# Signale & Systeme
# Analyse eines Systems

**Matrikelnr.:** 9085107 & 2147057
**Dozent:** Apl. Prof. Dr. Lutz Gröll
**Kurs:** TINF22B1
**System:** $\mathrm{PT_1T}_t$-Glied

25. November 2023

# 1 Vorgehen in Regelungstechnik

Zur Ermöglichung einer strukturierten Vorgehensweise bedarf es einer Orientierung. Dieser Arbeit liegt die in der Vorlesung behandelte Vorgehensweise in der Regelungstechnik zugrunde, welche nachfolgend erläutert wird.

1. Detaillierungsgrad für das System festlegen
2. Physikalisches Modell einschließlich der Störsignale erstellen
3. Eingangs-, Ausgangs-, Zustands- und Störgrößen des Systems festlegen
4. Physikalische Einheiten festlegen und ggf. Normierung in Prozent
5. Analyse des Modells
   (a) Ruhelagen, Anfangswerte, evtl. Linearisierung nichtlinearer Systeme
   (b) Systemeigenschaften ermitteln
6. Entwurf
   (a) Ziele, Arbeitspunkte und Trajektorien festlegen
   (b) Entwurf: Struktur, Bereich, Verfahren und Kriterien
7. Simulation und Rückinterpretation

In dieser Vorlesung wird die Regelungstechnik außen vor gelassen, weswegen im Folgenden eine leicht abgewandelte Struktur angewendet wird. Bezüglich des Detaillierungsgrads sind alle in der Vorlesung behandelten Darstellungsformen darzulegen. Weiterführend werden die Punkte 3 und 4 des Vorgehens aus der Regelungstechnik zusammengefasst und tabellarisch dargestellt. Schwerpunktmäßig wird in dieser Arbeit Punkt 5 behandelt, in dem alle Darstellungsformen aus der Vorlesung dargelegt werden. Punkt 6 und 7 werden in dieser Arbeit nicht betrachtet.

# 2 Unser System

Wir betrachten einen Wassertank, der oben an einen Zufluss mit sehr hohem Druck angeschlossen ist. Am unten Ende des Wassertanks befindet sich ein deutlich kleiner Abfluss, an den eine lange Wasserleitung angeschlossen ist. Erst am Ende der Wasserleitung befindet sich ein Messgerät, welcher den Volumenstrom misst. Eine beispielhafte Skizze sieht folgendermaßen aus:

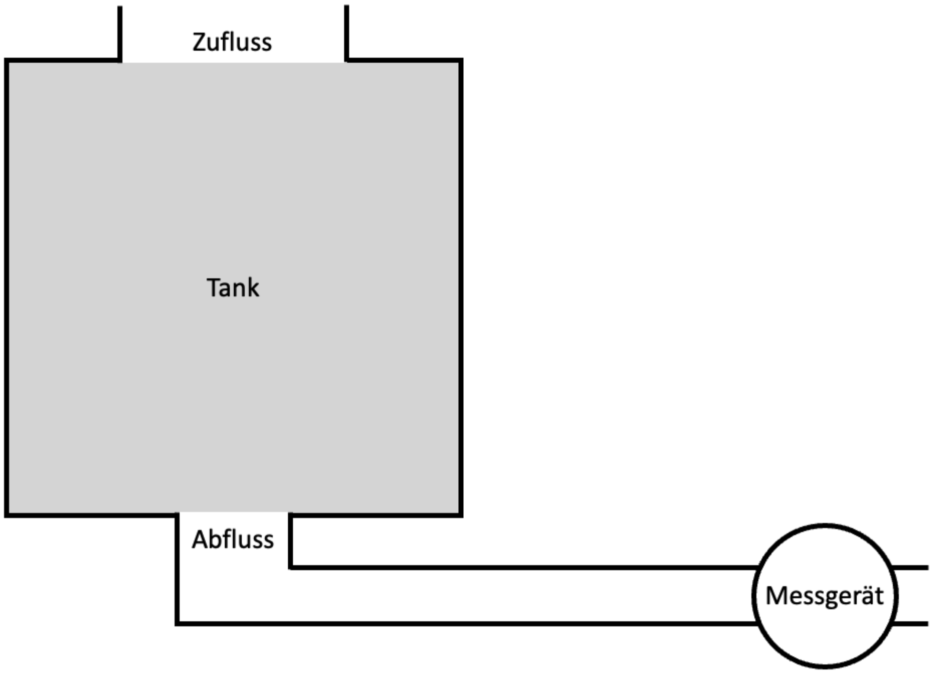


Abbildung 1: Skizze des Systems

Der Ablauf stellt sich wie folgt dar: Über den Zufluss fließt Wasser in den Tank, da der Ausfluss aber deutlich kleiner ist, fließt erstmal deutlich weniger Wasser heraus. Erst wenn der Tank voller wird, erhöht sich der Wasserdruck und es kommt zu einem immer größeren Volumenstrom bis schließlich der Tank voll ist und der Druck des Zuflusses dafür sorgt, dass der Volumenstrom am Ausfluss gleich dem am Abfluss ist. Da das Messgerät aber weit entfernt vom Tank liegt, misst dieses den ganzen Vorgang erst mit einer gewissen Verzögerung.

Je nachdem wie groß der Tank ist, verändert sich die Konstante $T_1$: Je kleiner der Tank, desto größer ist $T_1$. Gleichzeitig bestimmt der Abstand zwischen dem Tank und dem Messgerät die Zeitkonstante $T_t$: Je größer der Abstand, destor größer ist $T_t$.

## 2.1 Größen

Im folgenden Werden Eingangs-, Zustands- und Ausgangsgrößen des Systems definiert.

| Systemtheorie | Größe im System | Einheit |
|---|---|---|
| $u(t)$ | Volumenstrom am Zufluss | $\frac{l}{min}$ |
| $x(t)$ | Volumenstrom am Ausfluss | $\frac{l}{min}$ |
| $y(t)$ | gemessene Volumenstrom am Messgerät | $\frac{l}{min}$ |

## 2.2 Bezeichnung

Das System hat den Nennergrad 1, eine Totzeit $T_t$ und einem proportionalen Übertragungsverhalten. Daraus ergibt sich die Bezeichnung $PT_1T_t$.

## 2.3 Übertragungsfunktion

Formel 1 zeigt die Übertragungsfunktion die in diesem Skript analysiert wird. Formel 3 zeigt eine etwas andere Darstellung der Formel, sodass die Eigenschaften des Systems leichter abzulesen sind.

$$
\begin{aligned}
G(s) &= \frac{Y(s)}{U(s)} && (1)\\
&= \frac{1 * e^{-2s}}{(1+s)} && (2)\\
&= \frac{1}{1+s} * e^{-2s} && (3)
\end{aligned}
$$

### 2.4 Eingabe in MATLAB

Für die Eingabe der Übertragungsfunktion in MATLAB gibt es zwei Möglichkeiten.
In Computer-Algebra führt man mit `sys s` und `sys G(s)` die symbolischen Variablen ein und schreibt dann `G(s) = 1 / (1 + s) * exp(-2 * s)`.
In manueller Eingabe wird der Befehl `sys = tf([1], [1, 1],"IODelay", 2)` benötigt. Hierbei werden in den beiden Vektoren die Koeffizienten vor $s^0$, $s^1$, $s^2$, ... (im Nenner) angegeben. Bei komplizierteren Gleichungen hilft dabei der Befehl `expand(G(s))`, der einem die benötigten Koeffizienten liefert, was hier jedoch nicht nötig war. Darauf folgt der Befehl `"IODelay"`, mit dem die Totzeit $t = 2$ angegeben wird.
In beiden Fällen wird in MATLAB damit die Formel 3 dargestellt.

## 3 Darstellungsformen des Systems

Im Folgenden werden die Darstellungen unseres Systems näher beleuchtet. Generell gilt: Mit jeder weiteren Darstellung verliert man bestimmte Informationen über das System, sodass die erste Darstellung die informationsreichste ist.

### 3.1 Explizite Darstellung des Übertragungsoperators

### 3.2 Zustandsraumdarstellung

Der Zustandsraum lässt sich sowohl implizit als auch explizit darstellen.

