Generierung des Eingangssingals für Barrier Bucket RF Systeme and der GSI

TECHNISCHE UNIVERSITÄT DARMSTADT

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

Projektseminar Beschleunigertechnik

Outline

Problemstellung

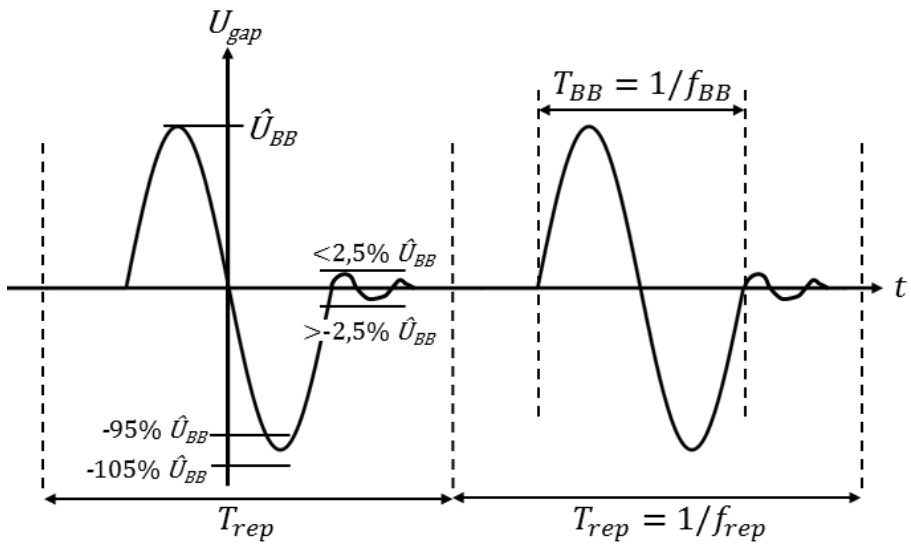
- Barrier-Bucket System
- Ziel

Problemstellung

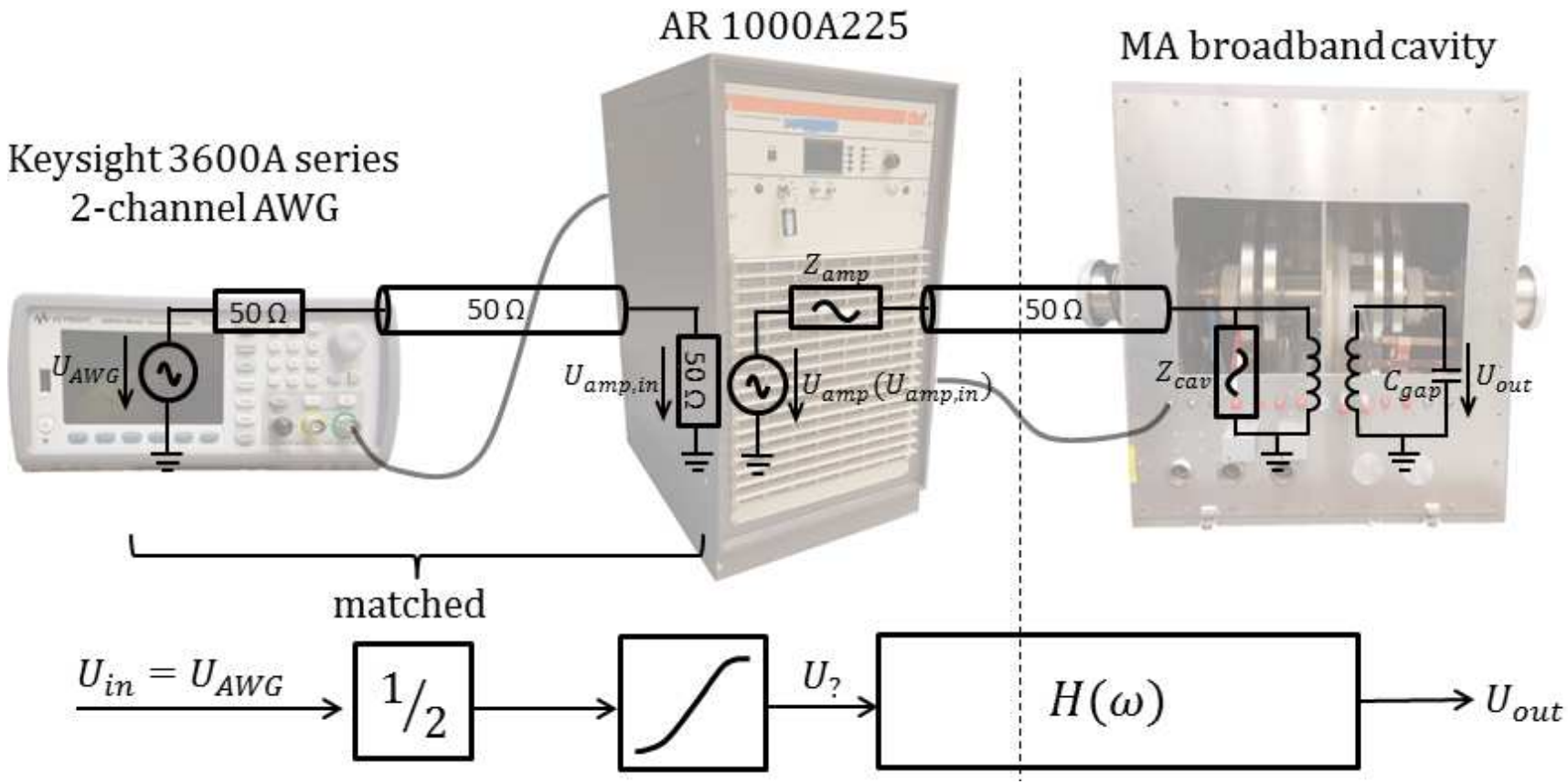
- Barrier-Bucket System :
  - Longitudinale Manipulation der Bunches
- Ziel

Problemstellung

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



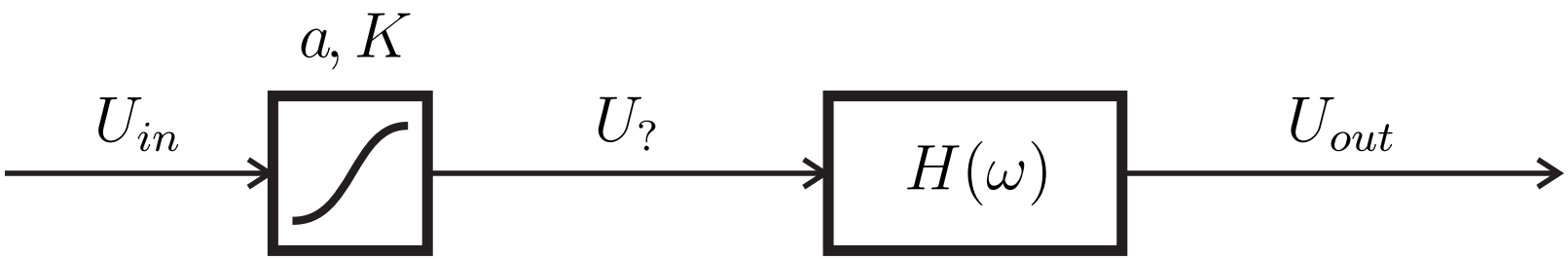
Aufbau und Modell



Aufbau und Modell

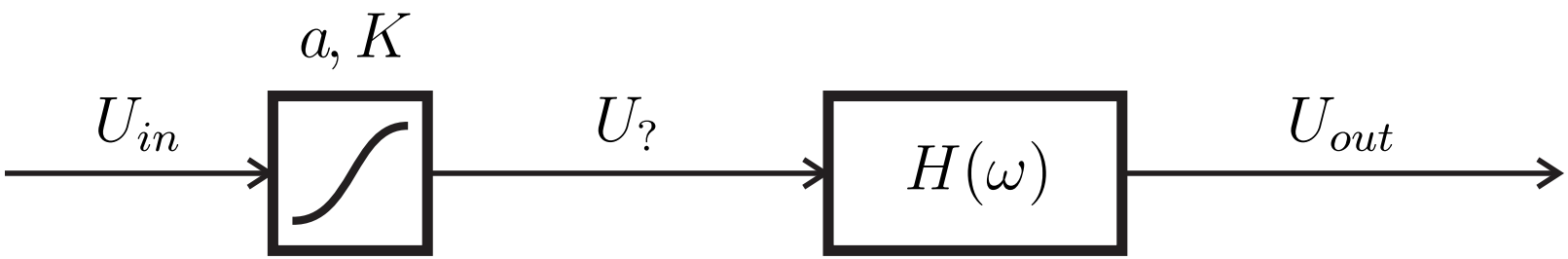
- Hammerstein Modell

- Zielsetzung



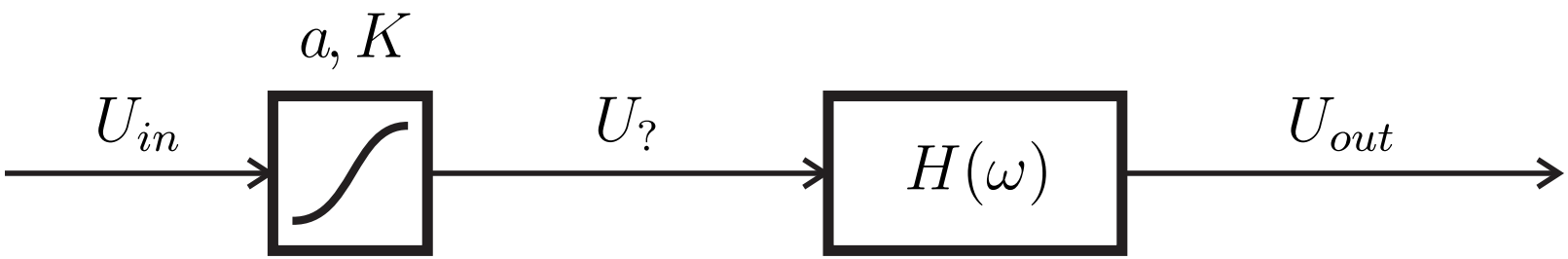
Aufbau und Modell

- Hammerstein Modell :
  - System ist linear bis $\hat{U}_{BB} \approx 550\,V$
  - Ergänzung um eine nichtlineare Vorverzerrung
  - Potenzreihenansatz $U_?(t) = \sum_{n=1}^{N} a_n \left[U_{in}(t)\right]^n$
- Zielsetzung



Aufbau und Modell

- Hammerstein Modell :
  - System ist linear bis $\hat{U}_{BB} \approx 550\,V$
  - Ergänzung um eine nichtlineare Vorverzerrung
  - Potenzreihenansatz $U_?(t) = \sum_{n=1}^{N} a_n \left[U_{in}(t)\right]^n$
- Zielsetzung :
  - Parameter a_n der Kennlinie zubestimmen



Dokumentation und Gerätekommunikation

- Dokumentation

- Gerätekommunikation

Dokumentation und Gerätekommunikation

- Dokumentation :
  - Handhabung der Geräte, Vorgehensweise bei Tests
  - Bedienung des Programms
  - Ausführliches Kommentieren der Code-Funktionalität
- Gerätekommunikation

Dokumentation und Gerätekommunikation

- Dokumentation :
  - Handhabung der Geräte, Vorgehensweise bei Tests
  - Bedienung des Programms
  - Ausführliches Kommentieren der Code-Funktionalität
- Gerätekommunikation :
  - Treiber und Programmer-Manuals zur Nutzung des Programms von anderen Geräten aus
  - Laufzeitoptimierung durch Abfrage von Gerätezuständen mittels VISA
  - Verbesserung der Auflösung des Signals durch Anpassung der Darstellung des Oszilloskops mittels VISA

