REIHE AUTOMATISIERUNGSTECHNIK · BAND 232

Georg Brack

Prozeß- und Regelkreisdynamik

Berechnung mit BASIC

REIHE AUTOMATISIERUNGSTECHNIK 232
Herausgegeben von
G. Brack, H. Fuchs, G. Paulin, R. Piegert, G. Schwarze und E.-G. Woschni

Prozeß- und Regelkreisdynamik

Berechnung mit BASIC

Georg Brack

VEB VERLAG TECHNIK BERLIN

Brack, Georg:
Prozeß- und Regelkreisdynamik : Berechnung mit BASIC / Georg Brack. – 1. Aufl. – Berlin : Verl. Technik, 1988. – 88 S. : 32 Bilder, 12 Taf. – (Reihe Automatisierungstechnik; 232)
ISBN 3-341-00531-5
NE: GT

ISBN 3-341-00531-5

ISSN 0484-3436 REIHE AUTOMATISIERUNGSTECHNIK (Berlin, DDR)

Federführender Herausgeber: *Gunter Schwarze*
Begutachtender Herausgeber: *Gerhard Paulin*
1. Auflage

Lizenz 201 · 370/79/88
Printed in the German Democratic Republic
Satz: Karl-Marx-Werk Pößneck V15/30
Druck und buchbinderische Verarbeitung:
Druckerei August Bebel, Gotha
Lektor: Jürgen Reichenbach
Einbandgestaltung: Kurt Beckert
LSV 3043 · VT 3/5951-1

Bestell-Nr.: 553 9317
00480

Vorwort

Mit der breiteren Verfügbarkeit von Mikrorechnern und Heimcomputern ist jedem einzelnen Ingenieur und Studenten die Möglichkeit gegeben, formalisierbare Arbeiten dem Rechner zu übertragen und sich selber auf geistig-schöpferische Tätigkeiten zu konzentrieren.
Der Band will dazu einen Beitrag leisten, indem er dem Leser neben den für das Verständnis notwendigen Grundlagen der Theorie zum dynamischen Verhalten von Prozessen und Regelkreisen eine Reihe erprobter Algorithmen und Rechenprogramme zur Verfügung stellt, die vom Nutzer unmittelbar angewendet werden können. Ausgangsbasis ist die Zustandsdarstellung kontinuierlicher Systeme, da sich die Zustandsgleichungen zwanglos aus den Bilanzgleichungen der theoretischen Prozeßanalyse herleiten lassen. Aus der Zustandsdarstellung der Regelstrecke folgt die des geschlossenen Regelkreises ebenso, wie sich daraus Klemmenmodelle herleiten lassen. Durch Diskretisierung der Zeit ergeben sich diskontinuierliche Modelle, die zur Simulation des Zeitverhaltens mit dem Rechner besonders geeignet sind. Für alle dabei notwendigen Aktivitäten wird ein Paket von Methoden und Programmen angeboten, die jedoch in jedem Fall auch selbständig verwendet werden können.
Der Band beschränkt sich aber nicht darauf, Rechenprogramme vorzustellen und zu erläutern. Eigentlich sollte man kein Programm nutzen, ohne eine Vorstellung von dem ihm zugrunde liegenden theoretischen Hintergrund zu haben. Deshalb stellt der Band zu jedem Programm auch die theoretischen Grundlagen bereit. Im Prinzip wird dabei nicht über das übliche Niveau der REIHE AUTOMATISIERUNGSTECHNIK hinausgegangen, wie es durch die Bände 151, 152 und 214 dieser Reihe fixiert ist. Trotzdem wird mancher Leser vielleicht mit gewisser Bestürzung feststellen, daß im Text ziemlich freizügig Beschreibungsmittel verwendet werden, die manchmal zur „höheren Theorie" gerechnet werden, obwohl es tatsächlich nur verkürzende und Schreibarbeit sparende Darstellungsformen sind. Man möge aber bitte bedenken: Der Rechner soll es ermöglichen, reale Systeme zu untersuchen. Diese Systeme sind nun aber komplizierter als die üblichen Schulbeispiele 1. und 2. Ordnung. Die Darstellung durch Vektoren und Matrizen, im Rechner durch entsprechende Felder, ermöglicht es erst, mit solchen komplizierteren Systemen fertig zu werden und die Zusammenhänge trotzdem noch überschaubar zu halten. Dem gleichen Zweck dient auch der hier eingesetzte Apparat der z-Transformation, der nur dazu da ist, die mühsame Schreibarbeit bei den Differenzengleichungen zu rationalisieren.
Die Programme in diesem Band wurden auf dem Rechner ZX Spectrum entwickelt und auf den KC 85/3 übertragen. In der hier abgedruckten Form sind sie auf diesem Rechner lauffähig und in jedem Einzelfall getestet worden. Auf weitere Vereinfachungen, die der BASIC-Interpreter dieses Rechners erlauben würde, wurde verzichtet. Viele Beispiele sollen dazu dienen, sowohl die Vorgehensweisen der

Theorie als auch die Arbeitsweise der Programme völlig durchschaubar zu gestalten.

