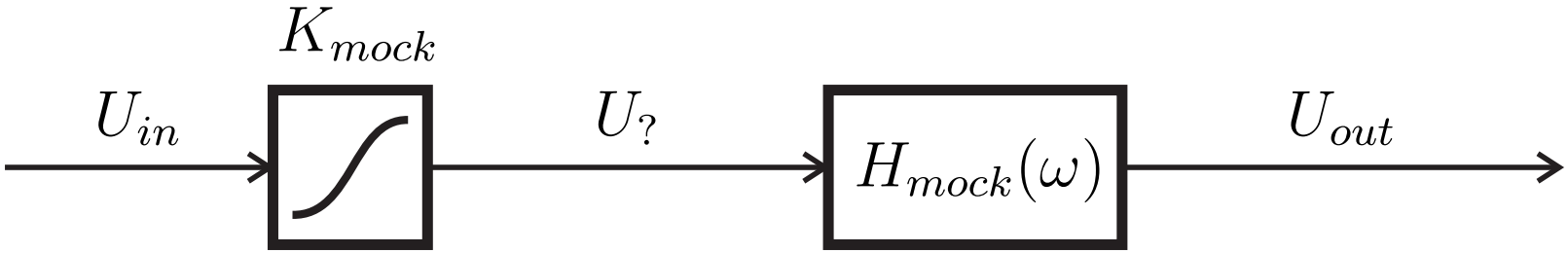
- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`

Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

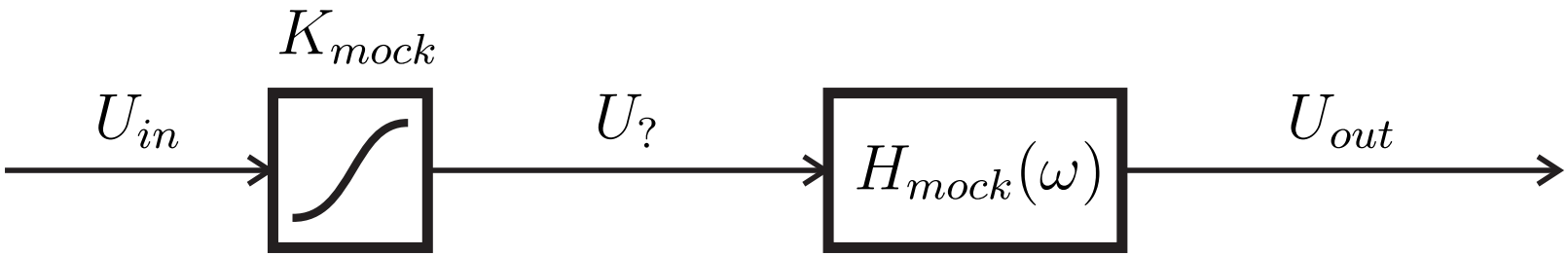
Das Mock-System

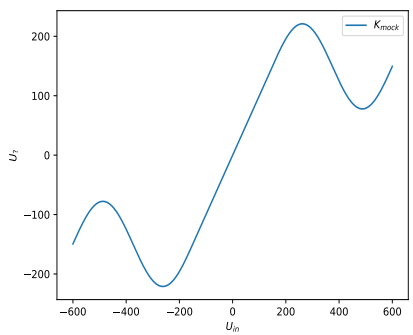
- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model



Das Mock-System

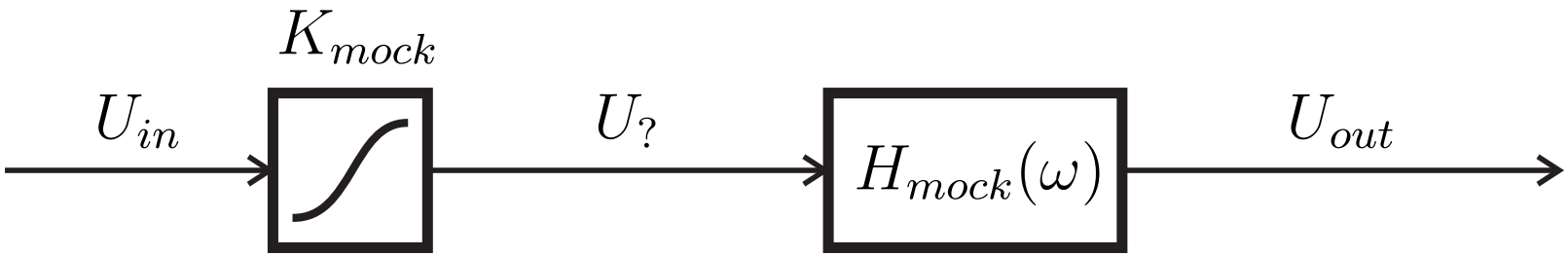
- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

