

# Inversion dynamischer Systeme mit Matlab

Jörg J. Buchholz (Hochschule Bremen)  
Wolfgang v. Grünhagen (DLR Braunschweig)

17. Dezember 2007

## 1 Einleitung

Eine Aufgabe im Rahmen des US/German Memorandum of Understanding (Helicopter Aeromechanics) [1] beinhaltet die gemeinsame und komplementäre Entwicklung von Turbulenzmodellen für Hubschrauber und deren Anwendung in der Simulation und Regelung. Das generelle Prinzip ist in Abb. 1 dargestellt, der Realisierungsansatz der US Army ist in [2] und [3] beschrieben. Die vorliegende Abhandlung beschreibt die alternative Realisierung auf der Basis von MATLAB/SIMULINK, die von den deutschen Partnern – Hochschule Bremen und DLR – eingebracht wurde. Beiden Ansätzen ist der Vergleich der Hubschrauberdynamik in ungestörter Atmosphäre zu der bei unterschiedlichen Turbulenzbedingungen gemeinsam.

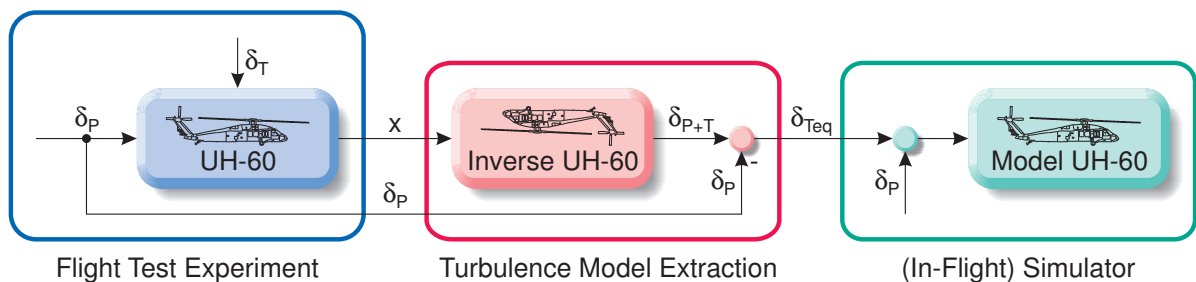


Abbildung 1: Extraktion des Turbulenzmodells

Dazu stabilisiert ein Pilot einen Hubschrauber in turbulenter ( $\delta_T$ ) Atmosphäre (beispielsweise durch Hovern auf der Leeseite eines hohen Gebäudes), wobei sowohl die Pilotensteureingabe ( $\delta_P$ ) als auch die Reaktion ( $x$ ) des Hubschraubers gemessen wird. In der anschließenden Offline-Rechnung wird  $x$  durch ein *inverses Modell* des Hubschraubers geschickt, so dass man die  $x$  reproduzierende Gesamtsteuereingabe  $\delta_{P+T}$  erhält. Durch Subtraktion der gemessenen Pilotensteuereingabe  $\delta_P$  verbleibt die turbulenzäquivalente Steuereingabe  $\delta_{Teq}$ , die in einem Simulator zur Turbulenzsimulation ohne expliziten Turbulenzeingang direkt auf die reale Pilotensteuereingabe gemischt werden kann. Von zentraler Bedeutung ist dabei die exakte und implementierbare Inversion des dynamischen Hubschraubermodelles. Die dazu verwendeten Verfahren werden im Folgenden diskutiert.

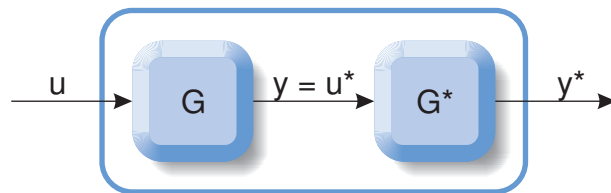


Abbildung 2: Inversion eines allgemeinen dynamischen Systems

Abb. 2 zeigt die Inversion eines allgemeinen dynamischen Systems. Dabei sind  $u$  und  $y$  die Ein- bzw. Ausgangsgröße des zu invertierenden Systems  $G$ , während  $u^*$  und  $y^*$  die Ein- bzw. Ausgangsgröße des inversen Systems  $G^*$  bezeichnen. Wenn  $G^*$  eine exakte dynamische Inverse von  $G$  ist, dann ergibt eine Reihenschaltung beider Systeme

$$u^* = y \quad (1)$$

ein System, in dem der Gesamteingang und -ausgang identisch sind

$$y^* = u. \quad (2)$$

## 2 Inversion einer Übertragungsfunktion

Wenn das zu invertierende System linear und zeitinvariant ist (LTI), keine Totzeiten und nur eine Eingangsgröße und eine Ausgangsgröße besitzt (SISO), lässt es sich als gebrochen rationale Übertragungsfunktion darstellen

$$G(s) = \frac{b_m s^m + \dots + b_1 s + b_0}{s^n + \dots + a_1 s + a_0} = \frac{Z(s)}{N(s)}. \quad (3)$$

Die Inversion geschieht dann einfach durch Vertauschen von Zähler und Nenner

$$G^*(s) = \frac{1}{G(s)} = \frac{N(s)}{Z(s)}. \quad (4)$$

Die Pole des Originalsystems werden daher zu den Nullstellen des invertierten Systems und umgekehrt. MATLAB führt die Inversion einer Übertragungsfunktion mit dem überladenen `inv`-Befehl aus

```
>> G_tf = tf ([0.1 1], [1 1])
```

Transfer function:

```
0.1 s + 1
-----
s + 1
```

```
>> G_star_tf = inv (G_tf)
```

Transfer function:

```
s + 1
-----
0.1 s + 1
```

SIMULINK kann jetzt das in Abb. 3 dargestellte Gesamtsystem simulieren.