Dokumentation und Gerätekommunikation - Evaluierung

Dokumentation und Gerätekommunikation - Evaluierung

- Unvollständige Dokumentation:
  - Parameter und Return-Variablen dokumentieren

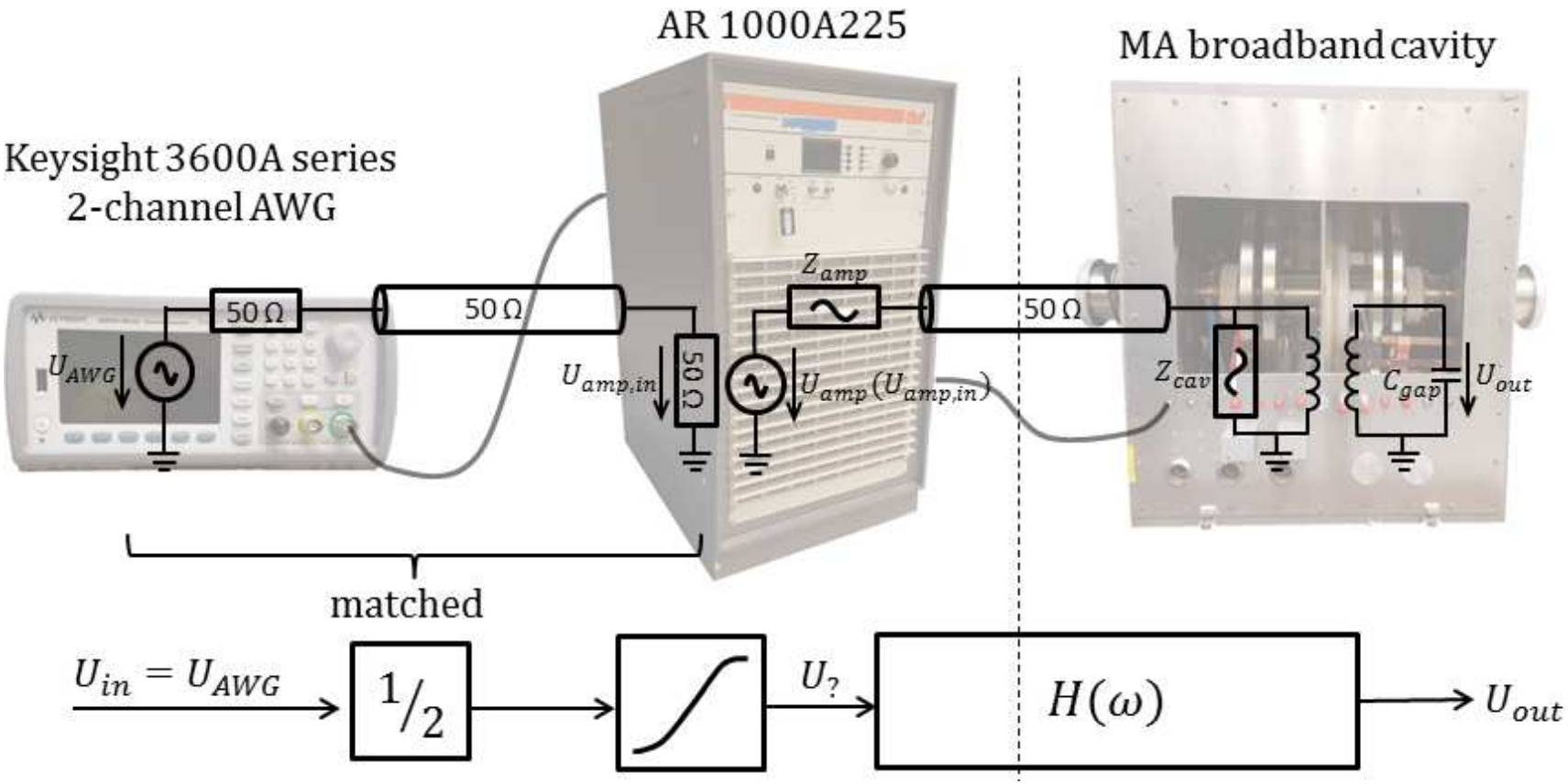
Dokumentation und Gerätekommunikation - Evaluierung

- Unvollständige Dokumentation:
  - Parameter und Return-Variablen dokumentieren
- Getestete Teile der Gerätekommunikation:
  - VISA-Protokoll und PyVisa Package Installation
  - Kommunikation mit AWG von anderem Laptop aus über USB

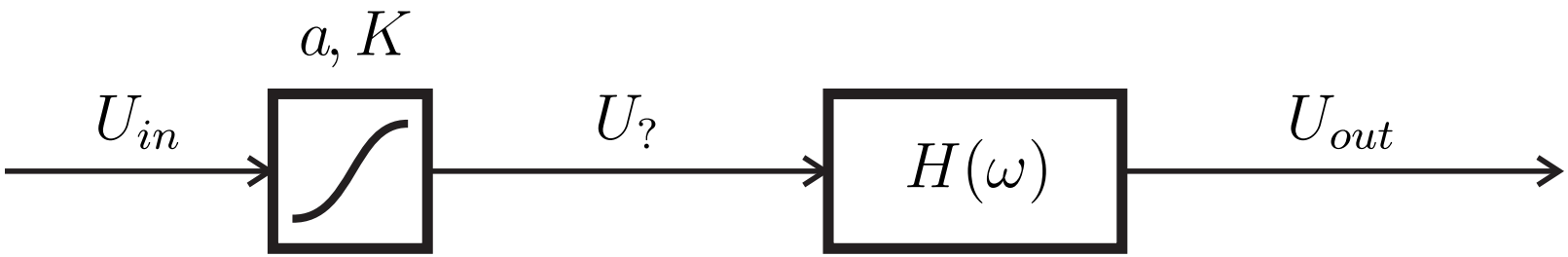
Dokumentation und Gerätekommunikation - Evaluierung

- Unvollständige Dokumentation:
  - Parameter und Return-Variablen dokumentieren
- Getestete Teile der Gerätekommunikation:
  - VISA-Protokoll und PyVisa Package Installation
  - Kommunikation mit AWG von anderem Laptop aus über USB
- Ausstehende Teile der Gerätekommunikation:
  - Kommunikation mit Oszilloskop von anderem Laptop
  - Laufzeit: Status-Abfrage der Geräte mit `BUSY?` oder `*WAI`
  - Anpassung der Auflösung des DSO

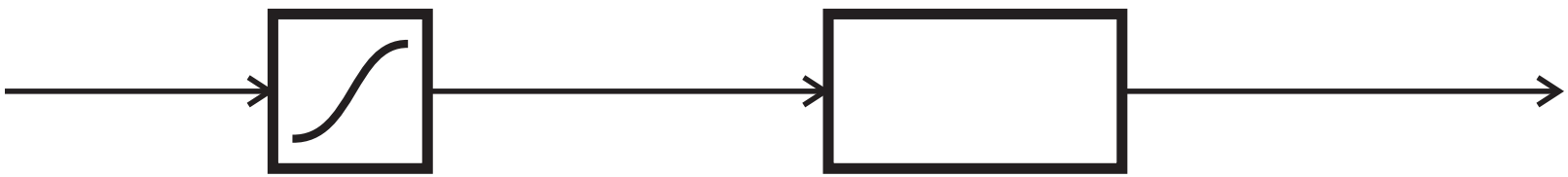
Code: Das Design



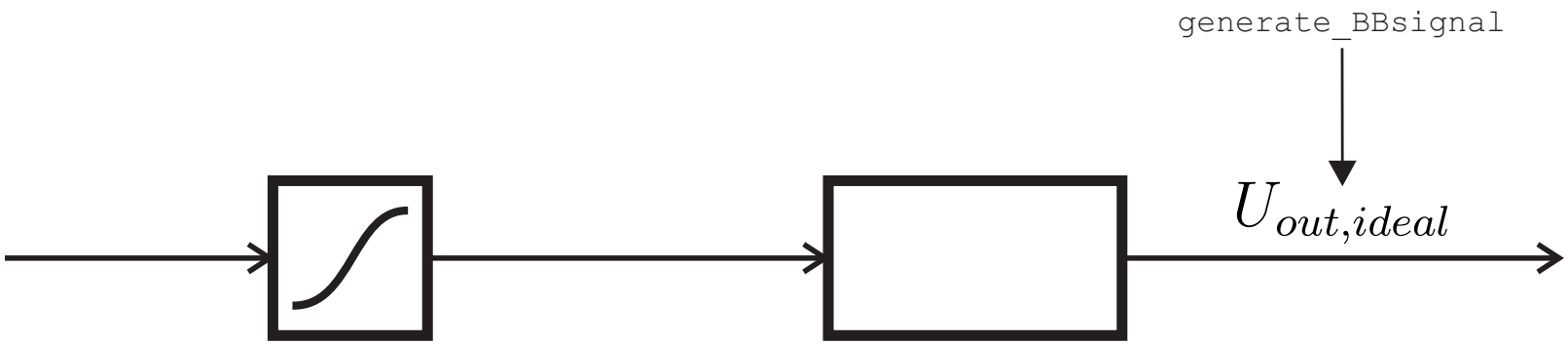
Code: Das Design



Code: Das Design

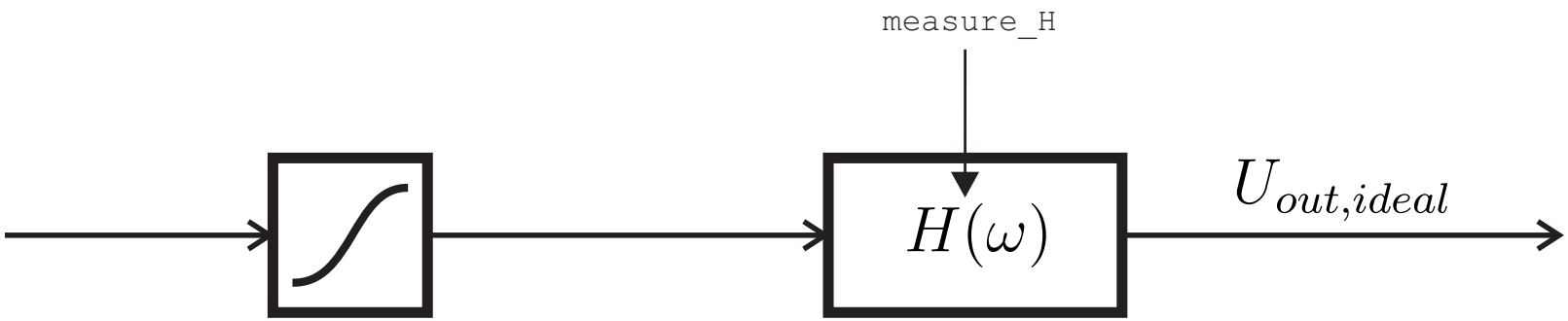


Code: Das Design



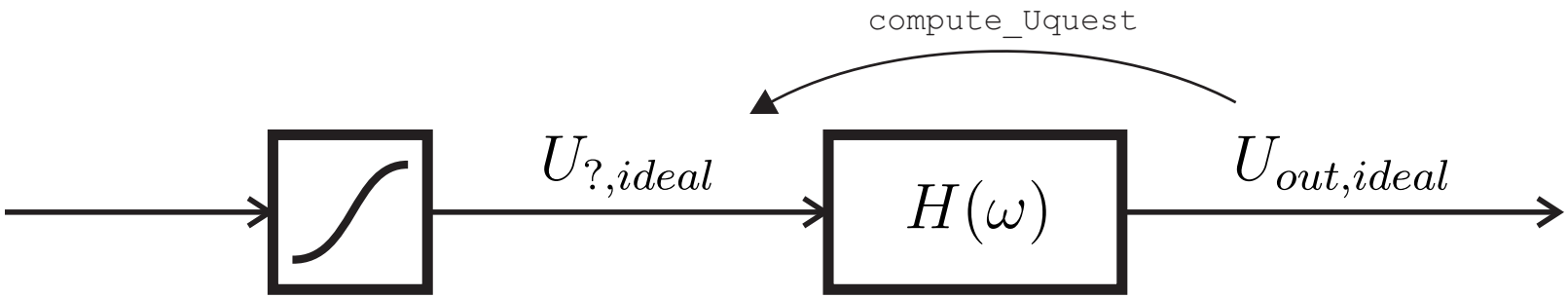

```
1 Uout_ideal = generate_BBsignal ( fq_rep , fq_bb , vpp )
```