#### 3.2.1 Implizite Darstellung

Die allgemeine Form der Zustandsraumdarstellung lautet:

$$
\begin{aligned}
\dot{x} &= Ax + Bu\\
y &= Cx + Du
\end{aligned}
$$

Mit dem MATLAB-Befehl `ss(sys)` (ss steht für State Space) berechnet MATLAB automatisch die Werte für unser System. Somit sieht die Zustandsraumdarstellung für unser System folgendermaßen aus:

$$
\begin{aligned}
\dot{x} &= \begin{bmatrix}-1\end{bmatrix} x + \begin{bmatrix}1\end{bmatrix} u\\
y &= \begin{bmatrix}1\end{bmatrix} x + \begin{bmatrix}0\end{bmatrix} u
\end{aligned}
$$

Sind die Matrizen $A, B, C, D$ bekannt, kommt man mit der folgenden Formel zur Übertragungsfunktion $G(s)$, wobei $I$ die Einheitsmatrix darstellt.

$$G(s) = C(s * I - A)^{-1} * B + D$$

Dies lässt sich auch in MATLAB eingeben, indem man man die Matrizen $A, B, C, D$ belegt und den Befehl `sys = ss(A, B, C, D)` ausführt. Von hier aus liefert der Befehl `tf(sys)` die Übertragungsfunktion.

Ohne Matlab kann die Zustandsraumdarstellung in einfachen Fällen mithilfe des Substitutionstricks aus der Übertragungsfunktion erstellt werden. Für Systeme, bei denen auch $\dot{u}, \ddot{u}, \ldots$ in der Differentialgleichung vorkommt, ist das Ganze ein bisschen komplizierter, wird aber in Wikipedia beschrieben und kann entsprechend angewendet werden. Bei sprungfähigen Systemen sieht das noch etwas anders aus und es muss erst eine Polynomdivision gemacht werden, um $D$ zu erhalten. Im Mehrgrößenfall ist es noch komplizierter.

In unserem System erhält man damit aber nicht die eigentliche Übertragungsfunktion, da die Totzeit $T_t$ verloren geht. Für die Funktionalbeziehung eines Totzeitglieds im Zeitbereich gilt: $y(t) = u(t - T_t)$. Somit ergibt sich für die Zustandsraumdarstellung eines zeitverzögerten Systems:

$$\dot{x}(t) = Ax(t) + Bu(t - T_t)$$
$$y(t) = Cx(t) + Du(t - T_t)$$

In unserem Beispiel mit $T_t = 2$ bedeutet das also:

$$\dot{x}(t) = \begin{bmatrix}-1\end{bmatrix} x(t) + \begin{bmatrix}1\end{bmatrix} u(t-2)$$
$$y(t) = \begin{bmatrix}1\end{bmatrix} x(t) + \begin{bmatrix}0\end{bmatrix} u(t-2)$$

–> HIER NOCHMAL GROELL NACHFRAGEN

**Anfangswerte**

Für die Simulation werden die Anfangswerte, da der Start der Funktion definiert sein muss. Hierbei müssen die linken Grenzwerte verwendet werden, da sonst eventuelle Sprünge miteinbezogen würden. Es gilt:

$$y(0^-) = Cx(0^-) + Du(0^-)$$
$$\dot{y}(0^-) = CAx(0^-) + CBu(0^-) + D\dot{u}(0^-)$$

Darüber hinaus gibt es einen Zusammenhang mit der Eingangs- / Ausgangs-Differentialgleichung: Die Anfangswerte des Zustandsraumes $x$ müssen mit denen der Eingangs- / Ausgangs-Differentialgleichung $y$ korrespondieren. Da wir bei unserem System davon anfangs kein Zufluss vorliegt, ist $u(0^-) = 0$. Daher lassen sich die Anfangswerte der mit folgender Formel berechnen:

$$\begin{pmatrix} y \\ \dot{y} \\ \ddot{y} \end{pmatrix} = \begin{pmatrix} C^T \\ C^T A \\ C^T A^2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$
$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} C^T \\ C^T A \\ C^T A^2 \end{pmatrix}^{-1} \begin{pmatrix} y \\ \dot{y} \\ \ddot{y} \end{pmatrix}$$

Weiterhin gehen wir bei unserem System davon aus, dass der Tank zu Beginn leer ist, somit kein Ausfluss vorliegt. Daraus ergibt sich $x(0^-) = 0$ und mit voriger Formel gilt:

$$\begin{pmatrix} y \\ \dot{y} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

### 3.2.2 Explizite Darstellung

Sind zusätzlich die Anfangswerte bekannt, so lässt sich $x(t)$ und $y(t)$ wie folgt berechnen:

$$x(t) = e^{A(t-t_0)} * x(0) * \int_0^t e^{A(t-t_0)} Bu(\tau)\, d\tau$$
$$y(t) = Ce^{A(t-t_0)} * x(0) + C \int_0^t e^{A(t-t_0)} Bu(\tau)\, d\tau + Du(t)$$

Hier ist bei der Eingabe in MATLAB wichtig zu beachten, dass für $e^{(\dots)}$ nicht `exp`, sondern `expm`, verwendet wird, da man hier die Matrizenmultiplikation benötigt.