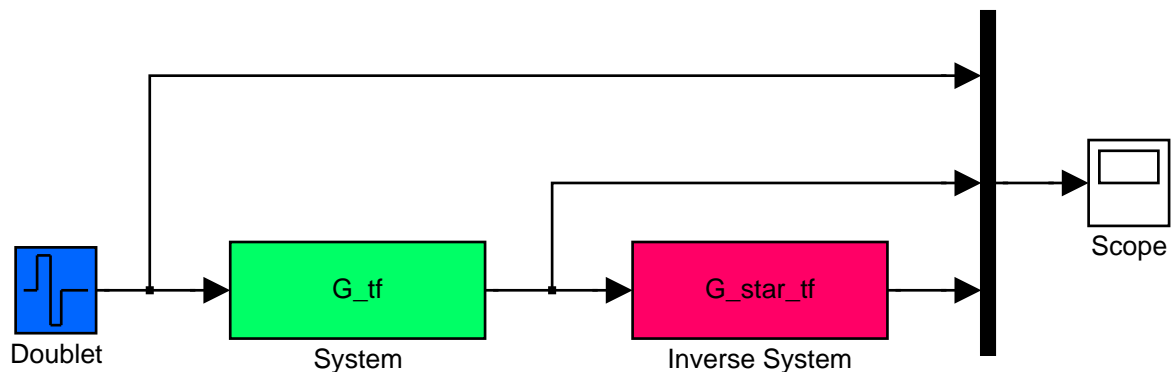


Abbildung 3: Blockschaltbild der Inversion einer Übertragungsfunktion

In Abb. 4 wird deutlich, dass die Eingangsdoulette, die ja mit ihren hohen Frequenzanteilen durchaus eine Herausforderung für den Inverter darstellt, schon mit SIMULINKs Standardsimulationsparametern perfekt wieder hergestellt wird.

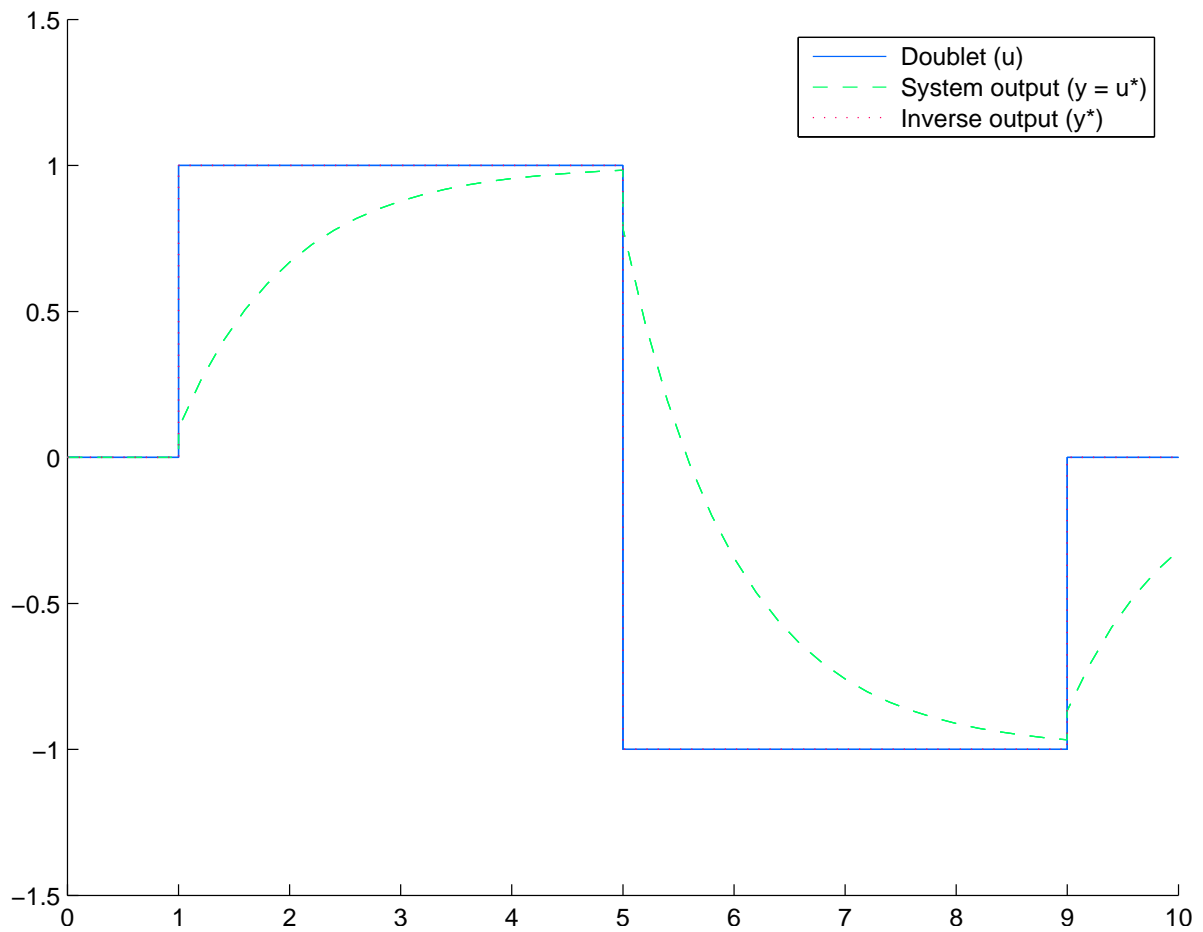


Abbildung 4: Ergebnis der Inversion einer Übertragungsfunktion

### 3 Inversion im Zustandsraum

Die Inversion eines dynamischen Systems kann natürlich auch im Zustandsraum durchgeführt werden. Die übliche Zustandsraumdarstellung eines LTI-Systems lautet

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \quad (5)$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}. \quad (6)$$

Wenn die Durchgangsmatrix  $\mathbf{D}$  regulär ist, kann Gl. 6 nach dem Eingangsvektor  $\mathbf{u}$  aufgelöst werden

$$\begin{aligned} \mathbf{u} &= \mathbf{D}^{-1}(\mathbf{y} - \mathbf{C}\mathbf{x}) \\ &= -\mathbf{D}^{-1}\mathbf{C}\mathbf{x} + \mathbf{D}^{-1}\mathbf{y}. \end{aligned} \quad (7)$$

Eingesetzt in Gl. 5 ergibt sich dann

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}(-\mathbf{D}^{-1}\mathbf{C}\mathbf{x} + \mathbf{D}^{-1}\mathbf{y}) \\ &= (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})\mathbf{x} + \mathbf{B}\mathbf{D}^{-1}\mathbf{y}.\end{aligned}\tag{8}$$

Für das inverse System müssen gemäß Gl. 1 – 2 in Gl. 7 – 8 die Eingangs- bzw. Ausgangsgrößen vertauscht werden

$$\dot{\mathbf{x}}^* = (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})\mathbf{x}^* + \mathbf{B}\mathbf{D}^{-1}\mathbf{u}^*\tag{9}$$

$$\mathbf{y}^* = -\mathbf{D}^{-1}\mathbf{C}\mathbf{x}^* + \mathbf{D}^{-1}\mathbf{u}^*.\tag{10}$$