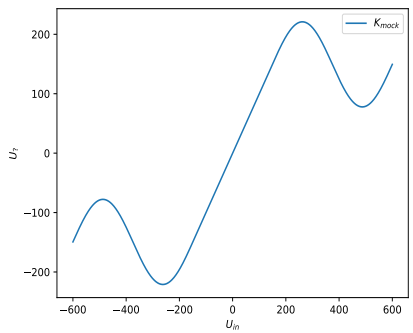


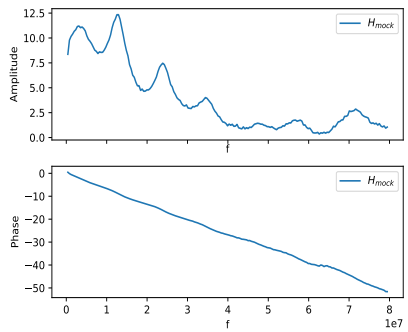


Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model







Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

Vorteile:

Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

Vorteile:

- Ermöglicht:
  - Unit Tests von Bausteinen, in den Gerätekommunikation stattfindet

Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

Vorteile:

- Ermöglicht:
  - Unit Tests von Bausteinen, in den Gerätekommunikation stattfindet
  - System Tests

Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

Vorteile:

- Ermöglicht:
  - Unit Tests von Bausteinen, in den Gerätekommunikation stattfindet
  - System Tests
  - Testen von Spezialszenarien

Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

Vorteile:

- Ermöglicht:
  - Unit Tests von Bausteinen, in den Gerätekommunikation stattfindet
  - System Tests
  - Testen von Spezialszenarien
- Hilft das System besser zu verstehen

Das Mock-System

- Wird genutzt, wenn mit Geräten kommuniziert wird:
  - `mock_system.write_to_AWG`
  - `mock_system.read_from_DSO`
- Simuliert das Verhalten des Messaufbaus nach dem Hammerstein Model

Vorteile:

- Ermöglicht:
  - Unit Tests von Bausteinen, in den Gerätekommunikation stattfindet
  - System Tests
  - Testen von Spezialszenarien
- Hilft das System besser zu verstehen

Nachteile:

- Extra Aufwand: mehr Code zu debuggen

Optimierung von K

- Bestimmung von K mit linear vorverzerrten Signal
- Anpassung von K für nichtliniear vorverzerrte Signale

$$U_{?,\mathrm{meas}}(t) = \sum_{n=1}^{N} \overline{a}_n \left[U_{in}(t)\right]^n \qquad U_{?,\mathrm{ideal}}(t) = \sum_{n=1}^{N} a_n \left[U_{in}(t)\right]^n \tag{1}$$

- Oder direkt über die Differenz der Signale

$$\Delta U_?(t) = U_{?,\mathrm{meas}}(t) - U_{?,\mathrm{ideal}}(t) = \sum_{n=1}^{N} \left(\overline{a}_n - a_n\right) \left[U_{in}(t)\right]^n = \sum_{n=1}^{N} \tilde{a}_n \left[U_{in}(t)\right]^n \tag{2}$$