Code: Das Design



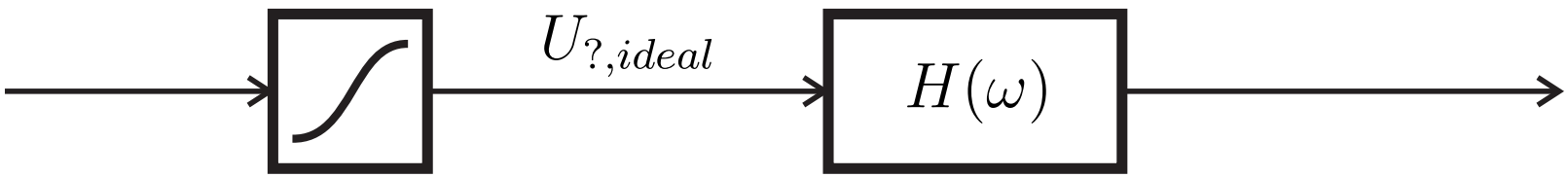

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
```

Code: Das Design



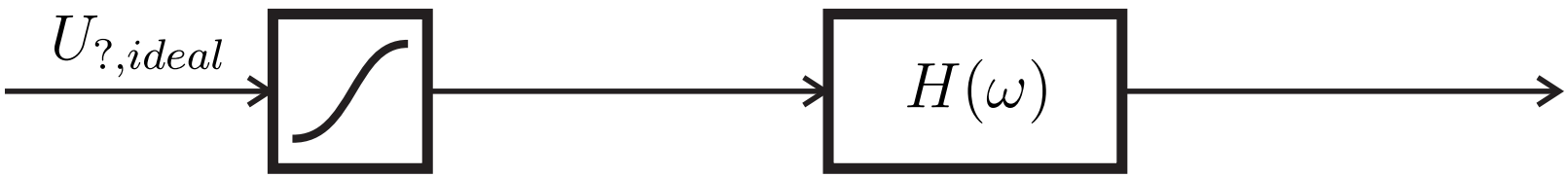

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```

Code: Das Design



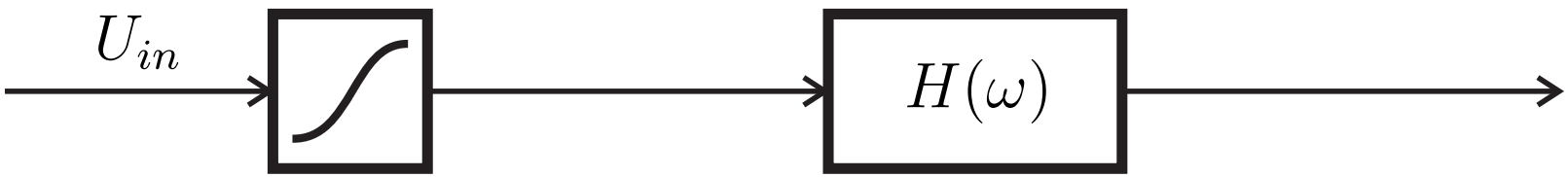

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```

Code: Das Design



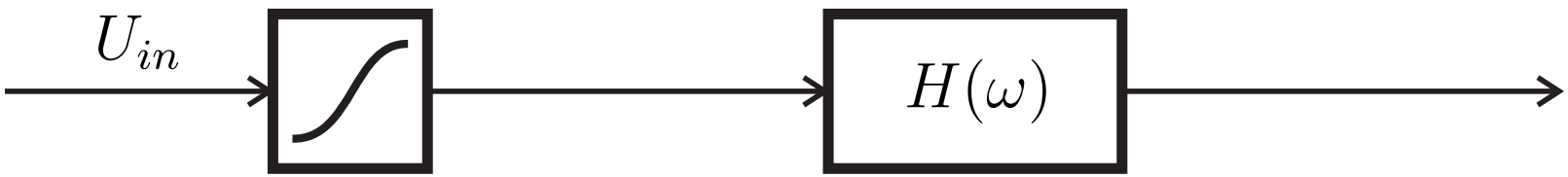

```
1 Uout_ideal = generate_BBsignal ( fq_rep , fq_bb , vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal , H )
```

Code: Das Design



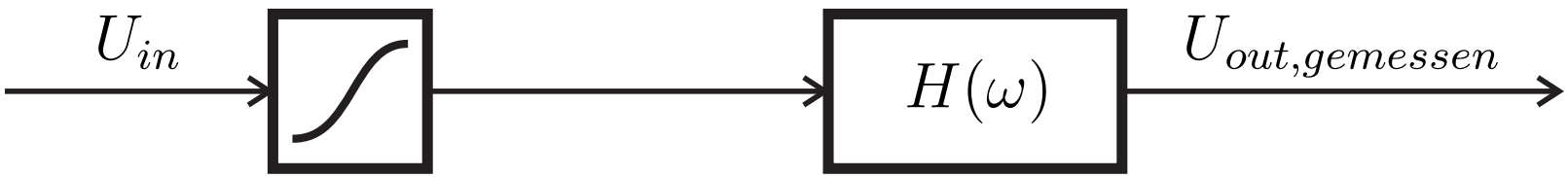

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
```

Code: Das Design



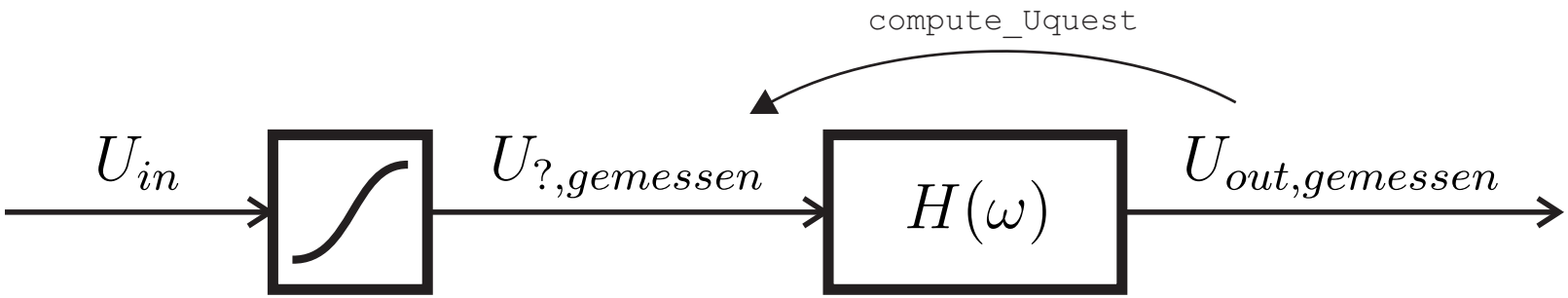

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
```

Code: Das Design



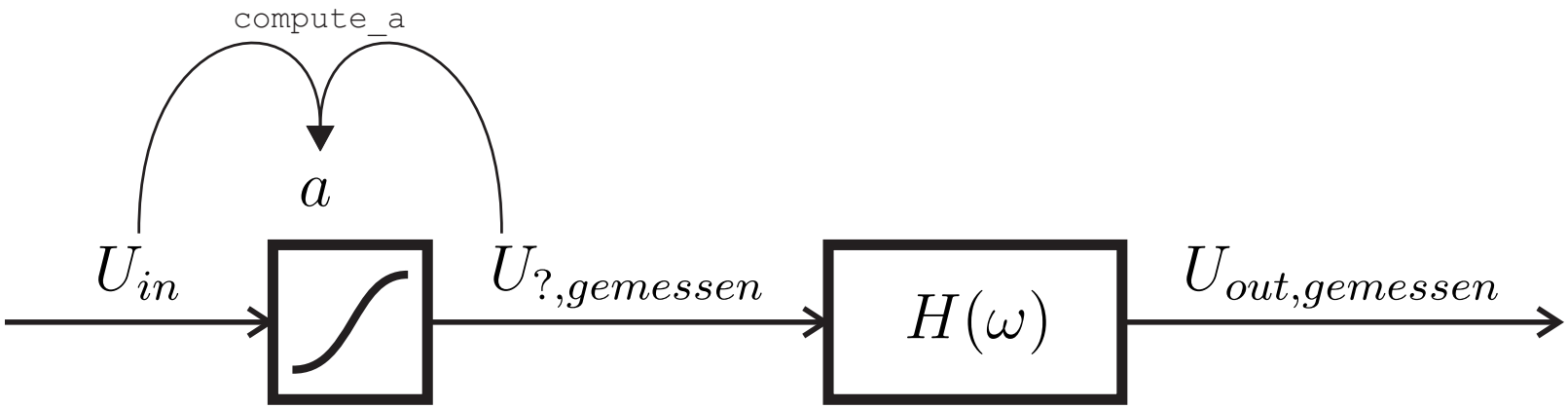

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
```

Code: Das Design



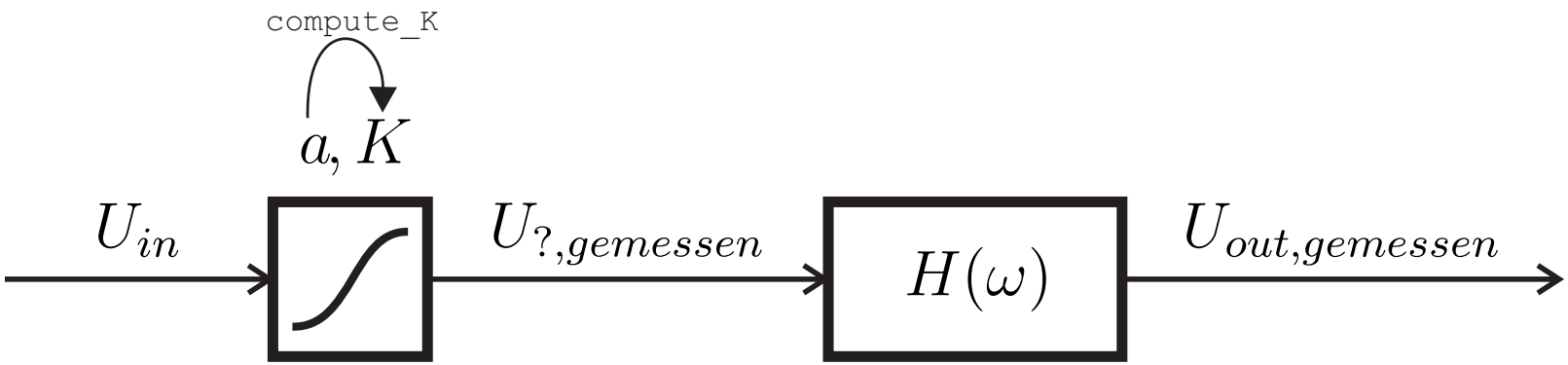

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
```

Code: Das Design



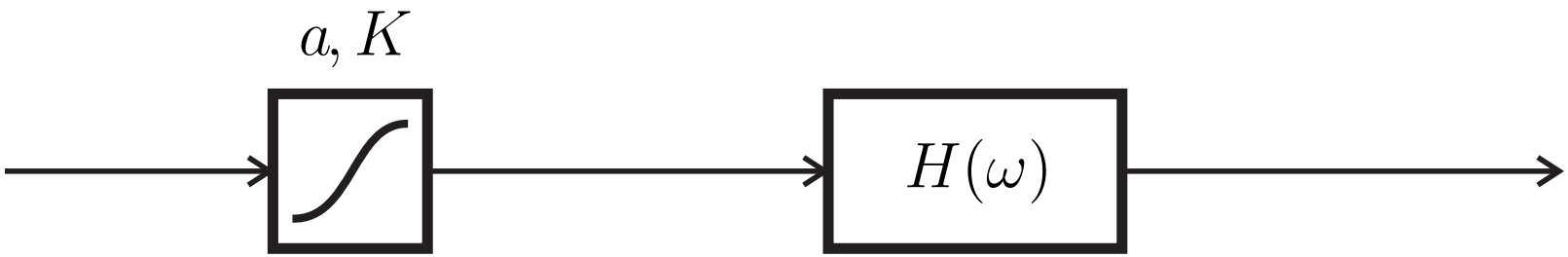

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
7 a = compute_a ( Uin, Uquest_measured, N )
```