In Gl. 9 – 10 lassen sich die Matrizen des inversen Systems direkt ablesen

$$\mathbf{D}^* = \mathbf{D}^{-1}\tag{11}$$

$$\mathbf{C}^* = -\mathbf{D}^{-1}\mathbf{C} = -\mathbf{D}^*\mathbf{C}\tag{12}$$

$$\mathbf{B}^* = \mathbf{B}\mathbf{D}^{-1} = \mathbf{B}\mathbf{D}^*\tag{13}$$

$$\mathbf{A}^* = \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C} = \mathbf{A} - \mathbf{B}^*\mathbf{C} = \mathbf{A} + \mathbf{B}\mathbf{C}^*.\tag{14}$$

MATLAB verwendet Gl. 11 – 14 bei der Anwendung des `inv`-Befehls auf ein System in Zustandsraumdarstellung

```
>> G_ss = ss (G_tf);
>> [A, B, C, D] = ssdata (G_ss)

A =
    -1
B =
     1
C =
    0.9000
D =
    0.1000

>> G_star_ss = inv (G_ss);
>> [A_star, B_star, C_star, D_star] = ssdata (G_star_ss)

A_star =
   -10
B_star =
     10
C_star =
    -9
D_star =
     10
```

Das Simulationsergebnis ist mit der in Abb. 4 dargestellten Inversion der Übertragungsfunktion identisch.

## 4 Improper Inversion

Da bei der gerade verwendeten Beispielübertragungsfunktion ( $\frac{0.1s+1}{s+1}$ ) Zähler- und Nennergrad gleich sind, ist diese, genau wie ihre Inverse, *biproper*, also *proper* (Nenner- größer oder gleich Zählergrad) aber nicht *strictly proper* (Nenner- größer Zählergrad). Beide Systeme können daher problemlos dargestellt und simuliert werden.

Leider aber sind viele technische Systeme auf Grund ihres Tiefpasscharakters *strictly proper*, so dass ihre Inversen *improper* (Zähler- größer Nennergrad) sind und sich daher zwar berechnen, aber weder in den Zustandsraum transformieren noch direkt simulieren lassen.

```
>> G_tf = tf (1, [1 1])
```

```
Transfer function:
```

```
  1
-----
s + 1
```

```
>> G_star_tf = inv (G_tf)
```

```
Transfer function: s + 1
```

```
>> G_star_ss = ss (G_star_tf)
```

```
??? Error using ==> tf/ss Improper system. Conversion to state-space is not possible
```

```
>> step (G_star_tf)
```

```
??? Error using ==> rfinputs Not supported for non-proper models.
```

Jetzt wäre es natürlich möglich, das inverse System unter SIMULINK mit einem *Derivative*-, einem *Gain*- und einem *Sum*-Block zu modellieren; üblicherweise wird die Übertragungsfunktion allerdings einfach durch so viele zusätzliche hochfrequente Pole ergänzt, bis sie proper ist.

```
>> G_filt = tf (1, [1e-3 1])
```

```
Transfer function:
```

```
  1
```

```

-----
0.001 s + 1

>> G_prop_star_tf = G_star_tf*G_filt

Transfer function:
   s + 1
-----
0.001 s + 1

```

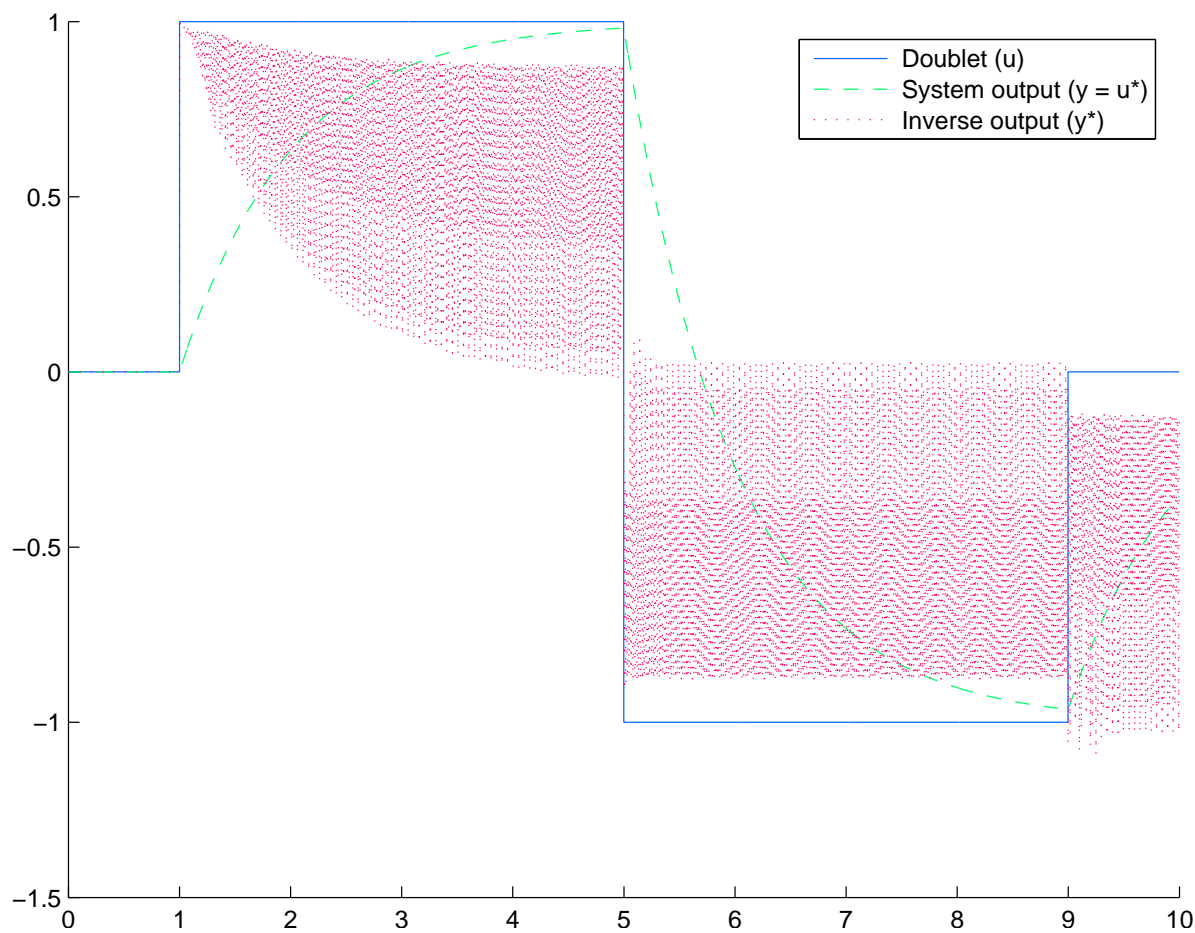


Abbildung 5: Instabilität verursacht durch nicht angepasstes Integrationsverfahren

Abb. 5 zeigt das eindrucksvolle Resultat des Versuches, das Gesamtsystem (gemäß Abb. 3) mit SIMULINKs Standardparametern (*Solver type: Variable-step ode45, Relative tolerance: 1e-3*) zu simulieren. Offensichtlich gelingt es dem Dormand-Prince-Integrationsverfahren nicht, die Schrittweite für das moderat steife System (Zeitkonstantenverhältnis 1000) korrekt anzupassen. Bei der Wahl der Frequenz der zusätzlichen

Pole muss ein Kompromiss gefunden werden, da die Pole auf der einen Seite schnell genug sein sollen, um keine zusätzliche signifikante parasitäre Dynamik einzuführen; auf der anderen Seite erhöhen Pole, die sehr viel schneller als das eigentliche System sind, die Bandbreite des Gesamtsystem, machen es steifer und damit für Integrationsverfahren, die nicht für steife Systeme vorgesehen sind, schwerer zu integrieren.