Für die gute Zusammenarbeit bei der Veröffentlichung dieses Bandes darf ich mich besonders bei meinem Herausgeber, Koll. Prof. *Schwarze*, sowie dem VEB Verlag Technik herzlich bedanken.

Georg Brack

Inhaltsverzeichnis

1. Kontinuierliche Zustandsmodelle

1.1. Modellierung

Am Anfang jeder rechnergestützten Untersuchung eines realen oder gedachten Systems steht die Aufstellung eines mathematischen dynamischen Modells. Je nach den gegebenen Möglichkeiten kann dabei experimentell vorgegangen werden, oder man stützt sich auf allgemeingültige Gesetzmäßigkeiten und nimmt eine theoretische Modellierung vor. Beide Grundmethoden werden ausführlich in der Literatur beschrieben (s. z. B. [1; 2; 3]). Das soll hier nicht wiederholt werden. Statt dessen wird als Beispiel die im Bild 1 dargestellte technologische Anlage betrachtet. An diesem Beispiel werden, soweit dies möglich ist, alle im weiteren behandelten Algorithmen und Programme beschrieben.

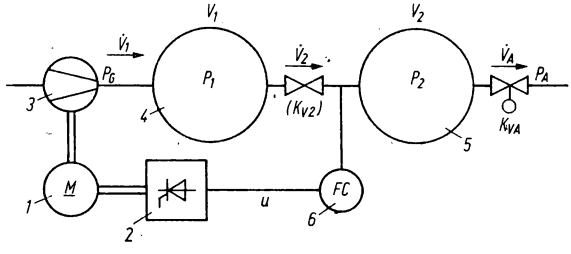


Bild 1. Gasdurchflußregelung
1 Motor; 2 Thyristorsteller; 3 Verdichter; 4, 5 Speicher; 6 Durchflußregler

Bild 1 zeigt das Schema einer Gas-Durchflußregelung. Ein Motor 1, gespeist von einem Thyristorsteller 2, treibt einen Verdichter 3 an, der einen Gasstrom in zwei hintereinanderliegende Speicher 4, 5 liefert. Ein Durchflußregler 6 soll den Strom $\dot{V}_2$ zwischen den beiden Speichern auf einem vorgegebenen Sollwert halten. Zwischen den Speichern und gegen den Abfluß nach Atmosphäre liegen Strömungswiderstände mit den Werten k_{V2} (konstant) bzw. k_{VA} (veränderbar).

1.1.1. Motor-Verdichter-Aggregat

Die Untersuchung des Motor-Verdichter-Aggregats erfolgte experimentell und erbrachte folgende Ergebnisse:

Stellverhalten

Bei einer Änderung des Steuersignals ũ für den Thyristorsteller im Bereich 0 bis 100% ergab sich für den Ausgangsdruck p_G des unbelasteten Verdichters ($\dot{V}_1 = 0$) die im Bild 2a gezeigte statische Kennlinie. Alle Druckwerte stellen Absolutdrücke dar.

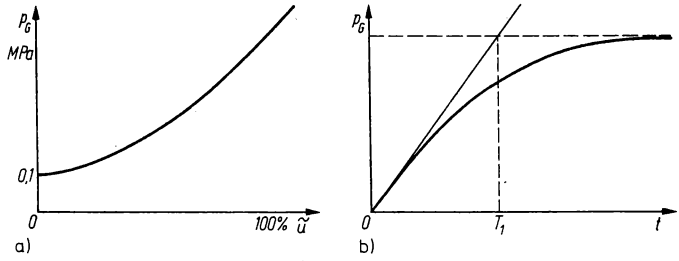


Bild 2. Stellverhalten
a) statische Kennlinie
b) Sprungantwort

Die Kennlinie kann durch folgende Gleichung dargestellt werden:

$$p_G = a\,\tilde{u}^2 + b\,.$$

Die Sprungantwort für das Stellverhalten, bedingt durch die mechanische Trägheit des rotierenden Systems, wird im Bild 2b gezeigt. Daraus läßt sich erkennen, daß ein Verzögerungsverhalten 1. Ordnung vorliegt. Die Differentialgleichung für das Motor-Verdichter-Aggregat, also sein dynamisches Modell, lautet danach

$$T_1 \frac{dp_G}{dt} + p_G = a\,\tilde{u}^2 + b\,. \tag{1}$$

Die aus den gemessenen Kurven entnehmbaren Zahlenwerte für a, b und T_1 sind zusammen mit anderen in Tafel 2 zusammengestellt.
In der Modellform eines Zustandsmodells stehen die Ableitungen der Veränderlichen nach der Zeit auf den linken Seiten der Gleichungen allein. Gl. (1) wird deshalb umgeformt:

$$\frac{dp_G}{dt} = -\frac{1}{T_1}\,p_G + \frac{a}{T_1}\,\tilde{u}^2 + \frac{b}{T_1}\,. \tag{2}$$

Um Schreibarbeit zu sparen, sollen in allen Modellgleichungen dieses Bandes die in Tafel 1 zusammengestellten Einheiten verwendet werden. Die Gleichungen sind somit Zahlenwertgleichungen mit den Einheiten nach Tafeln 1 und 2.