Code: Das Design



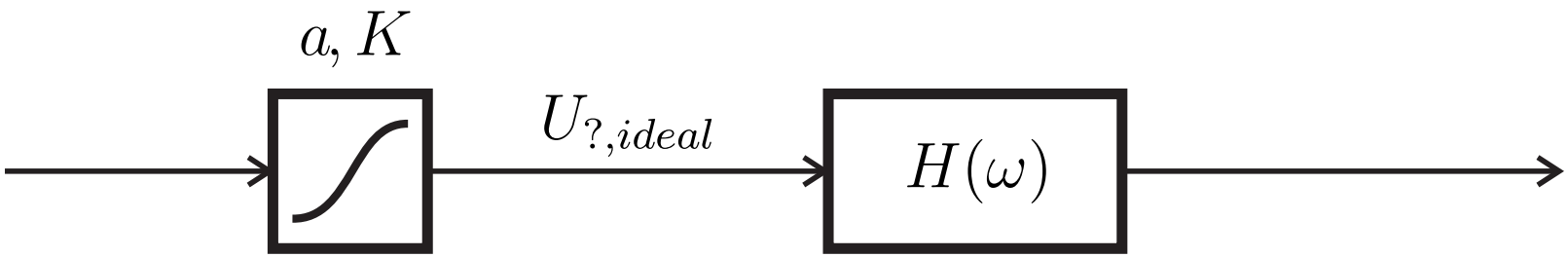

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Das Design



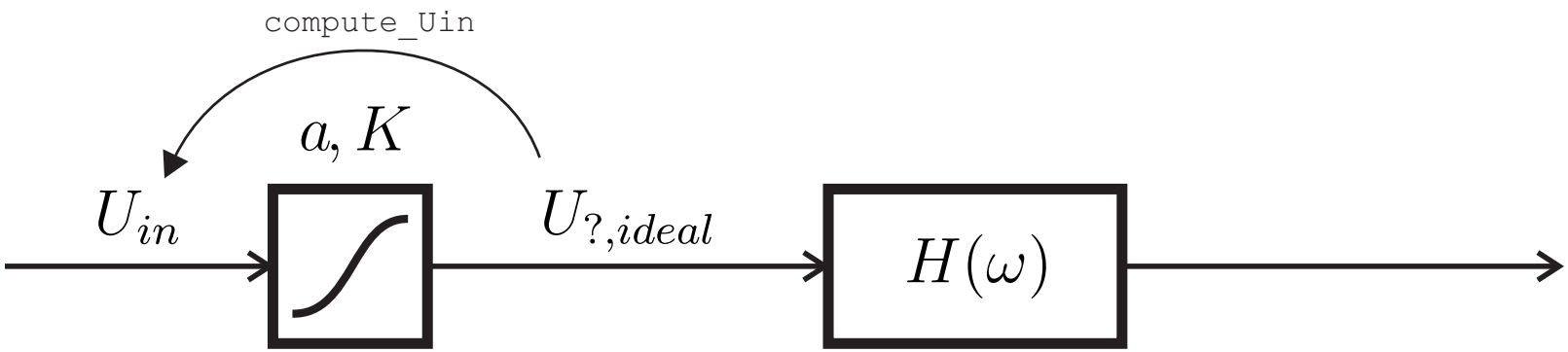

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Das Design



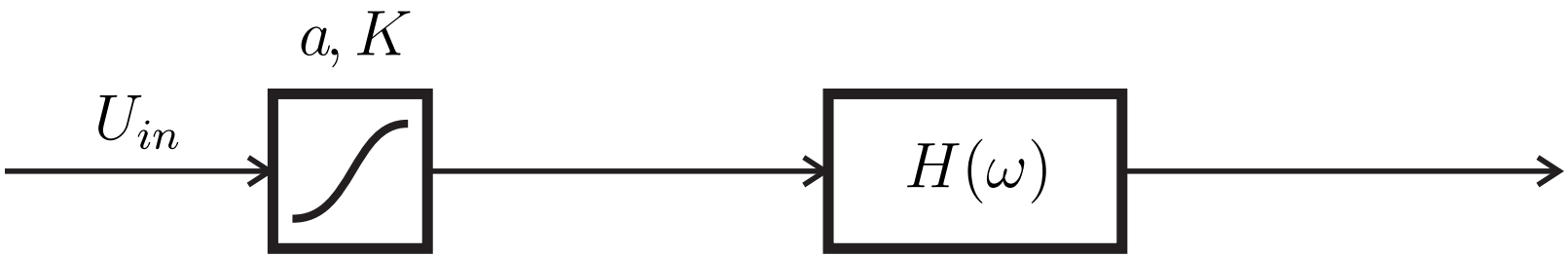

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Das Design



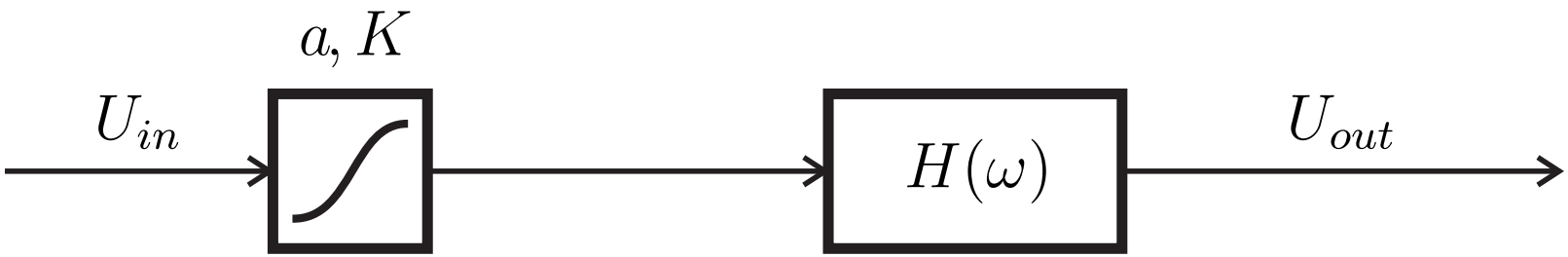

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
7 a = compute_a ( Uin, Uquest_measured, N )
8 K = compute_K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

Code: Das Design




```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_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: Das Design




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

Code: die Bausteine

Code: die Bausteine

```
generate_BBsignal
measure_H
compute_Uquest
compute_Uin
measure_Uout
compute_a
compute_K
```

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H`

`compute_Uquest`

`compute_Uin`

`measure_Uout`

`compute_a`

`compute_K`

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

`compute_Uquest`

`compute_Uin`

`measure_Uout`

`compute_a`

`compute_K`

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

`compute_Uquest` : zum Teil gegeben in Matlab und Python

`compute_Uin`

`measure_Uout`

`compute_a`

`compute_K`

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

`compute_Uquest` : zum Teil gegeben in Matlab und Python

`compute_Uin` : bereits gegeben in Matlab

`measure_Uout`

`compute_a`

`compute_K`

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

`compute_Uquest` : zum Teil gegeben in Matlab und Python

`compute_Uin` : bereits gegeben in Matlab

`measure_Uout` : `writeAWG.py` und `readDSO.py` waren gegeben

`compute_a`

`compute_K`

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

`compute_Uquest` : zum Teil gegeben in Matlab und Python

`compute_Uin` : bereits gegeben in Matlab

`measure_Uout` : `writeAWG.py` und `readDSO.py` waren gegeben

`compute_a` : bereits gegeben in Matlab

`compute_K`

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

`compute_Uquest` : zum Teil gegeben in Matlab und Python

`compute_Uin` : bereits gegeben in Matlab

`measure_Uout` : `writeAWG.py` und `readDSO.py` waren gegeben

`compute_a` : bereits gegeben in Matlab

`compute_K` : bereits gegeben in Matlab und Python

Code: die Bausteine

`generate_BBsignal` : musste implementiert werden

`measure_H` : bereits gegeben in Python

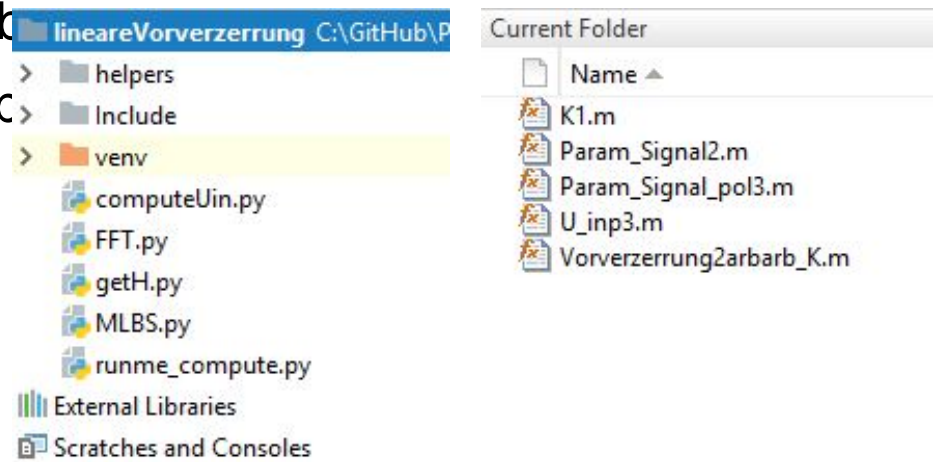
`compute_Uquest` : zum Teil gegeben in Matlab und Python

`compute_Uin` : bereits gegeben in Matlab

`measure_Uout` : `writeAWG.py` und `readDSO.py` waren gegeben

`compute_a` : bereits gegeben in Matlab

`compute_K` : bereits gegeben in Matlab



Code: Vorgehensweise

Code: Vorgehensweise

- Refactoring / Anpassung der Matlab-Funktionen an unser Design

Code: Vorgehensweise

- Refactoring / Anpassung der Matlab-Funktionen an unser Design
- Portierung der Matlab-Funktionen nach Python

Code: Vorgehensweise

- Refactoring / Anpassung der Matlab-Funktionen an unser Design
- Portierung der Matlab-Funktionen nach Python
- Überprüfung der portierten Funktionen mithilfe von **TDD**

Code: Vorgehensweise

- Refactoring / Anpassung der Matlab-Funktionen an unser Design
- Portierung der Matlab-Funktionen nach Python
- Überprüfung der portierten Funktionen mithilfe von **TDD**
  - Momentan jeweils 10 korrespondierende **Unit Tests** in Matlab und Python

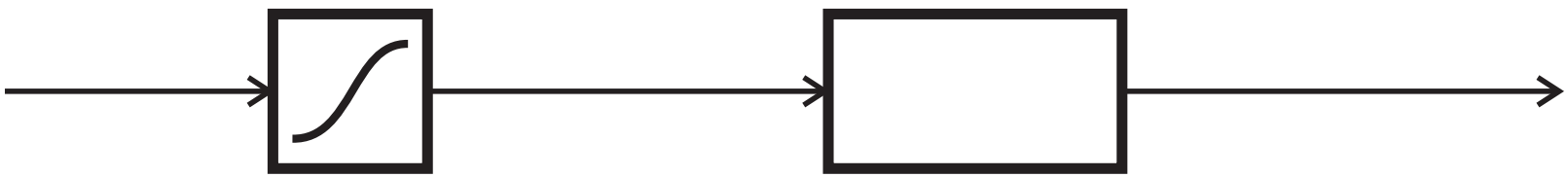
Code: Vorgehensweise

- Refactoring / Anpassung der Matlab-Funktionen an unser Design
- Portierung der Matlab-Funktionen nach Python
- Überprüfung der portierten Funktionen mithilfe von **TDD**
  - Momentan jeweils 10 korrespondierende **Unit Tests** in Matlab und Python
- Maximale Vorbereitung der Funktionen ohne Messaufbau dank **TDD**

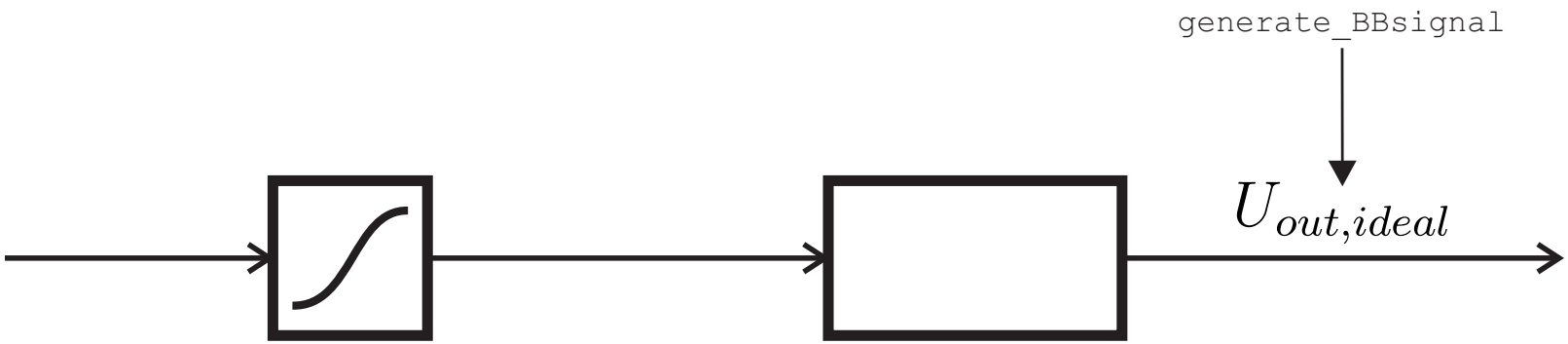
Code: Vorgehensweise

- Refactoring / Anpassung der Matlab-Funktionen an unser Design
- Portierung der Matlab-Funktionen nach Python
- Überprüfung der portierten Funktionen mithilfe von **TDD**
  - Momentan jeweils 10 korrespondierende **Unit Tests** in Matlab und Python
- Maximale Vorbereitung der Funktionen ohne Messaufbau dank **TDD**
  - Nur zum Testen von `measure_Uout` sind Geräte notwendig

Code: Evaluierung

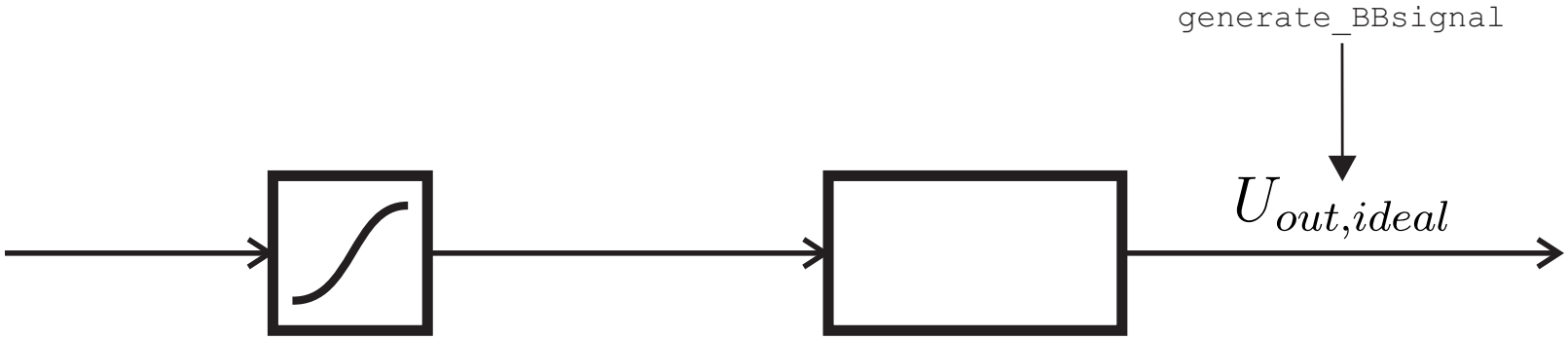


Code: Evaluierung

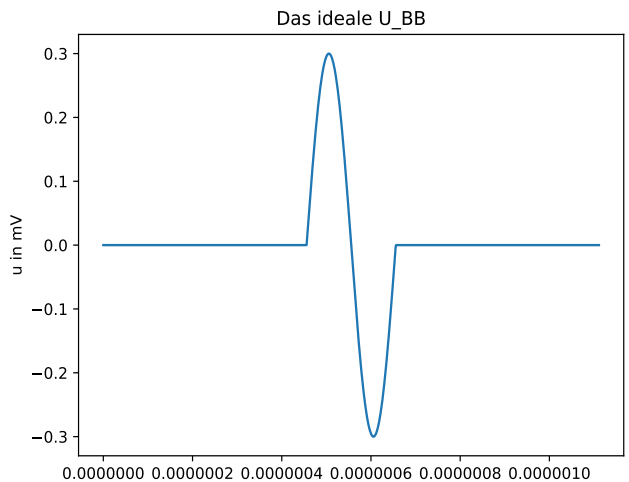