Bei Verwendung eines Verfahrens mit hinreichend kleiner fester Schrittweite (*Solver type: Fixed-step ode4, Fixed-step size: 0.001*) oder eines für steife Systeme ausgelegten Verfahrens (*Solver type: Variable-step ode15s*) sieht das Simulationsergebnis dann wieder so unspektakulär wie in Abb. 4 aus. Dabei geht die Wirkung der hochfrequenten Pole in der Strichstärke unter.

## 5 Proper Inversion

Das in Abb. 6 dargestellte Inversionsverfahren lässt sich auch auf MIMO (Multi Input Multi Output) und auf nichtlineare Systeme anwenden. Dazu wird das zu invertierende System in den Rückführzweig eines Regelkreises mit sehr hoher Reglerverstärkung (z.B.  $K = 999$ ) eingesetzt und die Eingangsgröße des Systems als Ausgangsgröße des inversen Systems aufgefasst.

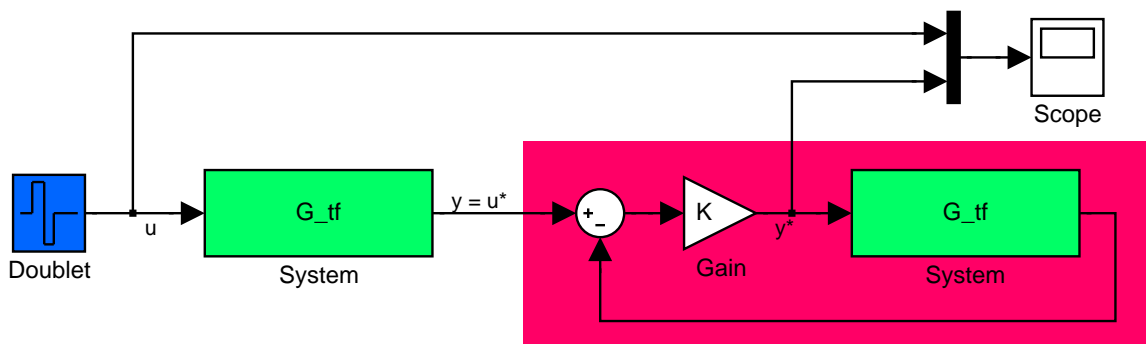


Abbildung 6: Blockschaltbild der Proper Inversion

Die Übertragungsfunktion des Inverters lautet dann

$$\begin{aligned}
 G^*(s) &= \frac{y^*}{u^*} = \frac{K}{1 + K \cdot G(s)} \\
 &= \frac{K}{1 + K \cdot \frac{Z(s)}{N(s)}} = \frac{K \cdot N(s)}{N(s) + K \cdot Z(s)}.
 \end{aligned} \tag{15}$$



Es wird deutlich, dass die Nullstellen von  $G^*$  den Polen von  $G$  entsprechen (unabhängig von der Größe von  $K$ ) und dass die Pole von  $K$  abhängen: Für kleine  $K$  „beginnen“ die Pole von  $G^*$  in den Polen von  $G$ ; wenn  $K$  wächst, nähern sich die Pole von  $G^*$  den Nullstellen von  $G$

$$\lim_{K \rightarrow \infty} G^*(s) = \frac{N(s)}{Z(s)}. \quad (16)$$

Gl. 16 ist - streng genommen - ein wenig irreführend, da es so aussieht, als ob  $G^*$  so viele Pole hat wie  $G$  Nullstellen, was aber natürlich nicht der Fall ist. Die Anzahl der Pole eines Systems ändert sich durch die Rückführung nicht. Wenn  $G$  proper ist ( $n = \text{Anzahl der Pole} \geq m = \text{Anzahl der Nullstellen}$ ) muss es daher  $n - m$  Pole geben, die, wenn  $K$  wächst, nicht in die Nullstellen, sondern nach unendlich laufen. Diese zusätzlichen Pole des inversen Systems machen das System automatisch proper und damit implementierbar. Ihre Position kann direkt durch  $K$  gesteuert werden. Für das Beispiel aus Abschnitt 4 ergibt sich

$$\begin{aligned} G^*(s) &= \frac{K}{1 + K \cdot G(s)} = \frac{K}{1 + K \frac{1}{s+1}} \\ &= \frac{K(s+1)}{s+1+K} = \frac{\frac{K}{1+K}(s+1)}{\frac{s}{1+K} + 1}. \end{aligned} \quad (17)$$

Wenn jetzt  $K$  beispielsweise auf 999 gesetzt wird, hat das inverse System (verglichen mit Abschnitt 4)

$$G^*(s) = \frac{999(s+1)}{s+1000} = \frac{0.999(s+1)}{0.001s+1} \quad (18)$$

die gleiche Nullstelle ( $-1$ ), den gleichen hochfrequenten Pol ( $-1000$ ) und eine marginal unterschiedliche stationäre Verstärkung (0.999 statt 1).

Wenn eine exakte stationäre Verstärkung wichtig ist, kann ein statisches Vorfilter  $G_P$  mit

$$G_P = \frac{1}{G(0) \cdot G^*(0)} = \frac{1 + K \cdot G(0)}{K \cdot G(0)} \quad (19)$$

mit dem Inverter in Reihe geschaltet werden.

Um die in Abb. 7 dargestellten Details korrekt zu simulieren, wurde das (für steife Systeme geeignete) ode15s-Integrationsverfahren mit einer relativen Toleranzforderung von  $1e-6$  verwendet.

Wieder muss ein Kompromiss gefunden werden, da hohe Verstärkungsfaktoren einerseits zwar den stationären Fehler und den Einfluss der hochfrequenten Pole verringern, andererseits aber die Bandbreite der Inversen vergrößern und damit dem Integrationsalgorithmus das Leben schwer machen.