Tafel 1. Einheiten für die Modellgleichungen

Zeit, Zeitkonstante	s
Druck, Druckdifferenz	MPa
Volumen	m^3
Durchfluß	m^3/s

Belastungsverhalten

Der Ausgangsdruck des Verdichters ist abhängig vom entnommenen Förderstrom $\dot{V}_1$. Die Messung ergab die Kennlinie nach Bild 3. Unbelastet ergibt sich der Leerlaufdruck p_G, der mit steigender Belastung absinkt.
Die Auswertung der Kennlinie ergab eine Geradengleichung

$$p_1 = p_G - c\,\dot{V}_1\,.$$

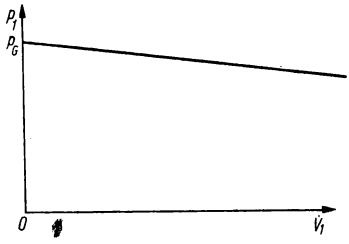


Bild 3. Belastungskennlinie des Verdichters
p_G Leerlaufdruck; p_1 Druck bei Belastung

Für den Durchfluß $\dot{V}_1$ (wie alle Durchflußwerte der folgenden Gleichungen auf Normbedingungen bezogen) folgt daraus

$$\dot{V}_1 = \frac{1}{c}(p_G - p_1) . \tag{3}$$

1.1.2. Speichersystem

Das Speichersystem kann auch ohne Experimente auf rein theoretischem Wege modelliert werden. Grundlage bilden die instationären Bilanzgleichungen für die in den beiden Speichern gespeicherten Gasmassen [1; 4].

Für die Änderungsgeschwindigkeit der im Speicher 4 enthaltenen Masse m_1 gilt

$$\frac{dm_1}{dt} = \dot{m}_1 - \dot{m}_2 = \varrho_N (\dot{V}_1 - \dot{V}_2) . \tag{4}$$

Für m_1 läßt sich mit der allgemeinen Gasgleichung schreiben

$$m_1 = p_1 V_1/(RT) .$$

Damit erhält man aus (4), wenn man für $\dot{V}_1$ die Gl. (3) einsetzt

$$\frac{dp_1}{dt} = \frac{RT\varrho_N}{V_1}\left(\frac{1}{c}p_G - \frac{1}{c}p_1 - \dot{V}_2\right) . \tag{5}$$

Ähnlich läßt sich für den Bilanzraum Speicher 5 mit der Masse m_2 und dem Innendruck p_2 schreiben

$$\frac{dm_2}{dt} = \frac{V_2}{RT}\frac{dp_2}{dt} = \dot{m}_2 - \dot{m}_A = \varrho_N (\dot{V}_2 - \dot{V}_A)$$

bzw.

$$\frac{dp_2}{dt} = \frac{RT\varrho_N}{V_2}(\dot{V}_2 - \dot{V}_A) . \tag{6}$$

R ist dabei die stoffspezifische Gaskonstante, T die absolute Temperatur und ϱ_N die Gasdichte im Normalzustand.
Es fehlen noch Beziehungen für die Volumenströme $\dot{V}_2$ bzw. $\dot{V}_A$. Diese hängen ab von den Drücken, die vor (p_v) und hinter (p_h) den jeweiligen Strömungswiderständen herrschen, sowie von deren Widerstandswerten, ausgedrückt durch k_V.

Setzt man dafür die aus der Literatur, z. B. [4], zu entnehmenden Beziehungen an

$$\dot{V} = k\, k_V \sqrt{\frac{(p_v - p_h)\, p_h}{\varrho_N\, T}} ,$$

dann erhält man nach Zusammenfassung aller konstanten Werte

$$\dot{V}_2 = d_2 \sqrt{(p_1 - p_2)\, p_2} \tag{7}$$

$$\dot{V}_A = d_3\, \tilde{k}_{VA} \sqrt{(p_2 - p_A)\, p_A} . \tag{8}$$

Dabei soll der Strömungswiderstand k_{V2} unveränderlich sein; sein Wert ist mit in d_2 enthalten. $\tilde{k}_{VA}$ ist der veränderbare, auf den Maximalwert bezogene Wert des Auslaßventils. $\tilde{k}_{VA}$ kann damit Werte zwischen 0 und 1 annehmen.
Ob die Beziehungen (7) und (8) die Vorgänge richtig widerspiegeln, hängt davon ab, ob die ihnen zugrunde liegenden Voraussetzungen erfüllt sind.