### 3.2.3 Integralgleichung

Im Gegensatz zur Differentialgleichung ergibt die Integralgleichung „mild solution“, welche Sprünge abbilden können. Als Integralgleichung gilt:

$$x(t) = x(0) + \int_0^t Ax(\tau) + Bu(\tau)\, d\tau$$

Für unser System ergibt sich somit:

$$x(t) = \int_0^t \begin{bmatrix} -1 \end{bmatrix} x(\tau) + \begin{bmatrix} 1 \end{bmatrix} u(\tau)\, d\tau$$

### 3.2.4 Differentialgleichung

Aufgrund unseres physikalischen Modells gilt:

$$y(t) = x(t-2)$$

## 3.3 Eingangs- /Ausgangs Differentialgleichung

Für unser System ergibt sich:

$$\begin{aligned} G(s) &= \frac{1}{1+s} * e^{-2s} \\ G_{PT_1}(s) = \frac{Y(s)}{U(s)} &= \frac{1}{1+s} \\ Y(s)(1+s) &= (1) * U(s) \\ Y(s) + Y(s) * s &= 1 * U(s) \\ \dot{y}(t) + y(t) &= u(t) \end{aligned}$$

In dieser Differentialgleichung taucht die Totzeit $T_t$ nicht auf, da das Totzeitglied nicht mit einer gewöhnlichen Differentialgleichung beschrieben werden kann. Für die Funktionalbeziehung eines Totzeitglieds im Zeitbereich gilt: $y(t) = u(t - T_t)$. Für unser Beispiel mit der Totzeit $T_t = 2$ folgt somit:

$$\dot{y}(t) + y(t) = u(t-2)$$

## 3.4 Übertragungsfunktion

Formel 4 zeigt die Übertragungsfunktion die in diesem Skript analysiert wird. Formel 6 zeigt eine etwas andere Darstellung der Formel, sodass die Eigenschaften des Systems leichter abzulesen sind.

$$
\begin{aligned}
G(s) &= \frac{Y(s)}{U(s)} & (4)\\
&= \frac{1 * e^{-2s}}{(1+s)} & (5)\\
&= \frac{1}{1+s} * e^{-2s} & (6)
\end{aligned}
$$

### 3.5 Gewichtsfunktion (Impulsantwort)

In Abbildung 3.5 ist die Reaktion des Systems auf technischen Eingangssprung zu sehen. Diese Gewichtsfunktion oder auch Impulsantwort bezeichnet man oft mit $\delta(t)$.
In MATLAB lässt sich das mit dem Befehl `impulse(sys)` simulieren, wodurch man die numerische Lösung erhält. In MATLAB lässt sich das mit dem Befehl `step(sys)` simulieren, wodurch man die numerische Lösung erhält. Die Übergangsfunktionfunktion $h(t)$ lässt sich aber auch analytisch berechnen und man erhält die explizite Lösung. Hierbei hilft einem der Befehl `ilaplace()`, der die Laplace-Transformierte zurückliefert:

```
syms g(t)
g(t) = ilaplace(H(s))
t = [0:0.1:8]
plot(t, g(t))
```

Beide Befehle führen zum folgenden Graphen:

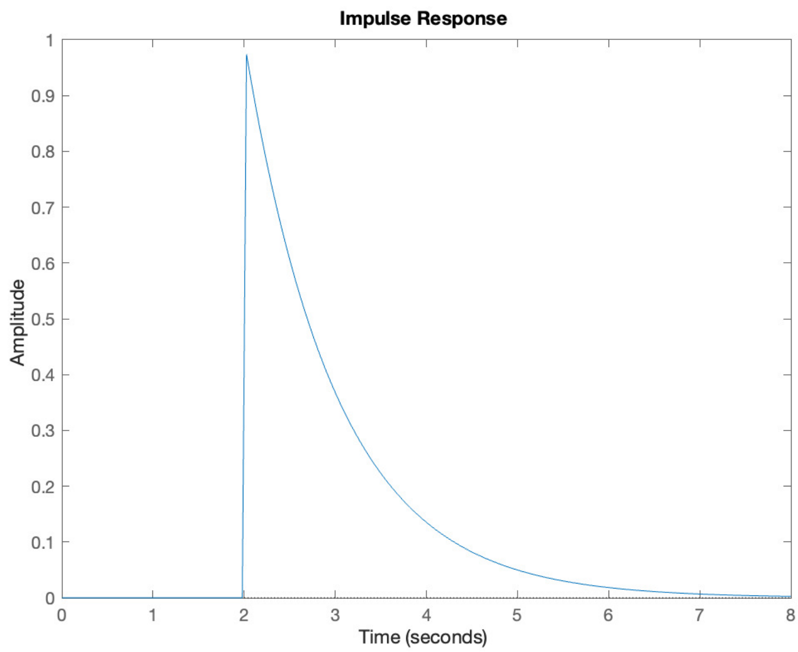


Abbildung 2: Impulsantwort

### 3.6 Übergangsfunktion (Sprungantwort)

In Abbildung 3.6 ist die Reaktion des Systems auf den Heaviside-Funktion zu sehen. Diese ist wie folgt definiert:

$$\sigma(t) = \begin{cases} 0 & \text{für t} < 0 \\ 1 & \text{für t} > 0 \end{cases}$$

Man bezeichnet diese Übergangsfunktion oder auch Sprungantwort oft mit $1(t)$.
In MATLAB lässt sich das mit dem Befehl `step(sys)` simulieren, wodurch man die numerische Lösung erhält. Die Übergangsfunktionfunktion $h(t)$ lässt sich aber auch analytisch berechnen und man erhält die explizite Lösung. Hierbei hilft einem der Befehl `ilaplace()`, der die Laplace-Transformierte zurückliefert:

```
H(s) = G(s)/s
syms h(t)
h(t) = ilaplace(H(s))
t = [0:0.1:8]
plot(t, h(t))
```

Beide Befehle führen zum folgenden Graphen:

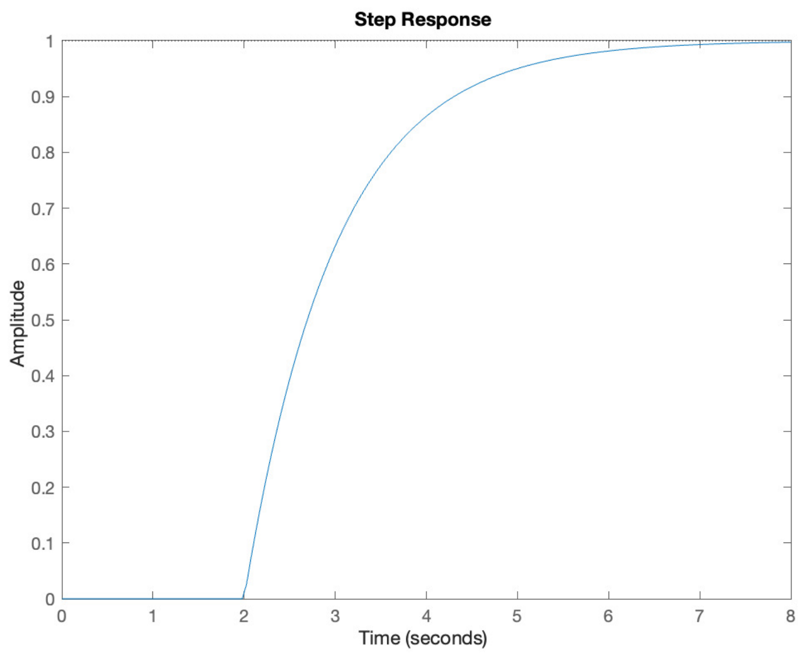


Abbildung 3: Sprungantwort

## 3.7 Frequenzgang

Zum Frequenzgang $G(j * \omega)$ gelangt man, indem man in $G(s)$ die Variable $s$ durch $j * \omega$ ersetzt.