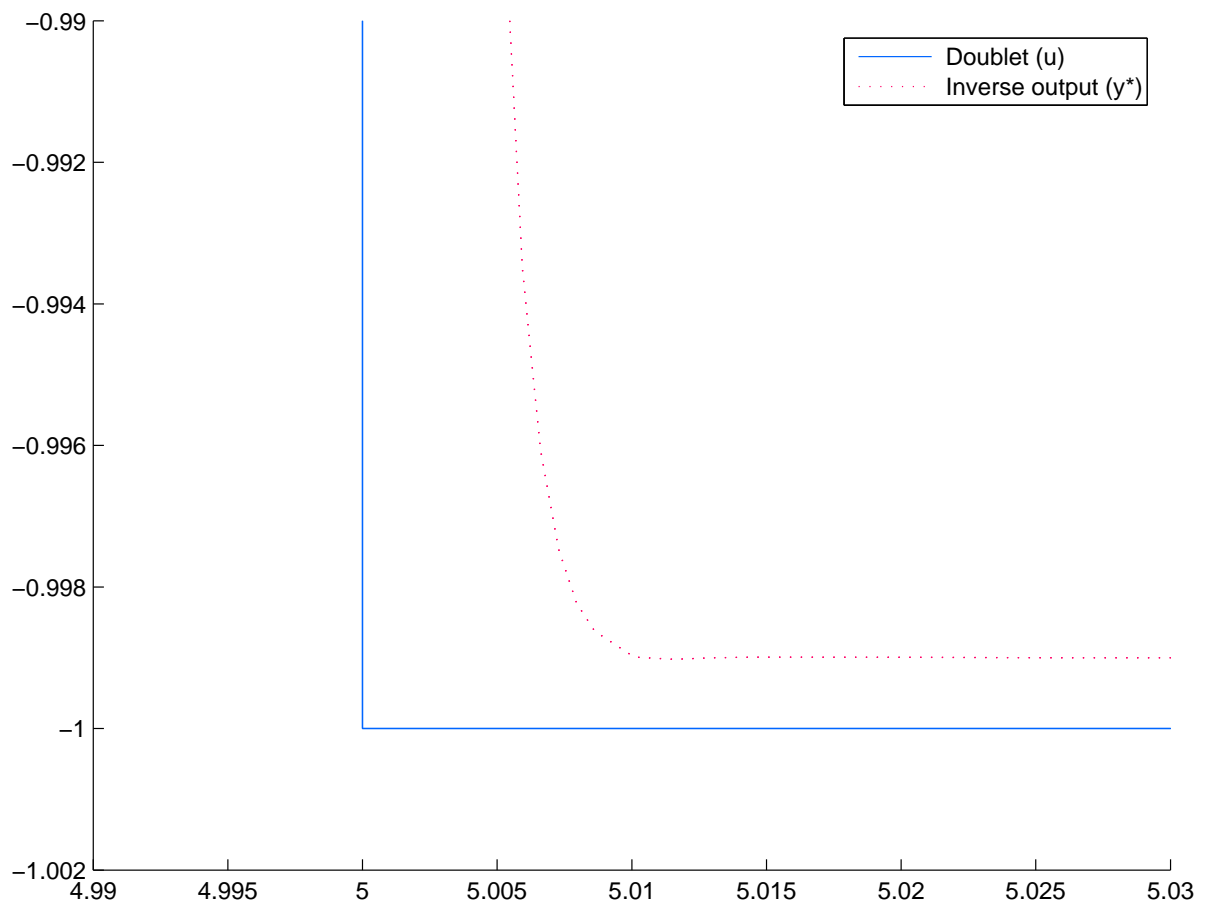


Abbildung 7: Ergebnis der Proper Inversion

## 6 Mehrgrößensysteme

Wenn das zu invertierende System mehr als eine Eingangs- bzw. Ausgangsgröße besitzt, wird seine Inversion deutlich komplizierter.

## 6.1 Beispiel

Das folgende quadratische System mit zwei Eingangs-, zwei Ausgangs- und drei Zustandsgrößen soll in diesem Abschnitt als MIMO-Beispiel dienen

$$\begin{bmatrix} \mathbf{A} & \vdots & \mathbf{B} \\ \dots & \cdot & \dots \\ \mathbf{C} & \vdots & \mathbf{D} \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & \vdots & 1 & 0 \\ 0 & -1 & 1 & \vdots & 1 & 0 \\ -1 & 0 & -1 & \vdots & 0 & 1 \\ \dots & \dots & \dots & \cdot & \dots & \dots \\ 0 & 1 & 0 & \vdots & 0 & 0 \\ 0 & 0 & 1 & \vdots & 0 & 0 \end{bmatrix}. \quad (20)$$

MATLAB kann die Pole und (invarianten) Nullstellen des Systems berechnen

```
>> A = [-1, 1, 0; 0, -1, 1; -1, 0, -1];
>> B = [1, 0; 1, 0; 0, 1];
>> C = [0, 1, 0; 0, 0, 1];
>> D = zeros (2,2);
>> G_ss = ss (A, B, C, D);
```

```
>> zero (G_ss)
```

```
ans =
```

```
-1
```

```
>> pole (G_ss)
```

```
ans =
```

```
-2.0000
```

```
-0.5000 + 0.8660i
```

```
-0.5000 - 0.8660i
```

Alle Pole und Nullstellen liegen in der linken komplexen Halbebene, so dass sowohl das System als auch seine Inverse stabil sind.

## 6.2 Numerische Inversion

Der Versuch, das System numerisch via `inv`-Befehl zu invertieren, führt zu einer Fehlermeldung

```
>> G_star_ss = inv (G_ss)
?? Error using ==> ss/inv Cannot invert system with singular D
matrix.
```

Wie schon erwähnt, verwendet MATLAB Gl. 11 – 14, um ein in Zustandsraumdarstellung gegebenes System zu invertieren. Gl. 11 versucht dabei, die Durchgangsmatrix  $D$  zu invertieren, was im Beispiel bei einer Nullmatrix natürlich nicht gelingt.