Optimierung von K

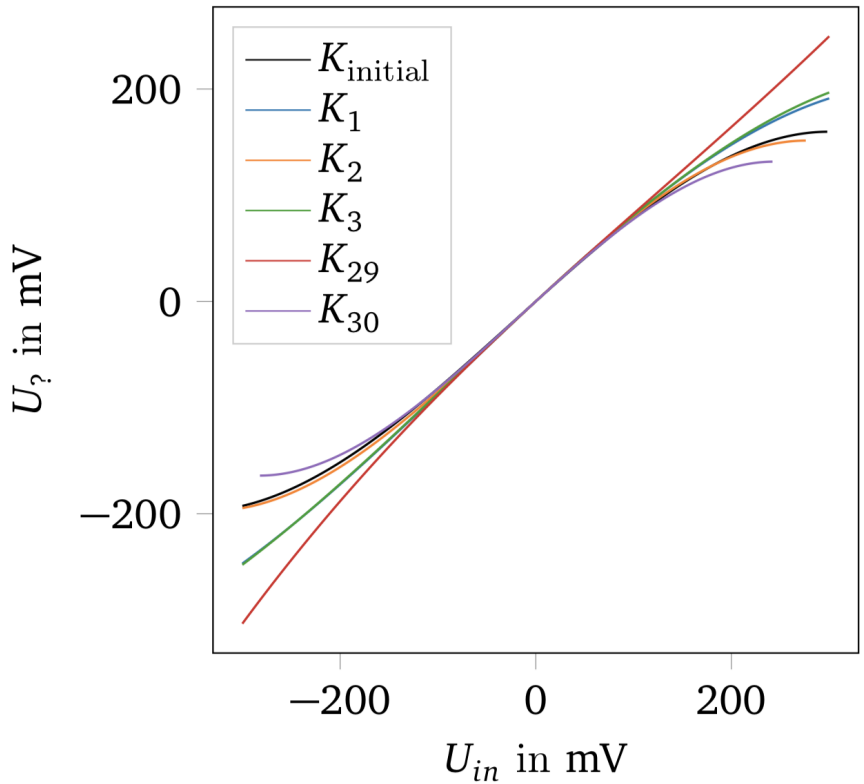
- Bestimmung der Parameter $\tilde{a}_n$
- Vergleichen der Samples $\Delta U_{?,i} = \Delta U_?(i \cdot \Delta t)$ mit $U_{in,i} = U_{in}(i \cdot \Delta t)$
- Lösung des linearen Optimierungsproblems ergibt die Anpassung der alten Parameter

$$a_n^{i+1} = a_n^i + \sigma_a^i \tilde{a}_n^i \tag{3}$$

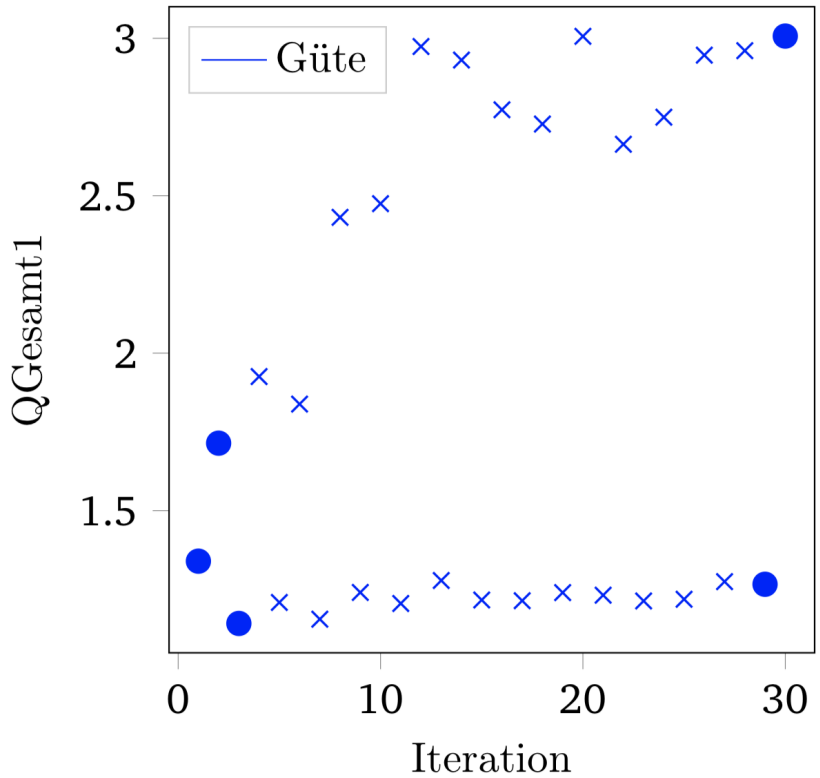
Erster Ansatz

- K im gleichen Spannungsbereich anpassen
- Referenz zum Rechnen $U_{out,ideal}$ mit $V_{PP} = 6\,\mathrm{V}$
- Eingangsspannung mit $V_{PP} = 587\,\mathrm{mV}$