- Die Durchflüsse müssen, wie im Bild 1 eingezeichnet, von „links" nach „rechts" fließen.

Das mag im Normalfall selbstverständlich erscheinen. Es sind aber auch Betriebszustände vorstellbar (z. B. beim Abfahren), bei denen die Strömung in umgekehrter Richtung erfolgt.

- Die Druckabfälle über den Strömungswiderständen sind unterkritisch, d. h., es gilt $p_1 < 2\,p_2$ bzw. $p_2 < 2\,p_A$.

Ob diese Bedingungen erfüllt sind, kann nur aus der Lösung der Modellgleichungen erkannt werden. Ist sie etwa beim Ausfluß nicht erfüllt, dann muß anstelle (8) geschrieben werden

$$\dot{V}_A = d_3\, \tilde{k}_{VA}\, p_2/2 . \tag{9}$$

1.1.3. Zustandsgleichungen

Setzt man die Beziehungen (7) und (8) bzw. (9) in die Gln. (5) und (6) ein, dann erhält man mit den aus Tafel 2 entnehmbaren Zahlenwerten für die Konstanten die endgültigen Gleichungen

$$\frac{dp_G}{dt} = -0.08\, p_G + 0.168\, \tilde{u}^2 + 0.008 \tag{10}$$

$$\frac{dp_1}{dt} = 0.25 \left[0.7\, p_G - 0.7\, p_1 - 0.4 \sqrt{(p_1 - p_2)\, p_2}\right] \tag{11}$$

$$\frac{dp_2}{dt} = \begin{cases} 0.33 \left[0.4 \sqrt{(p_1 - p_2)\, p_2} - 0.25\, \tilde{k}_{VA}\, p_2\right] & \text{(12a)} \\ 0.33 \left[0.4 \sqrt{(p_1 - p_2)\, p_2} - 0.5\, \tilde{k}_{VA} \sqrt{(p_2 - p_A)\, p_A}\right] & \text{(12b)} \end{cases}$$

Gl. (12a) gilt im überkritischen Fall, in dem $p_2 \geqq 2p_A$ ist; im unterkritischen Fall ($p_2 < 2p_A$) gilt (12b).
Die Gln. (10), (11), (12) sind ein sog. Zustandsmodell mit den Zustandsgrößen p_G, p_1, p_2 und den Eingangsgrößen (Steuergrößen) $\tilde{u}$, $\tilde{k}_{VA}$. Mit den allgemeinen Symbo-

Tafel 2. Werte der Modellkonstanten

a	2.1	(MPa)
b	0.1	(MPa)
T_1	12.5	(s)
1/c	0.7	($m^3MPa^{-1}s^{-1}$)
d_2	0.4	($m^3MPa^{-1}s^{-1}$)
d_3	0.5	($m^3MPa^{-1}s^{-1}$)
$RT\varrho_N$	111	(MPa)
V_1	444	(m^3)
V_2	333	(m^3)

len q_i für die Zustandsgrößen und u_j für die Eingangsgrößen entsprechen sie der allgemeinen Form

$$\begin{aligned} \dot{q}_1 &= f_1(q_1, q_2, \ldots, q_n, u_1, u_2, \ldots, u_m) \\ \dot{q}_2 &= f_2(q_1, q_2, \ldots, q_n, u_1, u_2, \ldots, u_m) \\ \dot{q}_n &= f_n(q_1, \ldots, \quad q_n, u_1, \ldots, \quad u_m) . \end{aligned} \tag{13}$$

Die $f_1, f_2, \ldots, f_n$ sind allgemeine nichtlineare Funktionen der Zustands- und Eingangsgrößen.
Zur Vereinfachung der Schreibweise und um Platz zu sparen, wird auch von der Vektorschreibweise Gebrauch gemacht, bei der der Vektor $\mathbf{q}$ für die Zusammenfassung aller Zustandsgrößen steht:

$$\mathbf{q}^T = (q_1\, q_2 \ldots q_n) .$$

Das hochgestellte T bedeutet „transponiert". Es wird nur aus Gründen der Platzersparnis eingeführt. Wollte man $\mathbf{q}$ ohne das Transponiert-Symbol darstellen, dann müßten die Komponenten $q_1, q_2 \ldots$ untereinander gesetzt werden.
Entsprechend werden die Eingangsgrößen zum Eingangsvektor $\mathbf{u}^T = (u_1\, u_2 \ldots u_m)$ zusammengestellt, und für die Funktionen auf den rechten Seiten des Systems (13) wird der formale Funktionsvektor $\mathbf{f}^T = (f_1\, f_2 \ldots f_n)$ gesetzt. Damit läßt sich das System (13) sehr kurz in folgender Form schreiben:

$$\frac{d\mathbf{q}}{dt} = \mathbf{f}(\mathbf{q}, \mathbf{u}) . \tag{14}$$

1.1.4. Ausgabegleichungen

Als Ausgangsgrößen x_k eines Zustandsmodells werden diejenigen Größen bezeichnet, für die man sich speziell interessiert, z. B. als Meß- oder Regelgrößen. Allgemein hängen die p Ausgangsgrößen von den Zustands- und Eingangsgrößen ab:

$$\begin{aligned} x_1 &= g_1(q_1, q_2, \ldots, q_n, u_1, u_2, \ldots, u_m) \\ &\vdots \\ x_p &= g_p(q_1, \ldots, \quad q_n, u_1, \ldots, \quad u_m) \end{aligned} \tag{15}$$