### 6.3 Analytische Inversion

Glücklicherweise bietet MATLABs *Symbolic Math Toolbox* einen sehr eleganten Weg, das System trotzdem zu invertieren. Dazu definiert man eine symbolische Laplace-Variable und verwendet direkt die aus der Laplace-Transformation der Zustandsgleichungen entstandene allgemeine Übertragungsmatrix

$$s\mathbf{X}(s) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s) \quad (21)$$

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) \quad (22)$$

$$\begin{aligned} \mathbf{Y}(s) &= \mathbf{C}\mathbf{X}(s) + \mathbf{D}\mathbf{U}(s) \\ &= \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) + \mathbf{D}\mathbf{U}(s) \end{aligned} \quad (23)$$

$$\mathbf{G}(s) = \frac{\mathbf{Y}(s)}{\mathbf{U}(s)} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}. \quad (24)$$

```
>> syms s
>> G_sym = simple (C*inv (s*eye (3, 3) - A)*B + D);
>> pretty (G_sym)
```

```
[      s          s + 1      ]
[ -----  ----- ]
[  2          2          ]
[ s + s + 1  (s + 2) (s + s + 1)]
[                                     ]
[                                     ]
[                                     ]
[          1          (s + 1)      ]
[ -----  ----- ]
[  2          2          ]
[ s + s + 1  (s + 2) (s + s + 1)]
```

Die analytische Inversion kann dann direkt mit dem symbolischen `inv`-Befehl erfolgen

```
>> G_star_sym = inv (G_sym);
>> pretty (G_star_sym)
```

$$\begin{bmatrix} s + 1 & -1 \\ s + 2 & s (s + 2) \\ \hline s + 1 & s + 1 \end{bmatrix}$$

Zur Kontrolle kann gezeigt werden, dass das Produkt beider Matrizen die Einheitsmatrix ergibt.

```
>> ident = simple (G_sym*G_star_sym)
```

```
ident =
```

$$\begin{bmatrix} 1, 0 \\ 0, 1 \end{bmatrix}$$

Wie erwartet, sind einige der Übertragungsfunktionen von `G_star_sym` nicht proper; die Inverse kann daher nicht direkt implementiert werden. Abhilfe schafft auch hier wieder ein schneller Tiefpass, der symbolisch definiert und mit dem System multipliziert wird.

```
>> T_filt_sym = 1e-3;
>> G_filt_sym = 1/(T_filt_sym*s + 1);
>> G_star_filt_sym = G_star_sym*G_filt_sym;
>> pretty (G_star_filt_sym)
```

$$\begin{bmatrix} s + 1 & 1 \\ \hline 1/1000 s + 1 & 1/1000 s + 1 \\ s + 2 & s (s + 2) \\ \hline (1/1000 s + 1) (s + 1) & (1/1000 s + 1) (s + 1) \end{bmatrix}$$

Die in [6] zu findende `sym2tf`-Funktion wandelt die symbolische Übertragungsmatrix wieder in ihr numerisches Äquivalent, das sich direkt implementieren lässt.

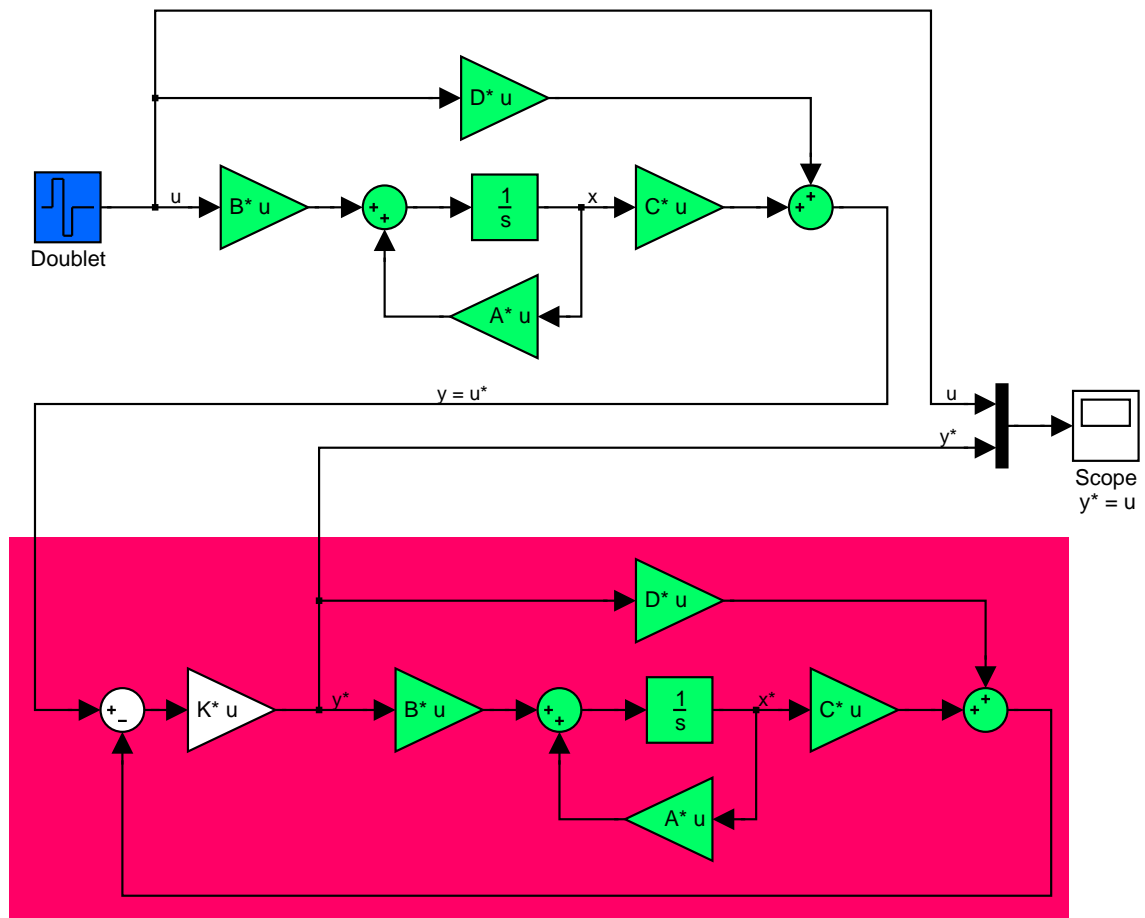


Abbildung 8: Blockschaltbild der proper Inversion im Zustandsraum

## 6.4 Proper MIMO Inversion

Die im Abschnitt 5 beschriebene Inversionsmethode kann, wie in Abb. 8 dargestellt, auch im Mehrgrößenfall verwendet werden, um die Inverse ohne weiteren Berechnungsaufwand zu simulieren.