Das bedeutet, das für jede Frequzenz $\omega$ eine komplexe Zahl $j$ mit Real- und Imaginärteil oder mit Betrag und Phase darstellt. Da in der Systemtheorie $i$ für den Strom steht, wird hier $j$ als imaginäre Einheit verwendet.

### 3.7.1 Ortskurve (Nyquist-Plot)

Im Nyquist-Plot wird der über $\omega$ parametrisierte Frequenzgang als Kurver in der komplexen Ebene mit Real- und Imaginärteil dargestellt. MATLAB bietet hier den Befehl `nyquist(sys)` an, welche zum folgenden Plot führt. Dabei zeigt MATLAB die Kurve von $-\infty$ bis $+\infty$ an, was in unserem Beispiel jedoch nicht nötig ist.

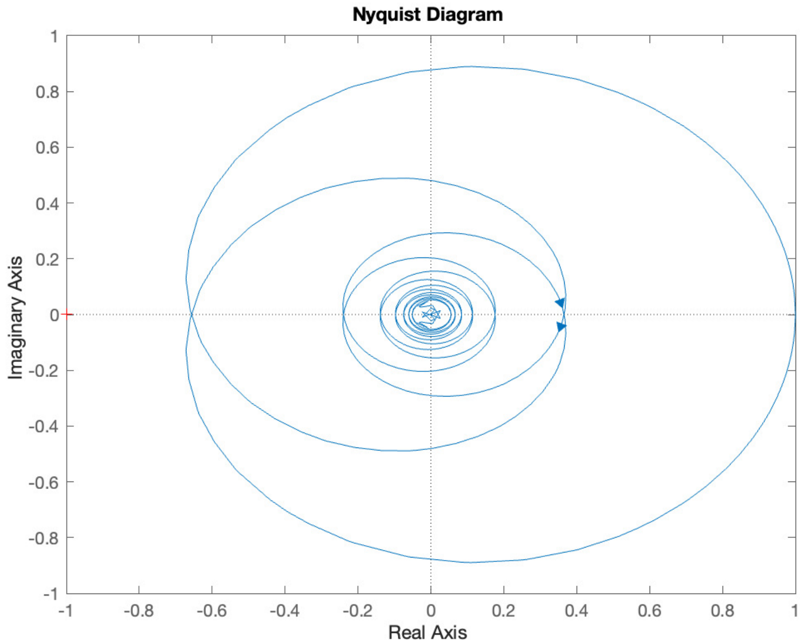


Abbildung 4: Nyquist-Plot

Darüber hinaus ist es in MATLAB möglich, mit dem Cursor entlang der Kurve zu gehen, um sich einzelne Werte genauer anzuschauen, wie in Abbildung 3.7.1 dargestellt.