Mit der Vektorschreibweise für den Ausgangsvektor $\mathbf{x}^T = (x_1\, x_2 \ldots x_p)$ und den Funktionenvektor $\mathbf{g}^T = (g_1\, g_2 \ldots g_p)$ läßt sich für das System (15) auch kurz schreiben:

$$\mathbf{x} = \mathbf{g}(\mathbf{q}, \mathbf{u}) . \tag{16}$$

Im konkreten Modellierungsbeispiel sollen die Ausgangsgrößen sein: x_1 Druck p_2, x_2 Volumenstrom $\dot{V}_2$. Damit lauten die beiden Ausgabegleichungen

$$x_1 = p_2 \tag{17}$$

$$x_2 = d_2 \sqrt{(p_1 - p_2)\,p_2} = 0.4 \sqrt{(p_1 - p_2)\,p_2}\ . \tag{18}$$

Die Gln. (10), (11), (12), (17), (18) bilden zusammen das vollständige Zustandsmodell, mit dessen Hilfe in den folgenden Abschnitten das Verhalten des Systems untersucht wird.

1.2. Lösung der nichtlinearen Modellgleichungen; Programm RUKU4

▶ Für Verweise auf Programmzeilen in Programmablaufplänen und BASIC-
▶ Programmen gilt folgende Vereinbarung: Zeilennummern in Programmab-
▶ laufplänen werden in Hochkommas eingeschlossen. Nicht derart gekenn-
▶ zeichnete Zeilennummern beziehen sich auf BASIC-Programme.

Das Gleichungssystem (13) bzw. in der Kurzdarstellung (14) ist ein System von n gewöhnlichen nichtlinearen Differentialgleichungen 1. Ordnung. [Die Gln. (15) bzw. (16) sind einfache algebraische Beziehungen, deren Auswertung keine besonderen Probleme mit sich bringt.]

Zur näherungsweisen Lösung von Differentialgleichungen und -systemen gibt es verschiedene Verfahren der numerischen Mathematik, die alle darauf beruhen, daß die kontinuierlich ablaufende Zeit t in eine Folge diskreter Zeitpunkte zerlegt wird, für die jeweils punktweise die Lösung berechnet wird (Bild 4). Die Abstände der diskreten Zeitpunkte, in der Mathematik gewöhnlich mit dem Symbol h bezeichnet, werden hier T genannt. Sie müssen im Verlaufe einer Rechnung nicht notwendigerweise konstant sein. Die Verfahrensfehler sind i. allg. um so kleiner, je kleiner T gewählt wird.

Von den verschiedenen bekannten Verfahren wird eine Methode nach Runge–Kutta ausgewählt, die auf unterschiedliche Weise dargestellt werden kann. Bild 5

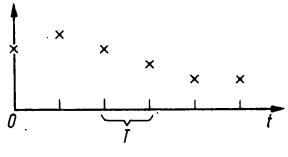


Bild 4. Diskretisierung der Zeit

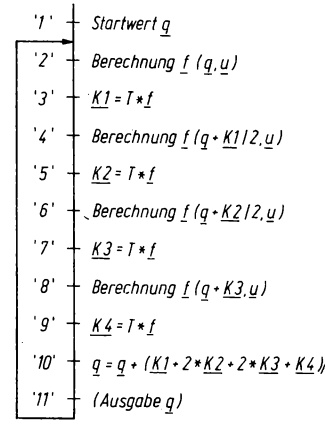


Bild 5. Algorithmus nach Runge–Kutta für $\boldsymbol{u}$ = const
Unterstrichene Symbole bedeuten Vektoren.

zeigt den Kern des Algorithmus als Programmablaufplan (PAP), und zwar für den Sonderfall, daß die Komponenten des Eingangsvektors **u** im betrachteten Zeitbereich konstant sind, also z. B. Sprungfunktionen darstellen, die im Startzeitpunkt t_0 ihren Wert gewechselt haben.
Zum Startzeitpunkt t_0 gehört der Startwert **q** des Zustandsvektors (Bild 5, Zeile '1'). Im Algorithmus taucht t_0 selber nicht auf, eben weil **u** als zeitunabhängig vorausgesetzt ist.
In Zeile '10' wird der neue Wert von **q** berechnet, der zum Zeitpunkt $t_0 + T$ gehört. Dazu werden in den Zeilen '2' bis '9' vier Hilfsgrößen **K1** bis **K4** berechnet.
Soll der Algorithmus auch für zeitabhängige u gelten (z. B. $u_i = \sin \omega t$), dann sind einige Zeilen wie folgt zu ergänzen:

Zeile '1'	Neben dem Startwert **q** ist auch der Startzeitpunkt t_0 vorzugeben; $t = t_0$
Zeile '2'	Berechnung $\mathbf{f(q, u(t))}$
Zeile '4'	Berechnung $\mathbf{f(q + K1/2, u(t + T/2))}$
Zeile '6'	Berechnung $\mathbf{f(q + K2/2, u(t + T/2))}$
Zeile '8'	Berechnung $\mathbf{f(q + K3, u(t + T))}$
Zeile '10a'	Einfügen: $t = t + T$.

Programm RUKU 4

```
1Ø  REM LOESUNG NICHTLINEARES DGL-SYSTEM
2Ø  INPUT "DIMENSION";N: DEFFN Q(X) = X*X: LET TP = Ø
3Ø  DIM Q(N), X(N), F(N), K(4,N)
4Ø  PRINT: PRINT "EINGABE ANFANGSWERTE": PRINT
5Ø  FOR I = 1 TO N: INPUT Q(I): NEXT I
6Ø  INPUT "EINGANGSAMPLITUDE";XE: INPUT "INDEX AUSGANGSGROESSE";R
7Ø  INPUT "ZEITABSTAND";T: CLS
8Ø  IF TP = Ø THEN PRINT TP, Q(R)
9Ø  FOR K = Ø TO 25
1ØØ FOR I = 1 TO N: LET X(I) = Q(I): NEXT I: GOSUB 25Ø
11Ø FOR I = 1 TO N: LET K(1,I) = T*F(I): LET X(I) = Q(I) + K(1,I)/2
12Ø NEXT I: GOSUB 25Ø
13Ø FOR I = 1 TO N: LET K(2,I) = T*F(I): LET X(I) = Q(I) + K(2,I)/2
14Ø NEXT I: GOSUB 25Ø
15Ø FOR I = 1 TO N: LET K(3,I) = T*F(I): LET X(I) = Q(I) + K(3,I)
16Ø NEXT I: GOSUB 25Ø: FOR I = 1 TO N
17Ø LET K(4,I) = T*F(I): LET Q(I) = Q(I) + (K(1,I) + 2*K(2,I) + 2*K(3,I) + K(4,I))/6
18Ø NEXT I: LET TP = TP + T: PRINT TP,Q(R)
19Ø NEXT K: INPUT "WEITERE WERTE J/N";I$
2ØØ IF I$ = "J" GOTO 7Ø
21Ø END
25Ø REM PARAMETERUEBERGABE
26Ø LET KV = 1: LET PG = X(1): LET P1 = X(2): LET P2 = X(3): LET U = XE
27Ø REM FUNKTIONEN
28Ø LET F(1) = -Ø.Ø8*PG + Ø.168*FN Q(U) + Ø.ØØ8
29Ø LET F(2) = (Ø.7*(PG - P1) - Ø.4*SQR((P1 - P2)*P2))/4
3ØØ IF P2 <= Ø.2 GOTO 32Ø
31Ø LET F(3) = (Ø.4*SQR((P1-P2)*P2) - Ø.25*KV*P2)/3: RETURN
32Ø LET F(3) = (Ø.4*SQR((P1-P2)*P2) - Ø.5*KV*SQR((P2 - Ø.1)/1Ø))/3
33Ø RETURN
```