Die algebraische Schleife, bestehend aus  $\mathbf{D}$ ,  $\mathbf{K}$  und den beiden äußeren Additionsblöcken in Abb. 8, kann aufgebrochen werden

$$\mathbf{y}^* = \mathbf{K}(\mathbf{u}^* - (\mathbf{C}\mathbf{x}^* + \mathbf{D}\mathbf{y}^*)). \quad (25)$$

Dies führt zu der inversen Ausgangsgleichung

$$\begin{aligned} (\mathbf{I} + \mathbf{K}\mathbf{D})\mathbf{y}^* &= \mathbf{K}\mathbf{u}^* - \mathbf{K}\mathbf{C}\mathbf{x}^* \\ \mathbf{y}^* &= -(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1}\mathbf{K}\mathbf{C}\mathbf{x}^* + (\mathbf{I} + \mathbf{K}\mathbf{D})^{-1}\mathbf{K}\mathbf{u}^*, \end{aligned} \quad (26)$$

der Zustandsgleichung

$$\begin{aligned}
\dot{\mathbf{x}}^* &= \mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{y}^* \\
&= \mathbf{A}\mathbf{x}^* + \mathbf{B} \left( -(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K}\mathbf{C}\mathbf{x}^* + (\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K}\mathbf{u}^* \right) \\
&= (\mathbf{A} - \mathbf{B}(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K}\mathbf{C}) \mathbf{x}^* + \mathbf{B}(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K}\mathbf{u}^*
\end{aligned} \tag{27}$$

und zum inversen System

$$\begin{bmatrix} \mathbf{A}^* & \mathbf{B}^* \\ \mathbf{C}^* & \mathbf{D}^* \end{bmatrix} = \begin{bmatrix} (\mathbf{A} - \mathbf{B}(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K}\mathbf{C}) & \mathbf{B}(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K} \\ -(\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K}\mathbf{C} & (\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K} \end{bmatrix}. \tag{28}$$

Zwei Eigenschaften dieses inversen Systems sollen hier erwähnt werden:

1. Wenn  $\mathbf{D}$  vorhanden ist und  $\mathbf{K}$  gegen unendlich läuft (wenn also wenigstens die Diagonalelemente von  $\mathbf{K}\mathbf{D}$  sehr viel größer als eins sind) kann die Einheitsmatrix vernachlässigt werden,  $\mathbf{K}$  kann „herausgekürzt“ werden und die Matrixdefinitionen entsprechen denen von Gl. 11 – 14; beispielsweise

$$\begin{aligned}
\lim_{\mathbf{K} \rightarrow \infty} \mathbf{D}^* &= \lim_{\mathbf{K} \rightarrow \infty} (\mathbf{I} + \mathbf{K}\mathbf{D})^{-1} \mathbf{K} \\
&= \lim_{\mathbf{K} \rightarrow \infty} (\mathbf{K}\mathbf{D})^{-1} \mathbf{K} = \lim_{\mathbf{K} \rightarrow \infty} \mathbf{D}^{-1} \mathbf{K}^{-1} \mathbf{K} = \mathbf{D}^{-1}.
\end{aligned} \tag{29}$$

2. Wenn das System strictly proper ist ( $\mathbf{D} = \mathbf{0}$ ), vereinfacht sich das inverse System zu

$$\begin{bmatrix} \mathbf{A}^* & \mathbf{B}^* \\ \mathbf{C}^* & \mathbf{D}^* \end{bmatrix} = \begin{bmatrix} (\mathbf{A} - \mathbf{B}\mathbf{K}\mathbf{C}) & \mathbf{B}\mathbf{K} \\ -\mathbf{K}\mathbf{C} & \mathbf{K} \end{bmatrix}. \tag{30}$$

Im Vergleich mit Gl. 11 – 14 sieht man, dass  $\mathbf{K}$  die Funktion von  $\mathbf{D}^{-1}$  übernommen hat; Große  $\mathbf{K}$ -Matrizen haben den gleichen „propernden“ Effekt wie die künstliche Addition kleiner Werte zur  $\mathbf{D}$ -Matrix.

## 7 Inversion eines nichtlinearen Systems

Die analytische Inversion nichtlinearer dynamischer Systeme ist im Allgemeinen nicht möglich. Schon die Lösung eines scheinbar einfachen Problems wie des angeregten mathematischen Pendels

$$\ddot{y} + \sin(y) = u, \tag{31}$$

wobei  $y$  der Auslenkungswinkel und  $u$  der normierte Schub ist, führt über elliptische Integrale zu Jacobischen Funktionen [4], die nicht gerade das tägliche Brot eines Ingenieurs darstellen.

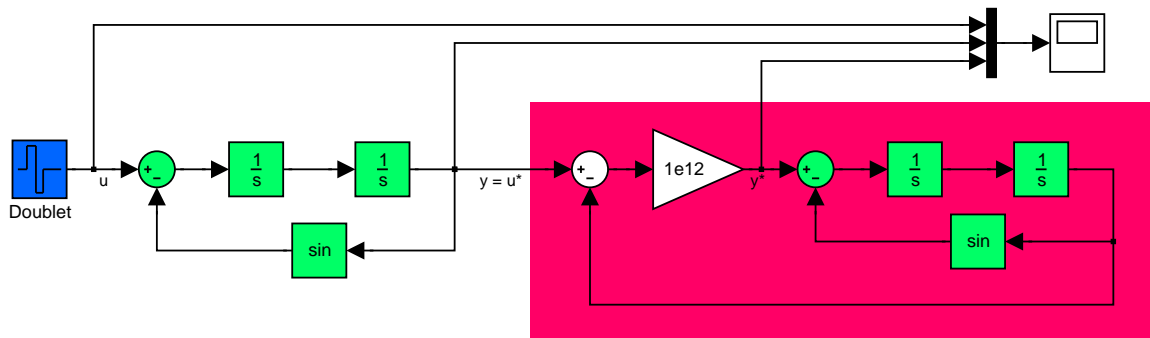


Abbildung 9: Blockschaltbild der Inversion eines nichtlinearen Systems

Auf der anderen Seite lässt sich die Simulation der Inversen des nichtlinearen Pendels nach der im Abschnitt 5 beschriebenen Methode problemlos durchführen. Die in Abb. 9 verwendete extrem hohe Verstärkung von  $1e12$  erreicht eine in Abb. 10 dargestellte nahezu perfekte Rekonstruktion der Eingangsdoublette. Dabei wurde der ode15s-Integrationsalgorithmus verwendet und alle seine Parameter (*max step size*, ..., *absolute tolerance*) auf *auto* gesetzt.