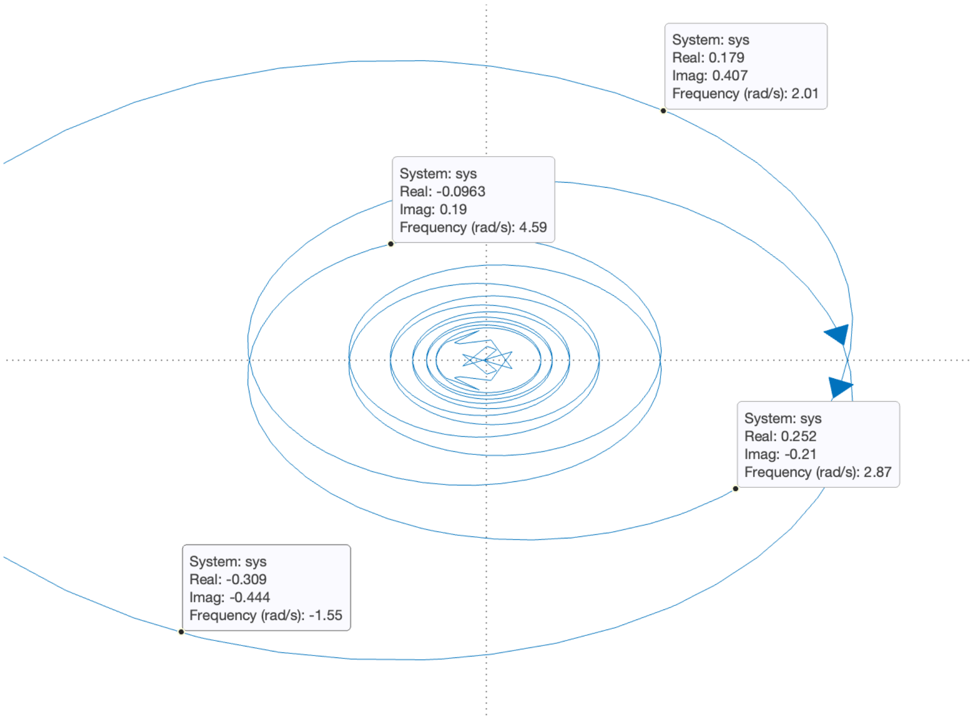


Abbildung 5: Screenshot: Informationen im Nyquist-Plot

Systemtheoretisch entspricht der Frequenzgang einem Schnitt der komplexen Funktion $G(s)$ entlang der imaginären Achse: $G(s)\Big|_{s=j*\omega}$

### 3.7.2 Bode-Diagramm

Im Bode-Diagramm sind der Betrag des Frequenzgangs über $\omega$ in Dezibel(dB) und die Phase des Frequenzgangs über $\omega$ in Grad($^\circ$) dargestellt. Damit die Kurven schön aussehen, verwendet MATLAB eine doppelt logarithmische Darstellung. Der dazugehörige MATLAB-Befehl ist `bode(sys)`. Streng genommen wird in MATLAB im oberen Bild die Amplitudenverstärkung angezeigt, die ein Sinus-Signal stationär erfährt (bei Eigenwert $= 1$), währenddessen im unteren Bild die Phasenverstärkung zu sehen ist.

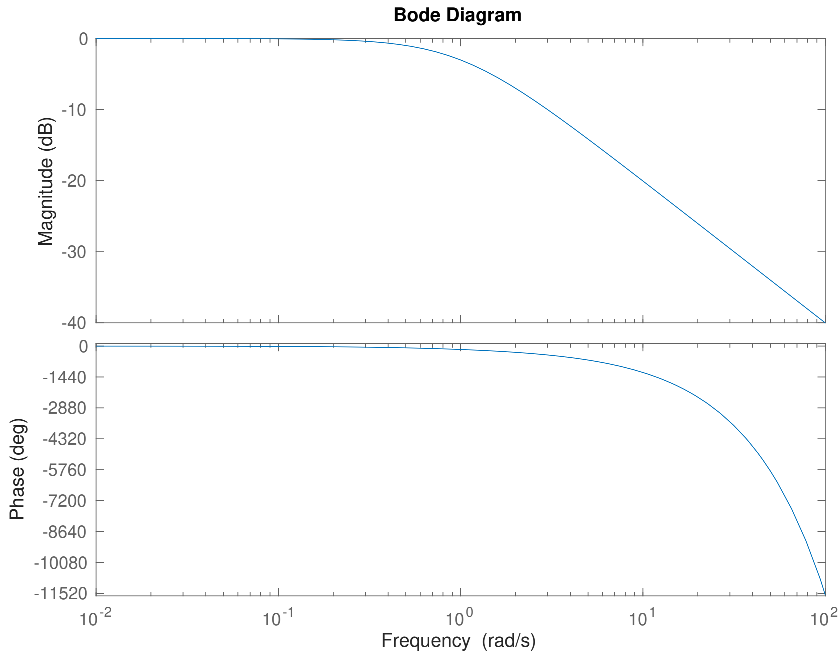


Abbildung 6: Bode-Plot

## 3.8 Statische Kennlinie

Da es sich bei unserem System um ein System mit proportionalen Übertragungsverhalten handelt, entspricht die statische Verstärkung $K = G(0)$ bzw. $K = \lim_{t\to\infty} h(t)$. Somit ergibt sich bei unserem System eine Verstärkung von 3, weshalb die statische Kennlinie nur eine konstante Funktion ist. Mithilfe des MATLAB-Befehls `fplot(1)` (wegen $K = 1$) lässt sich das graphisch darstellen. Der Nachteil der statischen Kennlinie ist, dass sie im Gegensatz zum Pol-Nullstellen-Plot keine Dynamik-Informationen enthält.