Zur Umsetzung des Algorithmus in das BASIC-Programm RUKU 4 wird von Bild 5 ausgegangen. Allgemeingültig sind die Zeilen '1', '3', '5', '7', '9', '10', '11'. Sie hängen nicht ab vom konkreten Problem und werden im unveränderlichen Programmrumpf aufgenommen (Programm RUKU 4, Zeilen 1ØØ bis 17Ø). In den Zeilen '2', '4', '6', '8' werden dagegen die Werte der Funktionen f_i, symbolisch zusammengefaßt zum Funktionsvektor **f**, berechnet. Das sind die Funktionen der rechten Seiten des Gleichungssystems (13). Sie sind problemabhängig und müssen jedesmal neu programmiert werden. Dazu bringt man sie am besten in einem Unterprogramm unter (Zeilen 28Ø bis 33Ø).
Die Vektoren **q**, **f** werden im BASIC-Programm durch Felder der Dimension N (Anzahl der Zustandsgleichungen) dargestellt: Q(N), F(N). Die **K1** bis **K4** werden zu einem Feld K(4,N) zusammengefaßt. Ein weiteres Feld X(N) ist notwendig, weil das Feld Q im Programm noch bis zur Zeile 15Ø benötigt wird und erst in Zeile 17Ø mit neuen Werten überspeichert werden darf. Die Variablen zur Übergabe an das Unterprogramm sind also X(1) bis X(N); sie werden in den Zeilen 1ØØ, 13Ø, 15Ø gebildet.
Die im PAP Bild 5 symbolisch dargestellten Vektoroperationen müssen im Programmrumpf komponentenweise durch Laufanweisungen verwirklicht werden. Man vergleiche z. B. die Zeilen '5', '6' und 13Ø, 14Ø.
Das Unterprogramm auf den Zeilen 28Ø bis 33Ø wurde für die Lösung der Gln. (10) bis (12) geschrieben. Die Zeilen 28Ø bis 32Ø stellen die rechten Seiten dieser Gleichungen dar. Für die Veränderlichen wurden die technologischen Symbole in angepaßter Schreibweise (PG, P1, P2) verwendet. Im allgemein verwendbaren Programmrumpf heißen die Veränderlichen dagegen X(1) bis X(N); hier N = 3. In Zeile 26Ø muß daher zunächst eine Namensanpassung vom Hauptprogramm zum Unterprogramm erfolgen. Da das Gesamtprogramm u. a. dafür vorgesehen ist, die Reaktion eines Systems auf verschiedene Werte einer Eingangssprungamplitude zu berechnen, wird in Zeile 6Ø die Sprungamplitude XE eingegeben und in Zeile 26Ø der konkreten Größe ũ (Programmname U) zugewiesen. Sollte dagegen etwa die Reaktion auf sprungförmige Veränderungen von $\tilde{k}_{VA}$ mit verschiedenen Amplituden untersucht werden, während ũ = 0.8 ist, wäre in Zeile 26Ø zu schreiben KV = XE : U = Ø.8.
Die übrigen Zeilen des Programms RUKU4 dienen dem Dialog mit dem Nutzer. Nach Eingabe der Anfangswerte der Zustandsgrößen (Zeilen '1' bzw. 5Ø) kann in Zeile 7Ø ausgewählt werden, welche der drei Zustandsgrößen Q(1) = PG, Q(2) = P1, Q(3) = P2 in Zeile 18Ø zusammen mit der laufenden Zeit t (Programmname TP) auf dem Bildschirm dargestellt werden soll. Nach jeweils 25 Werten besteht die Möglichkeit, die jeweils folgenden mit einer ggf. veränderten Schrittweite T berechnen und ausgeben zu lassen bzw. die Rechnung zu beenden (Zeilen 19Ø bis 2ØØ, 7Ø).
Ein mit diesem Programm berechneter Verlauf des Druckes p_2 ist im Bild 6 zu sehen, und zwar für folgende Bedingungen: Anfangszustand $p_G = p_1 = p_2 = Ø.1$ (das heißt Atmosphärendruck); Amplitude des sprungförmig vom Wert null eingeschalteten Eingangssignals ũ : Ø.8.
Versucht man dagegen, mit diesem Programm die Reaktion des Systems auf folgende Situation zu berechnen:

Anfangsdrücke $p_G = 0.1$, $p_1 = 0.1$, $p_2 = 0.5$, $\tilde{u} = 0.8$,

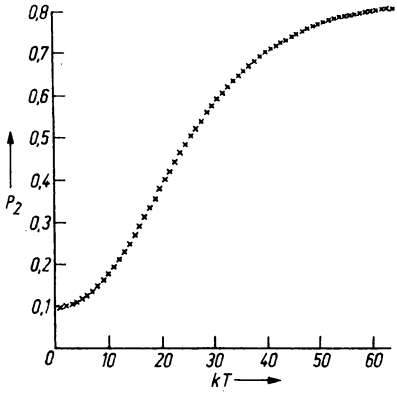


Bild 6. Druckverlauf p_2 nach einer sprungförmigen Erhöhung von ũ vom Wert 0 auf 0.8
Anfangszustand: Atmosphärendruck

dann erhält man sofort Fehlermeldung (unzulässiges Funktionsargument in Zeile 29Ø). Der Grund ist einfach: Nach diesen Anfangsbedingungen wäre $p_1 < p_2$; das Gas würde vom Speicher 5 zurück zum Speicher 4 fließen. Das ist im Modell nicht vorgesehen; Gl. (7), die der Zeile 29Ø zugrunde liegt, gilt nur für $p_1 \geqq p_2$.

1.3. Bestimmung des stationären Arbeitspunktes

1.3.1. Aufgabenstellung

Um den stationären Arbeitspunkt des dynamischen Systems zu ermitteln, in dem alle Veränderlichen konstant sind, werden in den Zustandsgleichungen (13) die linken Seiten, d. h. die Ableitungen nach der Zeit, gleich null gesetzt. Das führt auf ein algebraisches Gleichungssystem mit mehreren Unbekannten, das gelöst werden muß. Die Gleichungen sind oftmals nichtlinear.