```
1 Uout_ideal = generate_BBsignal ( fq_rep , fq_bb , vpp )
```

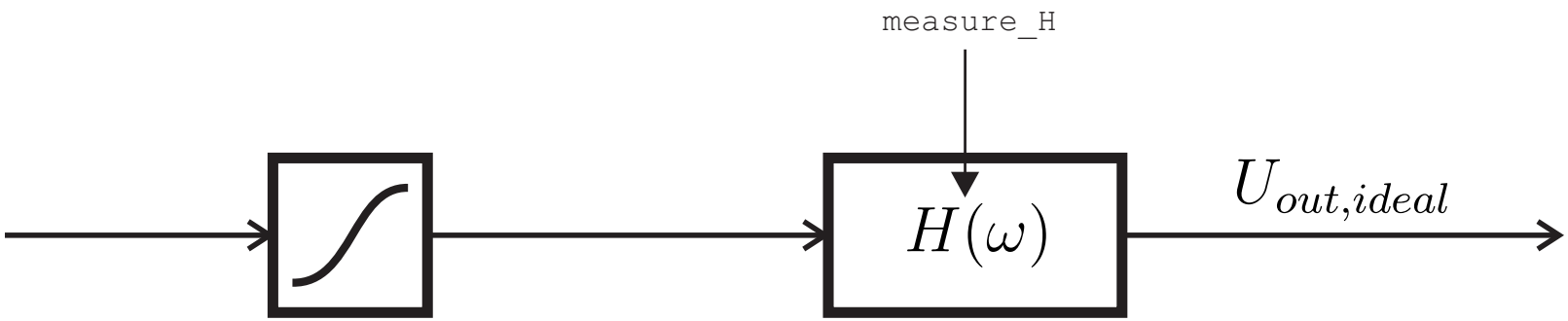
Code: Evaluierung




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

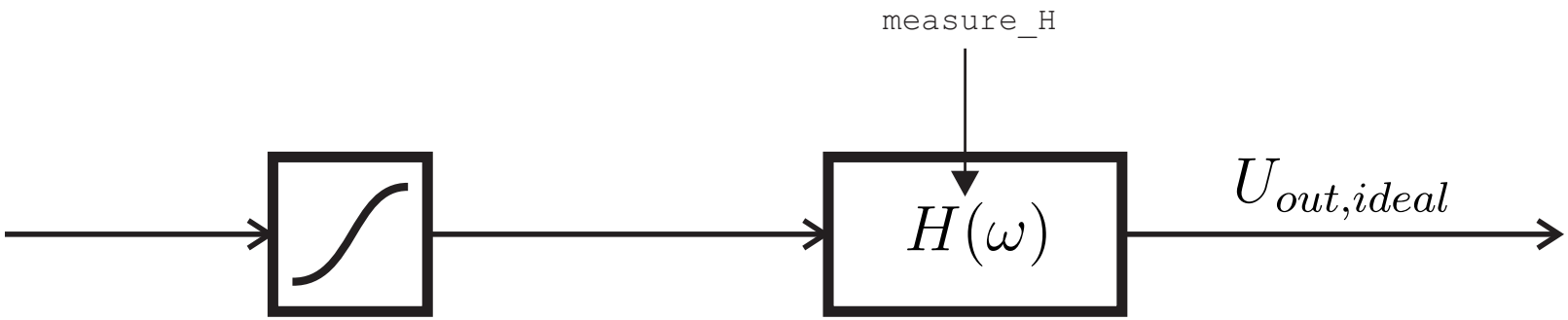


Code: Evaluierung

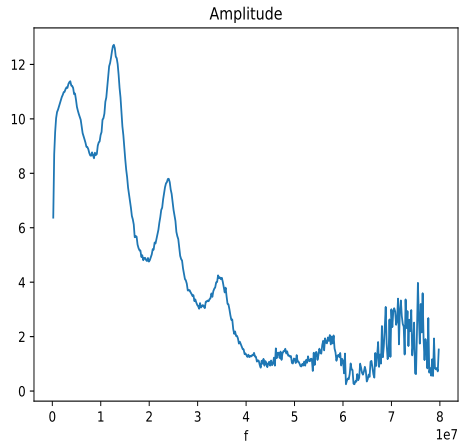


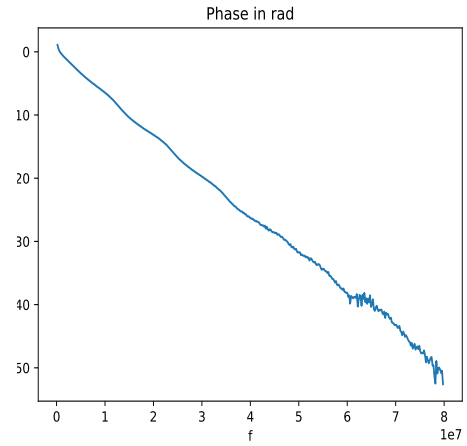

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
```

Code: Evaluierung

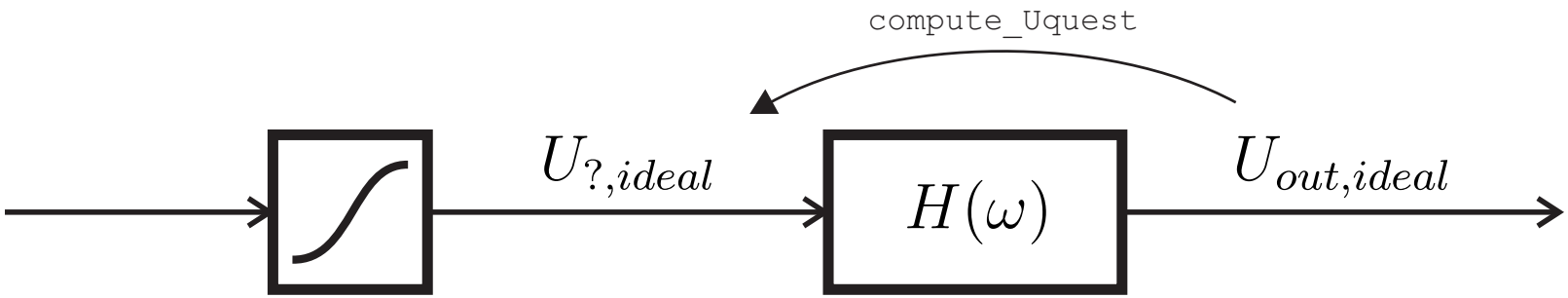



```
1 Uout_ideal = generate_BBsignal ( fq_rep ,
2 H = measure_H ( )
```