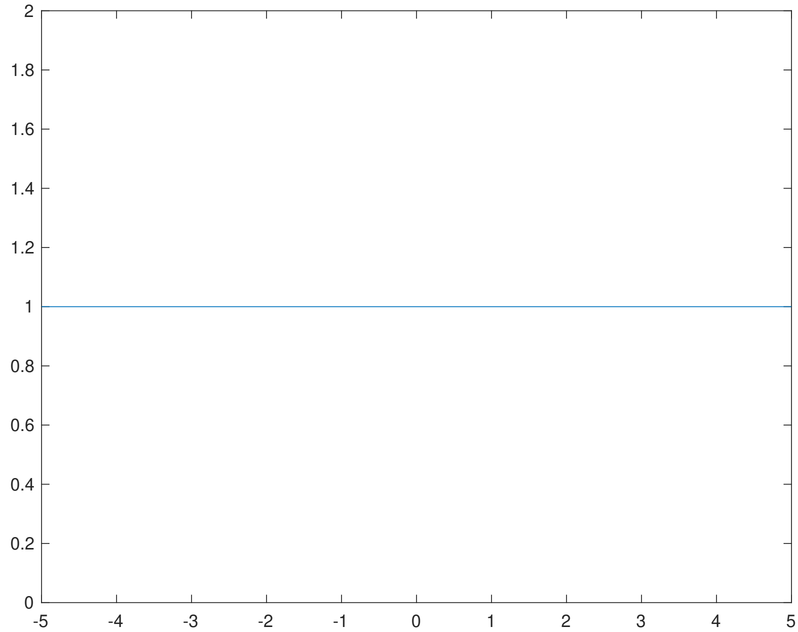


Abbildung 7: Statische Kennlinie

## 3.9 Pol-Nulstellen-Plot

Abbildung 3.9 zeigt den Pol-Nulstellen-Plot zu unserem System. Der passende MATLAB-Befehl hierfür lautet `pzplot(sys)`. MATLAB markiert mit **o** die Nullstellen und mit **x** die Polstellen des Systems. In unserem Beispiel hat das System aber keine Nulstellen, da der Zählergrad 0 ist, weshalb in der Abbildung kein **o** zu sehen ist. Da der Nenner unseres Systems aber den Grad 1 mit $1 + s$ ist, hat unserer System eine reale Polstelle bei s = -1, was in der Abbildung mit den **x** gekennzeichnet ist. Der Nachteil des Pol-Nullstellen-Plots ist, dass er im Gegensatz zur statischen kennlinie keine Statik-Informationen enthält.

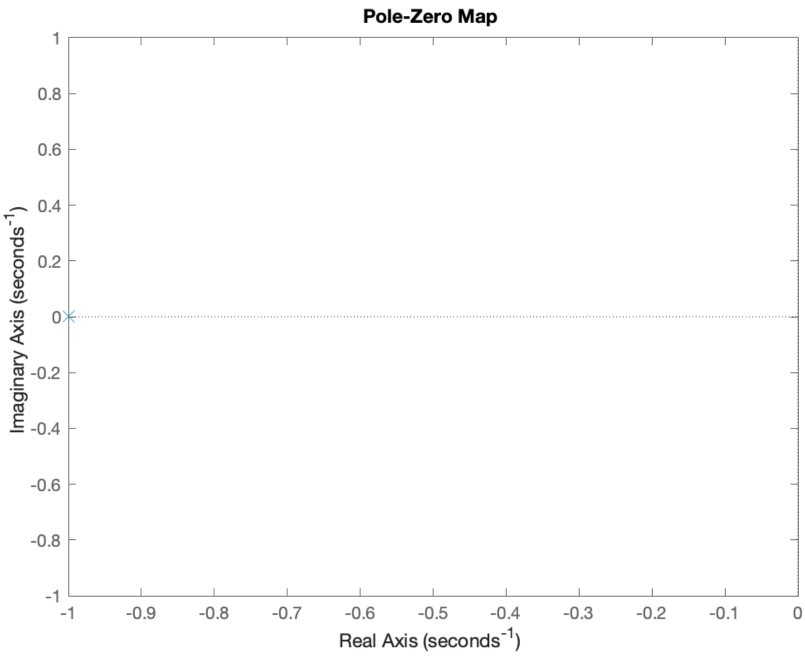


Abbildung 8: Pol-Nulstellen-Plot