Erster Ansatz

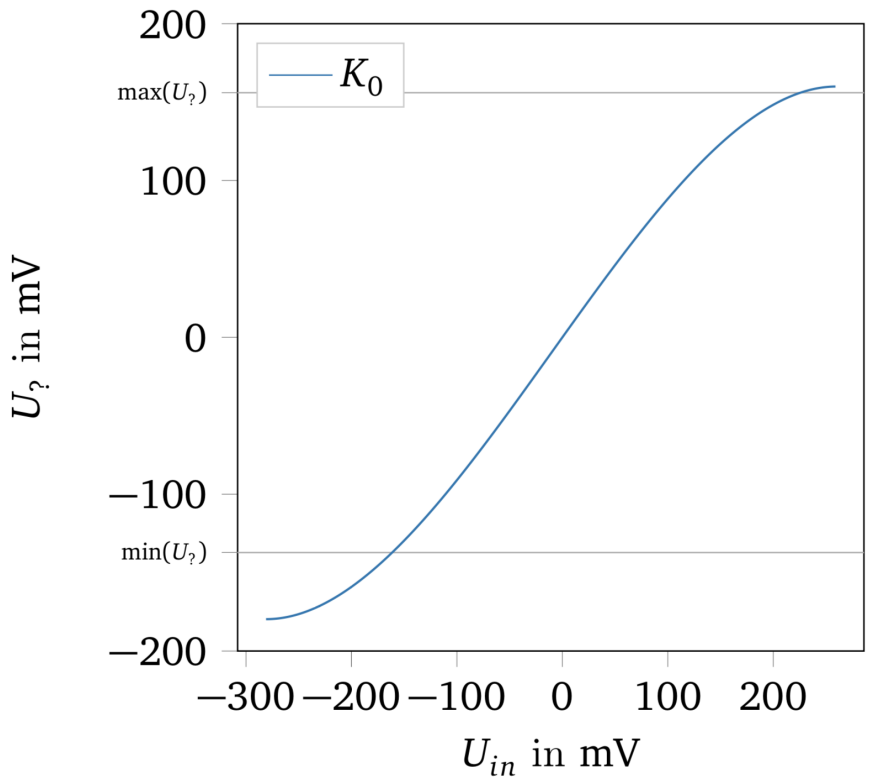


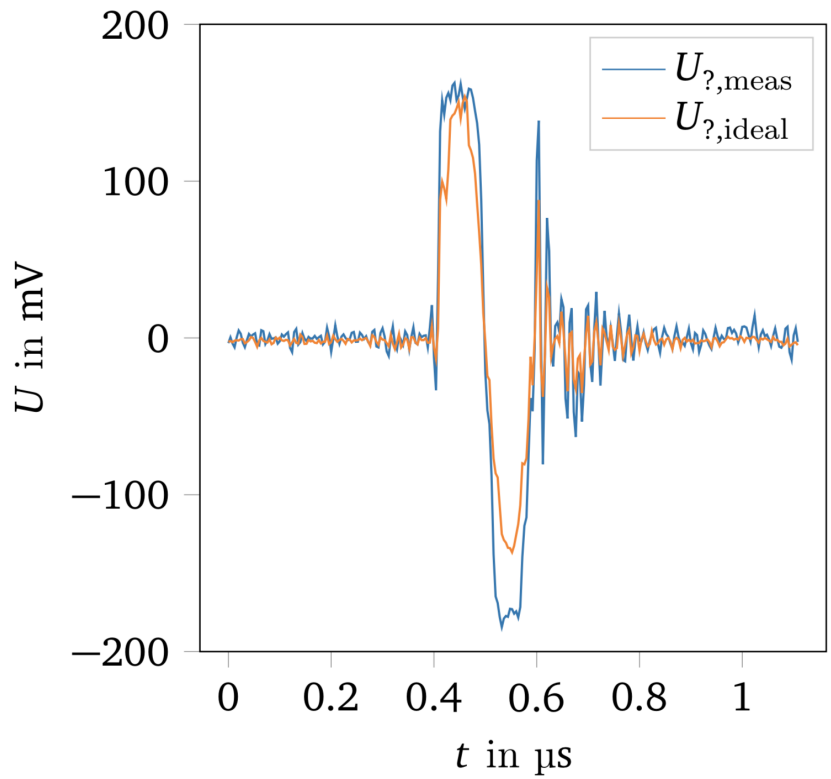
(a) Kennlinien



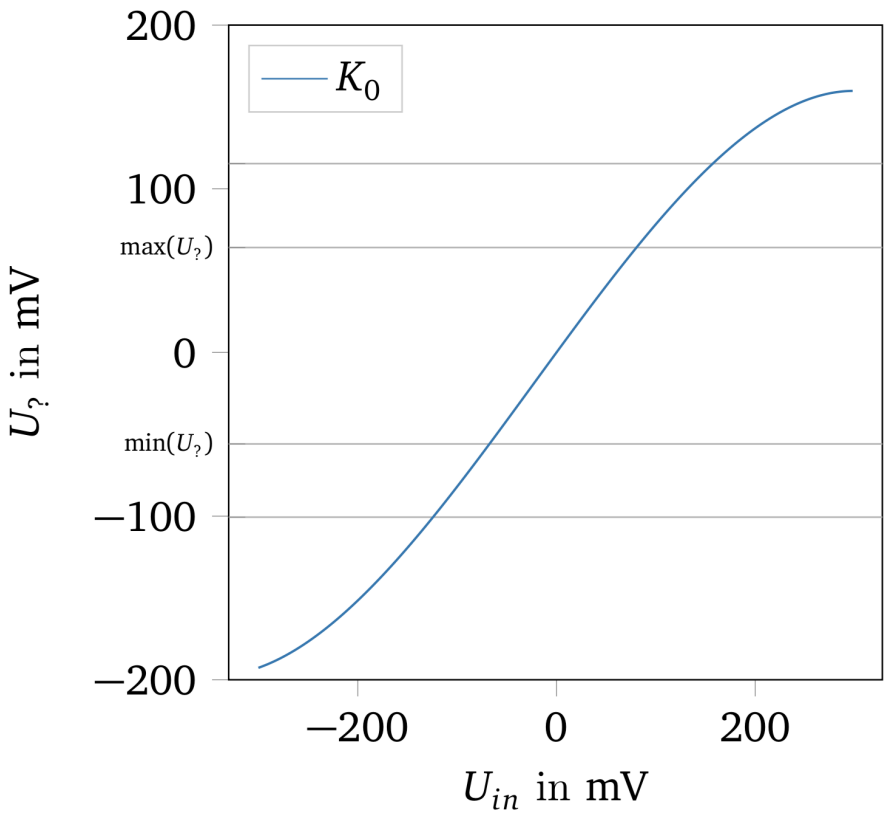
(b) Qualität

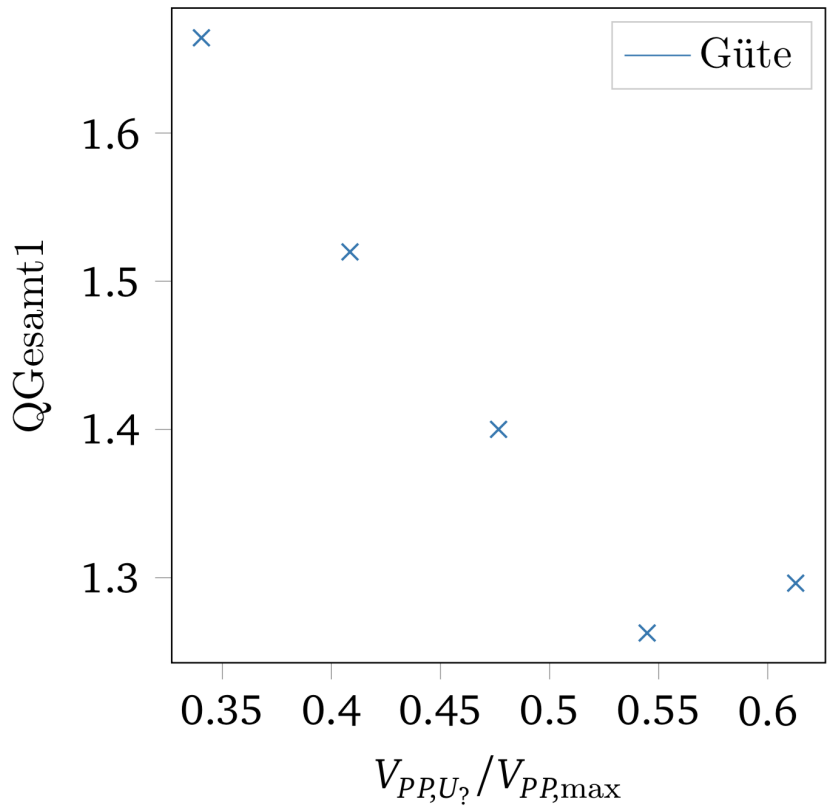
Grenzen der Kennlinie





Grenzen der Kennlinie





Zweiter Ansatz

- K in einem kleineren Spannungsbereich anpassen
- Referenz zum Rechnen $U_{out,ideal}$ mit $V_{PP} = 3\,\mathrm{V}$
- Eingangsspannung mit $V_{PP} = 290\,\mathrm{mV}$
- Ausgangsspannung gemessen über Gapspannungsteiler $V_{PP} = 2.7\,\mathrm{V}$

Zweiter Ansatz