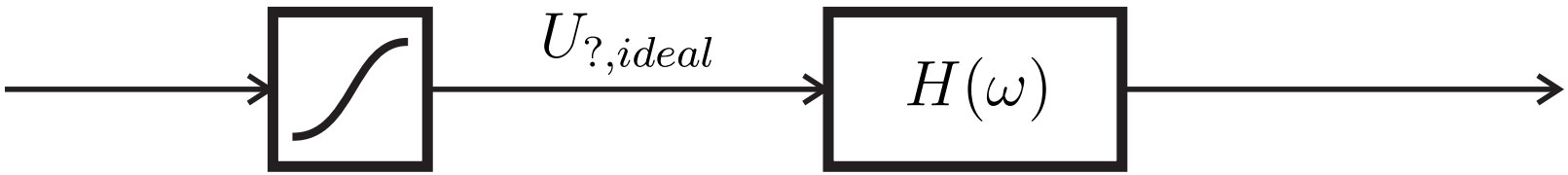


Code: Evaluierung

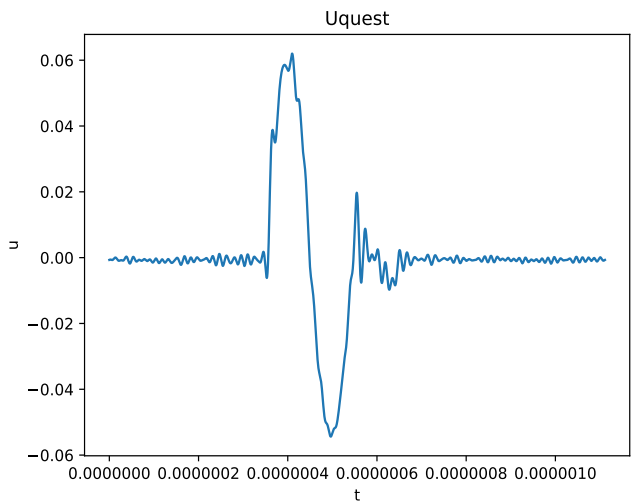



```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```

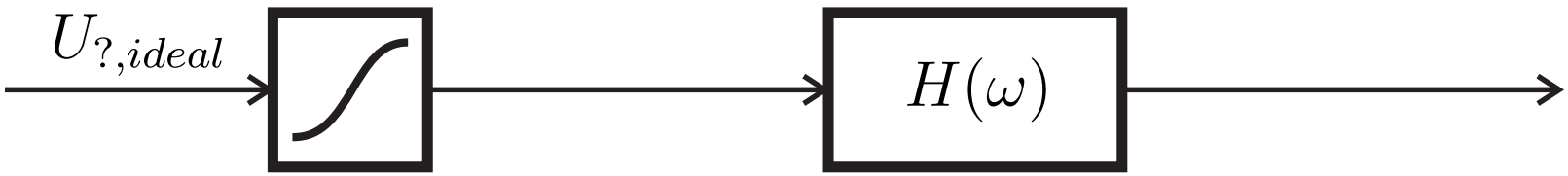
Code: Evaluierung



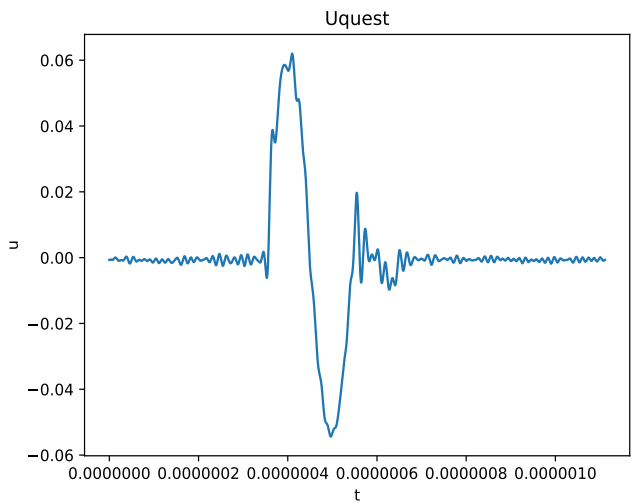

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```



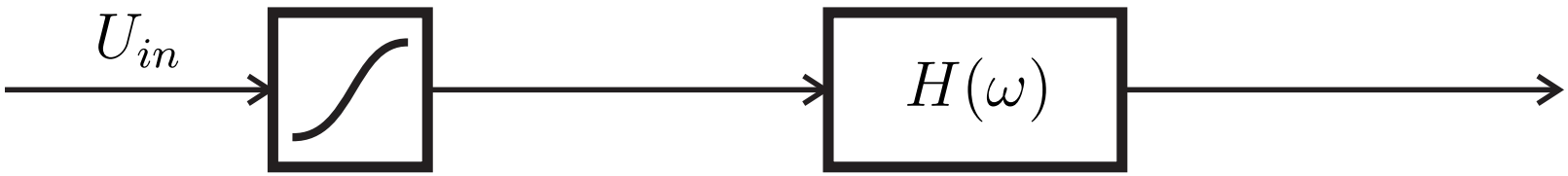
Code: Evaluierung



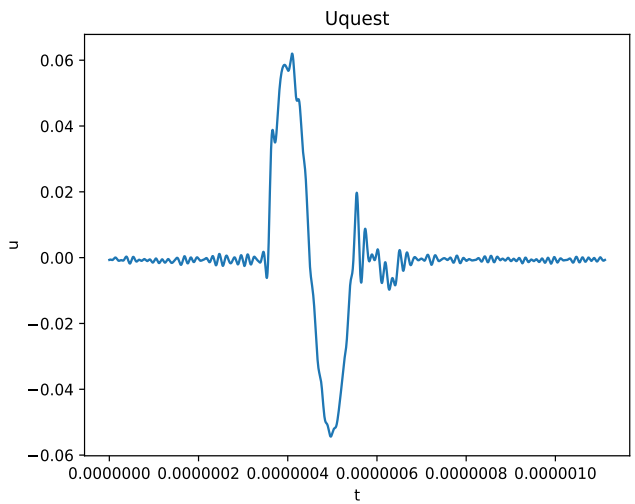

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
```



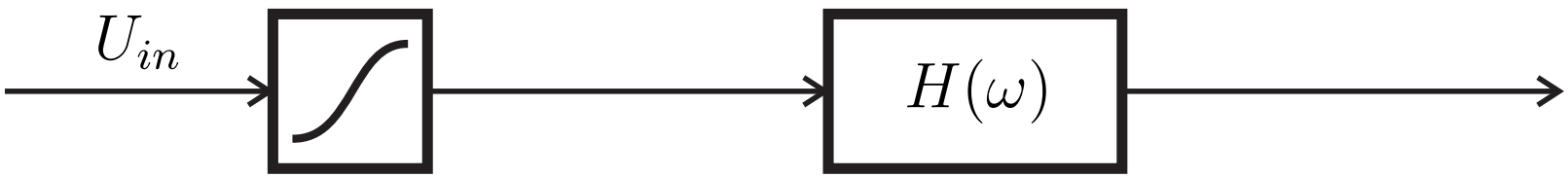
Code: Evaluierung




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

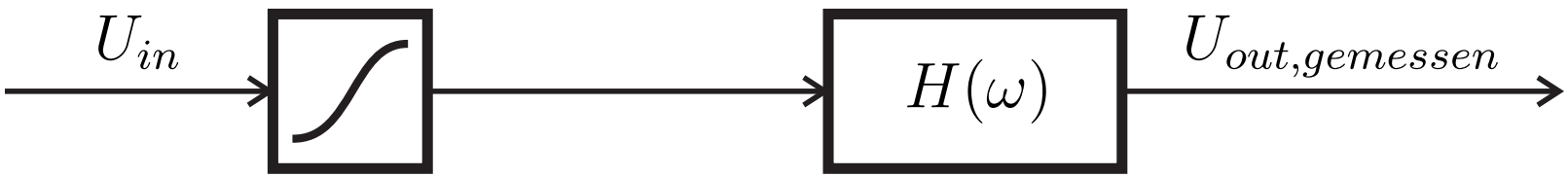


Code: Evaluierung



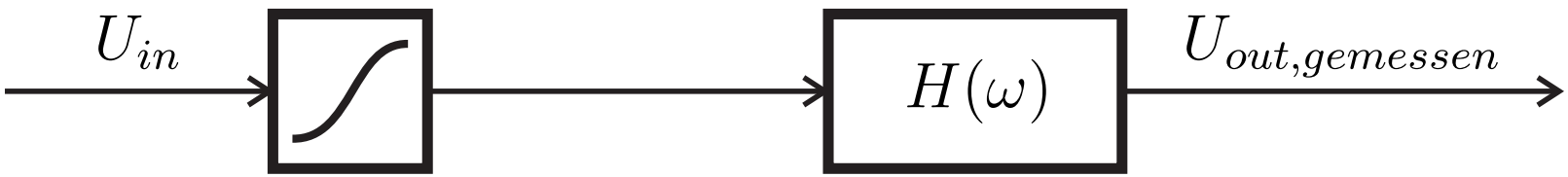

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
```

Code: Evaluierung

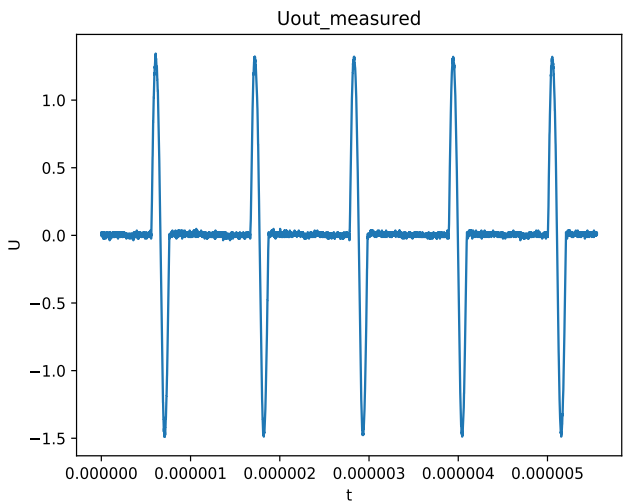



```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
```

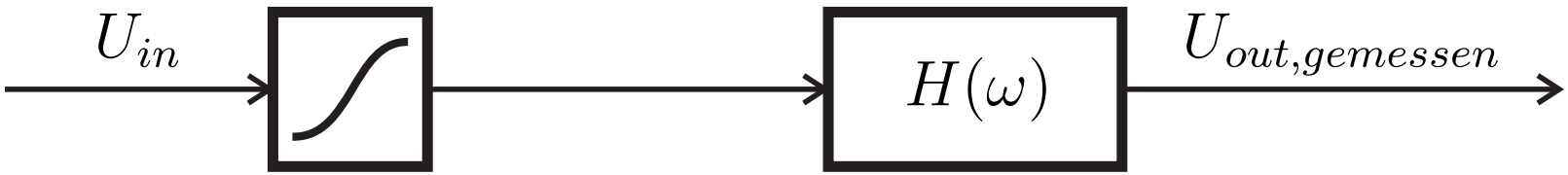
Code: Evaluierung



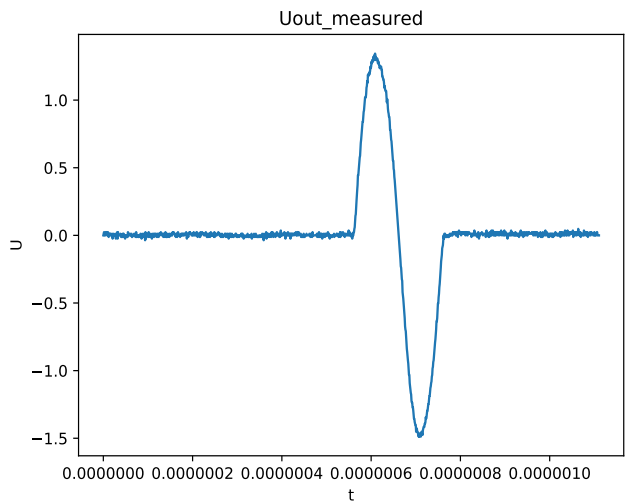

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
```