## 8 Anwendungen

Im Rahmen des MoU wurden zur Turbulenzmodell-Identifikation beide Inversionsverfahren („Analytische Inversion“ nach Abschnitt 6.3 und „Proper Inversion“ nach Abschnitt 6.4) erfolgreich auf drei verschiedene lineare Helikoptermodelle (UH-60 der NASA in Ames und Bo-105 und EC-135 des DLR in Braunschweig) und reale Flugversuchsdaten angewandt [3] und [6]. Bei der UH-60 wurde ein Modell mit 41 Zustandsgrößen (inklusive Rotor- und Stellerdynamik), 4 Eingangsgrößen (lateraler und longitudinaler Stick, Pedal und Kollektiv) und 4 Ausgangsgrößen (Rollrate, Nickrate, Gierrate und Vertikalgeschwindigkeit) verwendet, die DLR-Modelle besitzen die gleichen Ein- und Ausgangsgrößen aber 8 (Bo-105) bzw. 10 (EC-135) Zustandsgrößen.

Hubschraubertypisch besitzen alle Modelle relativ langsame instabile Pole, so dass eine Simulation der Modelle auch mit einer Inversen, die versucht, alle Pole durch entsprechende Nullstellen zu kompensieren, aus numerischen Gründen nur für einen begrenzten Zeitraum stabil durchgeführt werden kann [6]. Erschwerend kommt hinzu, dass beim UH-60-Modell vier Nullstellen in der rechten Halbebene liegen, so dass die entsprechende Inverse zwei instabile Polpaare besitzt. Da die Nullstellen aber außerhalb des Nutzfrequenzbereichs des Hubschraubers liegen, lässt sich ihre Auswirkung durch eine in [5] und



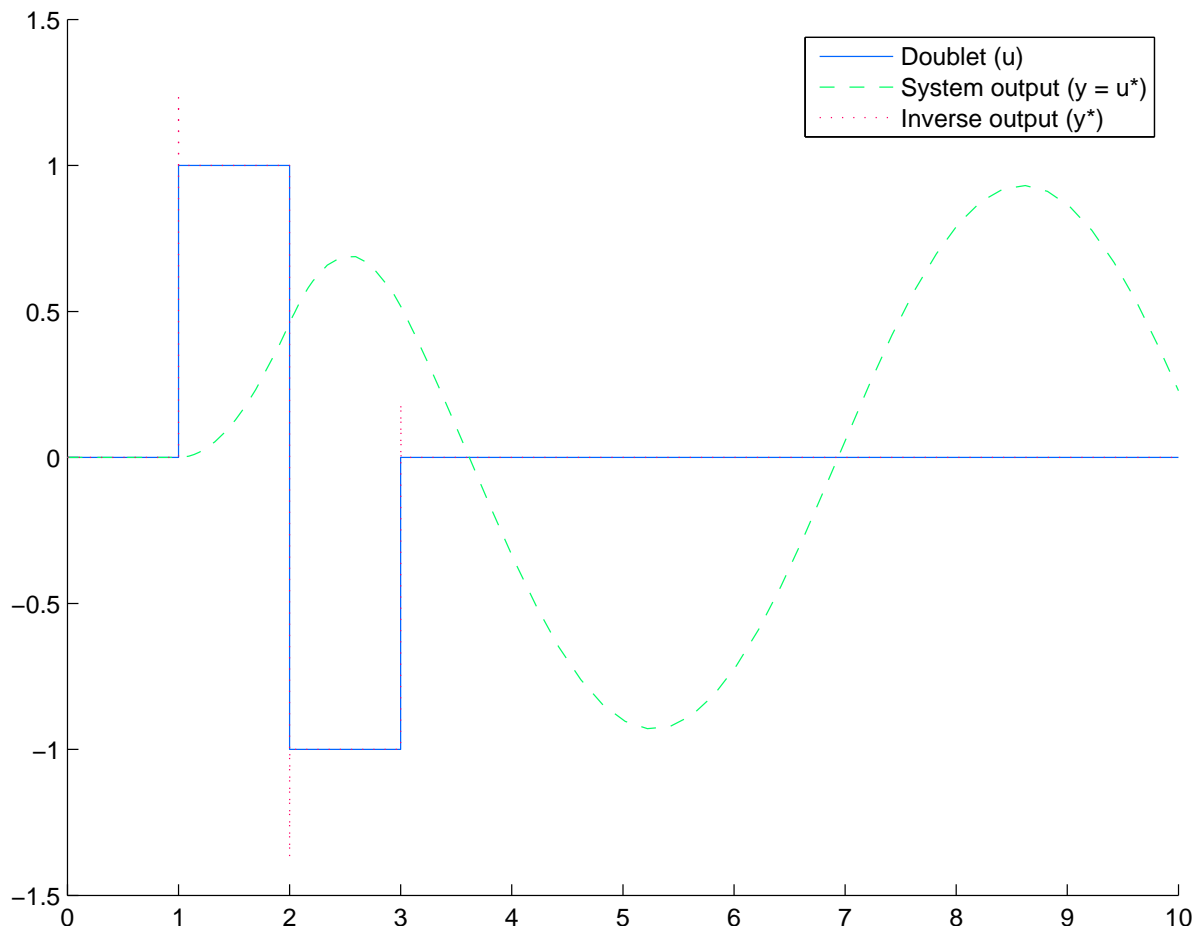


Abbildung 10: Ergebnis der Inversion eines nichtlinearen Systems

[6] beschriebene Spiegelung mit Hilfe eines Allpasses in eine für Offline-Anwendungen unkritische Totzeit umwandeln.

## Literatur

- [1] U. S. State Department: *Defense: Helicopter Aeromechanics: Memorandum of Understanding between the United States of America and the Federal Republic of Germany*. Signed at Alexandria and Bonn, 1994.
- [2] Labows, S. J.; Blanken, C. L.; Tischler M. B.: UH-60 Black Hawk Disturbance Rejection Study for Hover/Low Speed Handling Qualities Criteria and Turbulence Modeling. In *American Helicopter Society 56<sup>th</sup> Annual Forum*. Virginia Beach, Virginia, May 2 – 4, 2000.

- [3] Lusardi, J. A.; Blanken, C.L.; Tischler M. B.: Piloted Evaluation of a Mixer Equivalent Turbulence Simulation Model. In *American Helicopter Society 59<sup>th</sup> Annual Forum*. Phoenix, Arizona, May 6 – 8, 2003.
- [4] Lawden, D.F.: Elliptic Functions and Applications. In *Applied Mathematical Sciences*. Vol 80. Springer, 1989.
- [5] Hess, R.A.; Siwakosit, W.: Assessment of Flight Simulator Fidelity in Multiaxis Tasks Including Visual Cue Quality. In *Journal of Aircraft*. Vol. 38, No. 4, 2001. S. 607 – 614.
- [6] Buchholz, J. J.; v. Grünhagen, W.: *Inversion Impossible?*, IB 111-2003/34. DLR Braunschweig. [http://buchholz.hs-bremen.de/inversion/inversion\\_impossible.pdf](http://buchholz.hs-bremen.de/inversion/inversion_impossible.pdf). 2003.