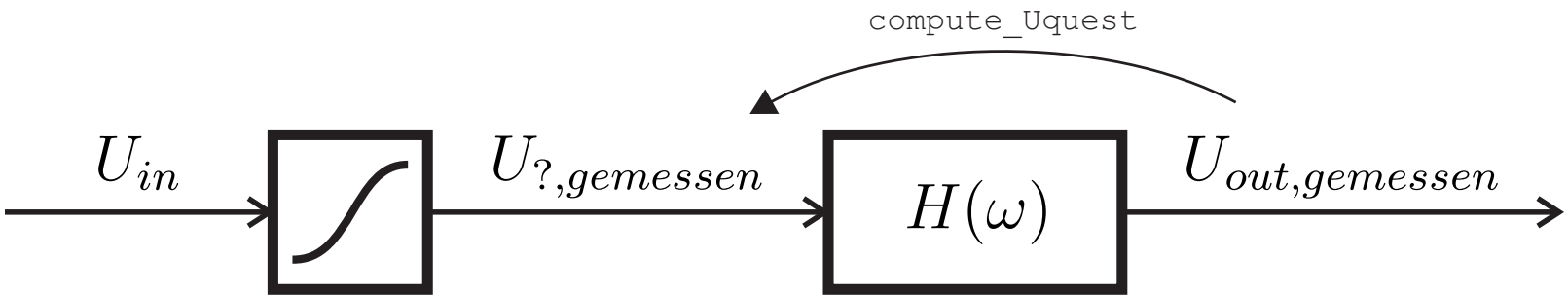
Code: Evaluierung




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

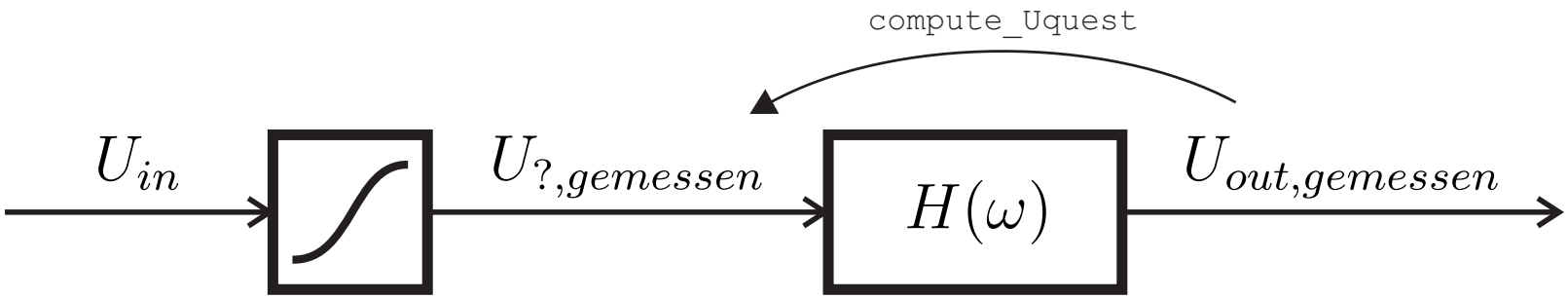


Code: Evaluierung

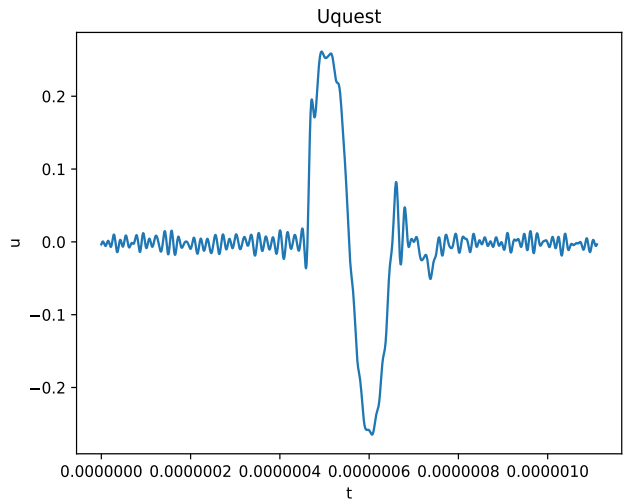



```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
```

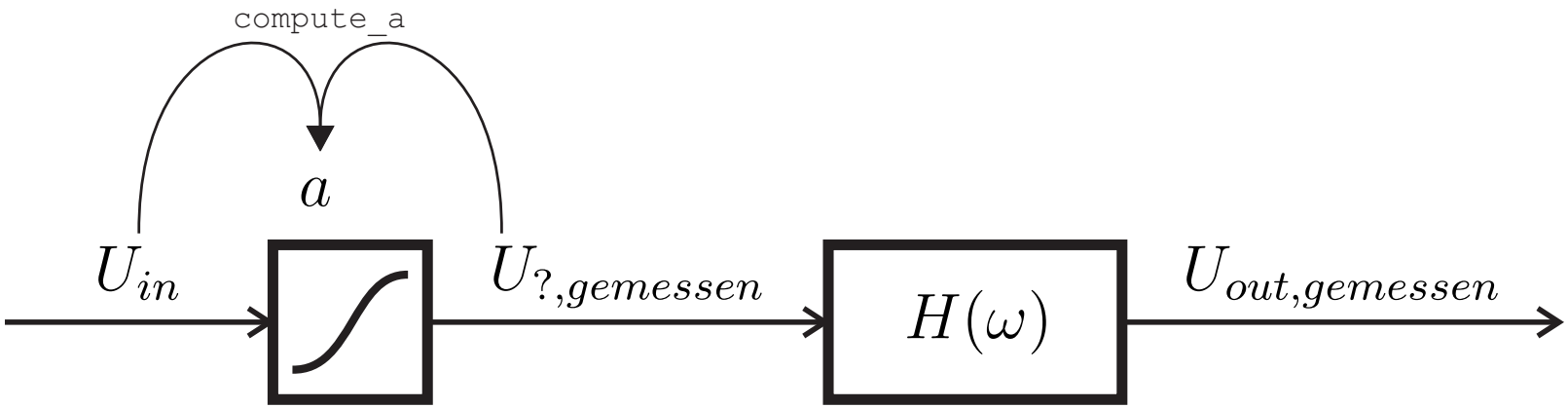
Code: Evaluierung




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

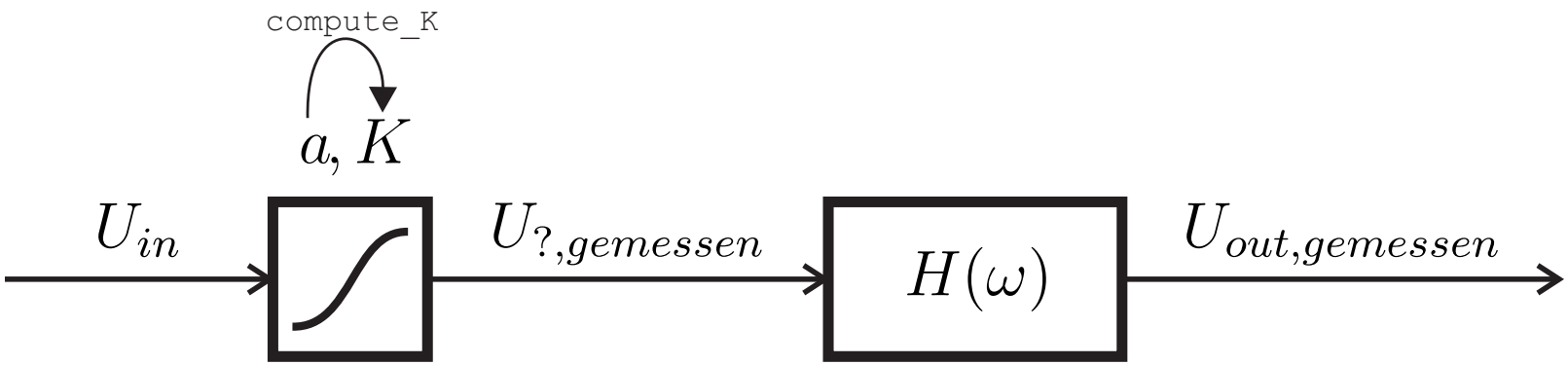


Code: Evaluierung



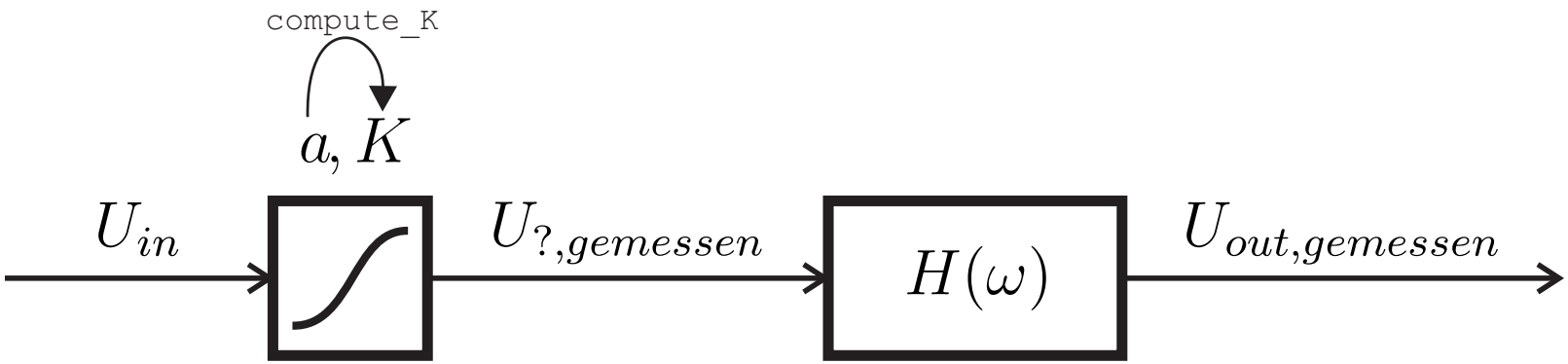

```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
7 a = compute_a ( Uin, Uquest_measured, N )
```

Code: Evaluierung

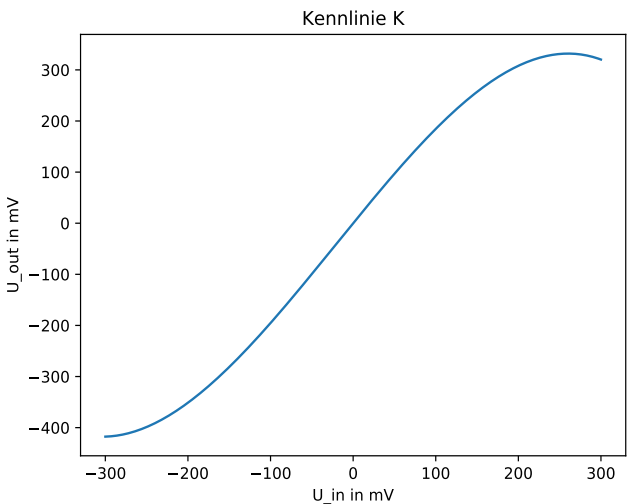



```
1 Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
2 H = measure_H ( )
3 Uquest_ideal = compute_Uquest ( Uout_ideal, H )
4 Uin = Uquest_ideal
5 Uout_measured = measure_Uout ( Uin )
6 Uquest_measured = compute_Uquest ( Uout_measured, H )
7 a = compute_a ( Uin, Uquest_measured, N )
8 K = compute_K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

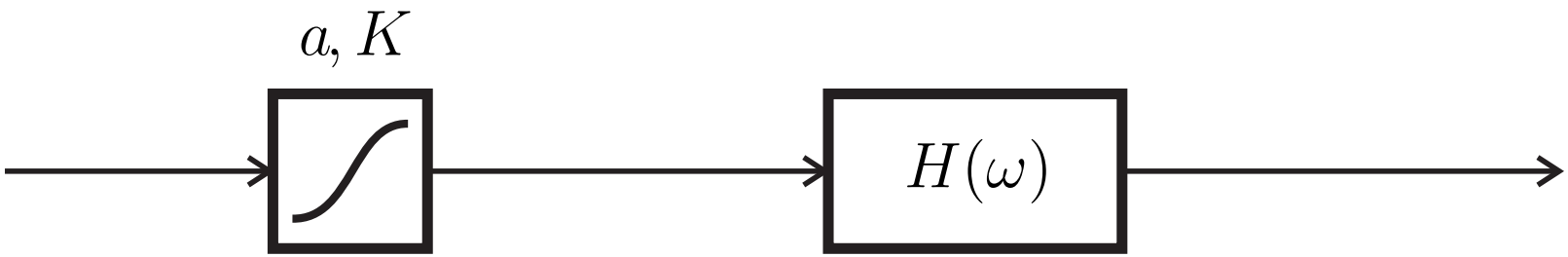
Code: Evaluierung




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

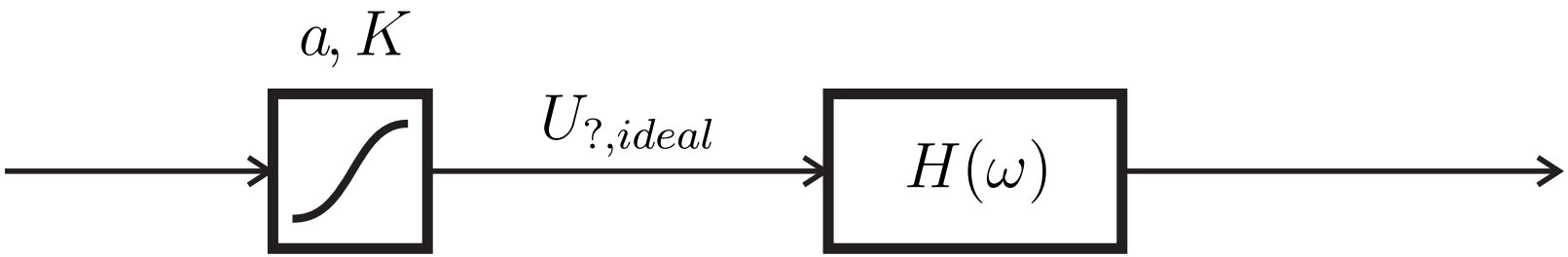


Code: Evaluierung



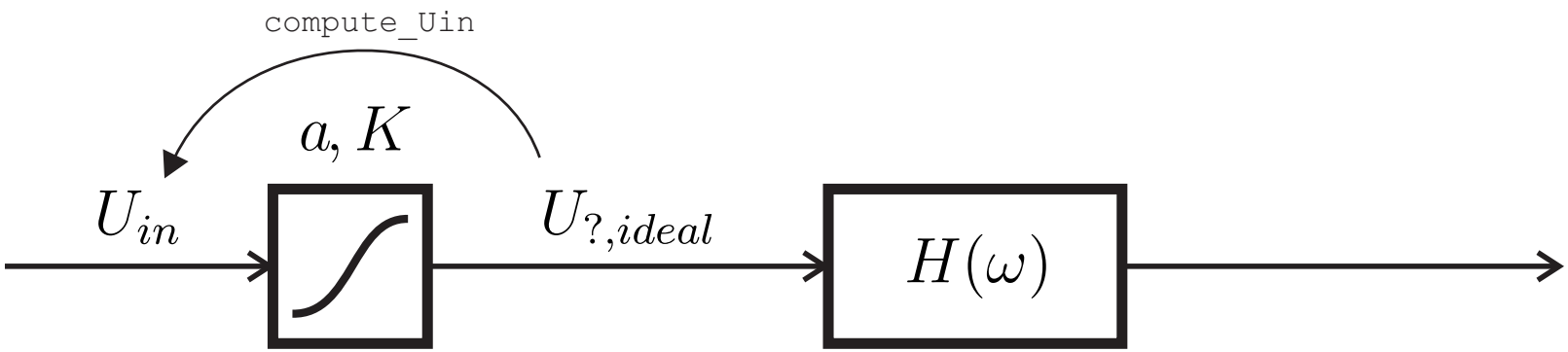

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Evaluierung



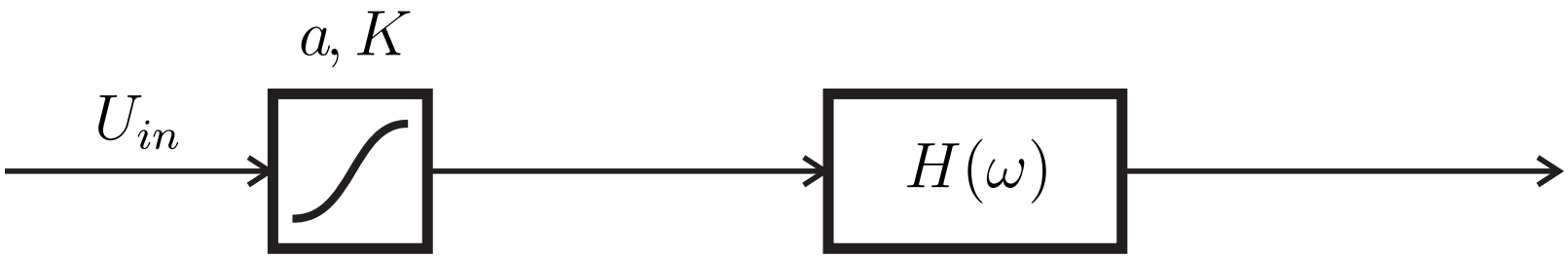

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
```

Code: Evaluierung



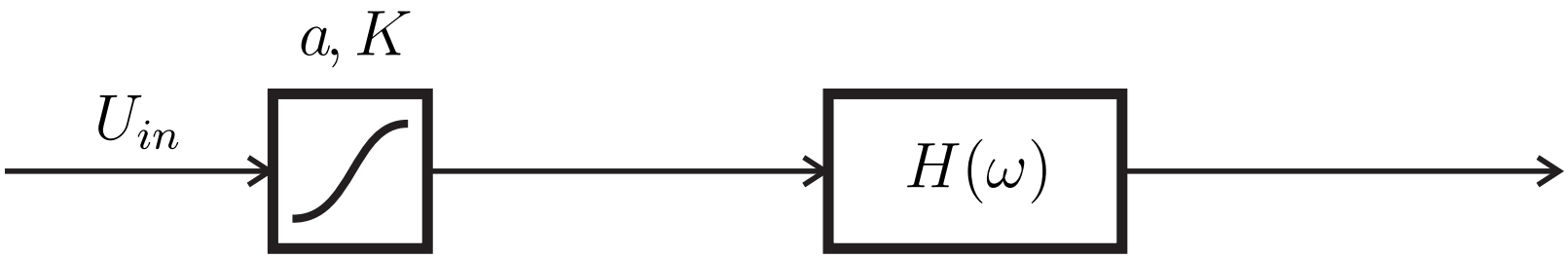

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
Uin =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

Code: Evaluierung

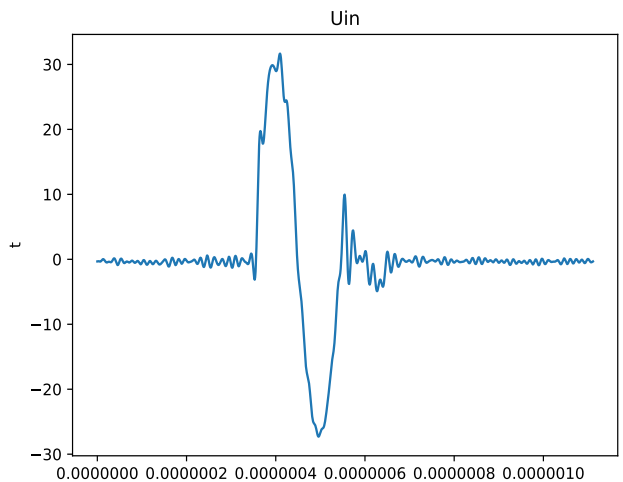



```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
Uin =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

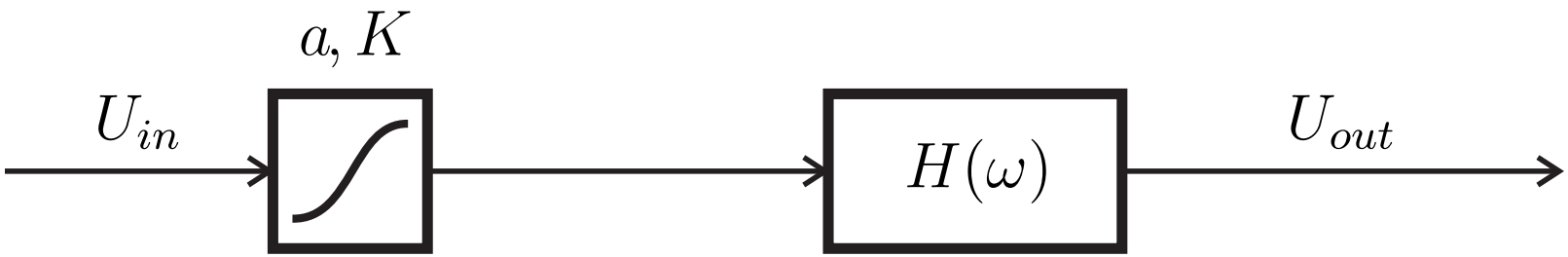
Code: Evaluierung




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



Code: Evaluierung



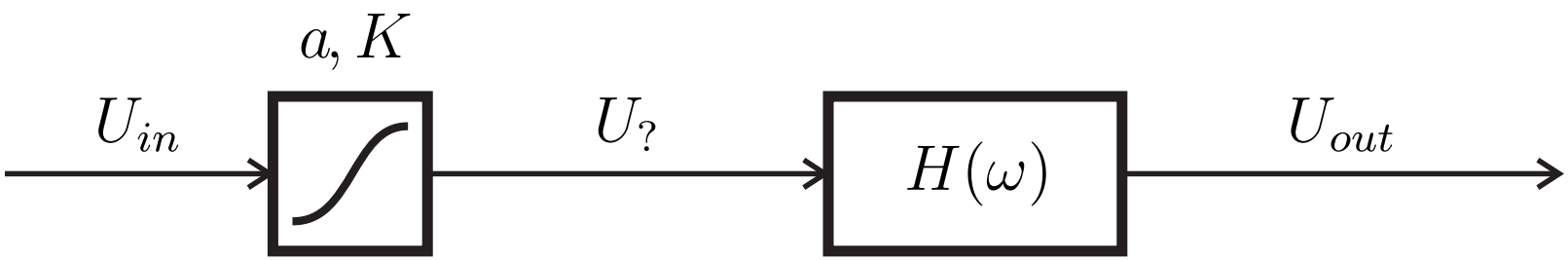

```
Uout_ideal = generate_BBsignal ( fq_rep, fq_bb, vpp )
H = measure_H ( )
Uquest_ideal = compute_Uquest ( Uout_ideal, H )
Uin = Uquest_ideal
Uout_measured = measure_Uout ( Uin )
Uquest_measured = compute_Uquest ( Uout_measured, H )
a = compute_a ( Uin, Uquest_measured, N )
K = compute_K ( a )
Uin =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

Ausblick

- Iterative Optimierung der linearen Übertragungsfunktion mittels Auswertung der erwarteten und gemessenen Ausgangssignale U_{out}:

$$\underline{H}^{\text{neu}}(\omega) = \underline{H}^{\text{alt}}(\omega) \cdot \frac{\underline{U}_{out,\text{ideal}}(\omega)}{\underline{U}_{out,\text{mess}}(\omega)} \cdot \sigma_H$$

  mit σ_H als Schrittweite der jeweiligen Iteration

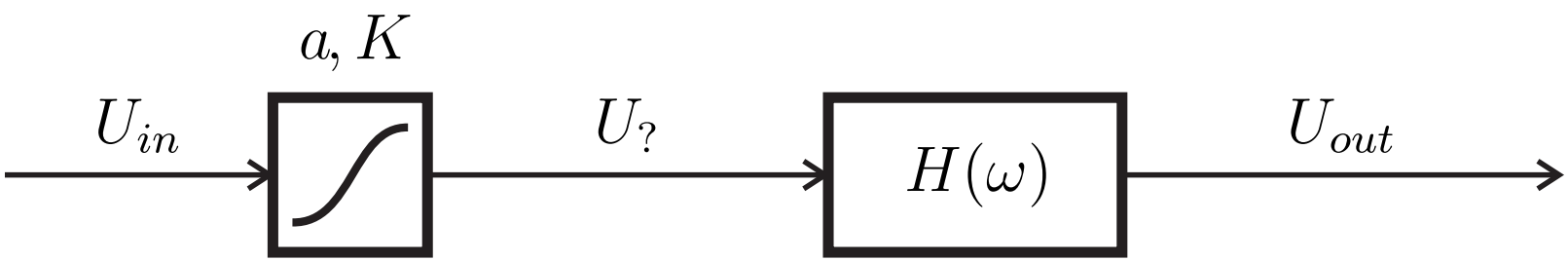


Ausblick

- Optimierung der nichtlinearen Kennlinie mittels Vergleich der Differenz der erwarteten und gemessenen Spannungssignale U_{quest} und der Faktoren a der polynomialen Kennlinie:

$$\Delta U_? = U_{?,\text{mess}} - U_{?,\text{berechnet}} = \sum_n \tilde{a}_n U_{in}^n \qquad a_n^{\text{neu}} = a_n^{\text{alt}} + \sigma_a \cdot \tilde{a}_n$$

  mit σ_a als Schrittweite der jeweiligen Iteration



Offene Fragen

- Reihenfolge der Optimierung: Parallele Iteration $\Leftrightarrow$ alternierende Iteration von H und K
- Einfluss von K auf das Spektrum von $U_?$ und damit auf Optimierung von H durch Oberschwingungen bei Potenzierung des Eingangssignals
- Bewertung der Qualität des Ausgangssignals nach einem Iterationsschritt

Quellen

- Denys Bast, Armin Galetzka, "Projektseminar Beschleunigertechnik", 2017
- Jens Harzheim *et al.*, "Input Signal Generation For Barrier Bucket RF Systems At GSI",
- Kerstin Gross *et al.*, "Test Setup For Automated Barrier Bucket Signal Generation", 2017