1.3.2. Lösungsalgorithmen; Rechenprogramm NEWT

Zur Lösung einer nichtlinearen Gleichung mit einer einzigen Unbekannten ist aus der Mathematik das Näherungsverfahren nach Newton bekannt (Bild 7). Gesucht ist die Nullstelle x* einer nichtlinearen Funktion f(x). Beginnt man mit einem Nä-

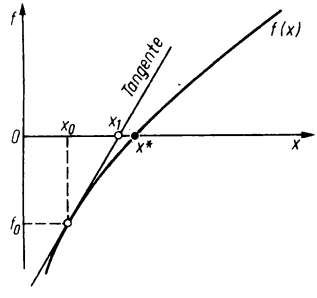


Bild 7. Lösungsverfahren nach Newton für eine nichtlineare Gleichung mit einer Unbekannten

herungswert x_0, dann erhält man eine bessere Näherung x_1 aus dem Schnittpunkt der Tangente im Punkt x_0 an die Kurve f(x) mit der Achse $f = 0$. Die Steigung der Tangente ist gleich der Ableitung f'(x) an der Stelle x_0. Daraus ergibt sich das Newtonsche Verfahren in der Form

$$\begin{aligned} x_1 &= x_0 - [f'(x_0)]^{-1} f(x_0) \\ x_2 &= x_1 - [f'(x_1)]^{-1} f(x_1) \quad \text{usw.} \end{aligned} \tag{19}$$

Dieses Verfahren läßt sich auch für n Gleichungen mit n Unbekannten anwenden. Die Unterschiede sind folgende:

1. An die Stelle der einfachen (skalaren) Veränderlichen x tritt der Vektor **x** mit n unbekannten Komponenten.
2. An die Stelle der skalaren Funktion f tritt der Funktionenvektor **f**, der ebenfalls n Komponenten (Gleichungen) umfaßt.
3. Anstelle einer einzigen Ableitung f' gibt es insgesamt n^2 Ableitungen. (Jede der n Funktionen ist nach jeder der n Veränderlichen abzuleiten.) Diese partiellen Ableitungen können zu einer Matrix **Y** zusammengestellt werden.
4. An die Stelle der Multiplikation mit $[f']^{-1}$ in (19) muß jetzt die Multiplikation mit der inversen Matrix $\mathbf{Y}^{-1}$ treten.

Das Newton-Verfahren (19) erhält damit die Form (20)

$$\begin{aligned} \mathbf{x}_1 &= \mathbf{x}_0 - \mathbf{Y}^{-1}(\mathbf{x}_0)\, \mathbf{f}(\mathbf{x}_0) \\ \mathbf{x}_2 &= \mathbf{x}_1 - \mathbf{Y}^{-1}(\mathbf{x}_1)\, \mathbf{f}(\mathbf{x}_1) \quad \text{usw.} \end{aligned} \tag{20}$$

Bild 8 zeigt diesen Algorithmus als PAP. Nach einer Vorgabe geeigneter Startwerte wird geprüft, ob schon eine Lösung vorliegt. Als Maß dafür dient die Erfüllung der Bedingung $\sum |f_i| \leqq eps$ für alle $i = 1, \ldots, n$ (Zeile '3'). Ist diese Bedingung nicht erfüllt, wird der Algorithmus nach (20) wiederholt abgearbeitet.

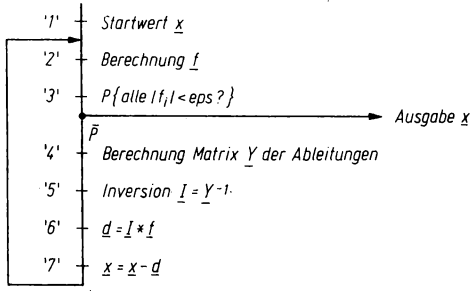


Bild 8. Programmablaufplan zum Newton-Verfahren zur Lösung nichtlinearer Gleichungssysteme

Programm NEWT

```
1Ø  REM LOESUNG NICHTLINEARES GLEICHUNGSSYSTEM
2Ø  INPUT "DELTA "; DEL: INPUT "DIMENSION "; N
3Ø  DIM X(N), D(N), F(N), G(N), H(N), Z(N), E(N,N), I(N,N), Y(N,N): PRINT
4Ø  DEFFN Q(X) = X * X: PRINT "STARTWERTE X(1) ...X(";N;")"
5Ø  FOR P = 1 TO N: INPUT X(P): NEXT P : CLS
6Ø  GOSUB 1ØØØ: LET D = Ø: FOR P = 1 TO N: LET G(P) = F(P): PRINT G(P)
7Ø  LET D = D + ABS(G(P)): NEXT P: PRINT
8Ø  IF D < 1E-3 THEN CLS: PRINT "NULLSTELLEN": FOR P = 1 TO N: PRINT X(P):
    NEXT P: END
```

```
 9Ø  FOR P = 1 TO N: LET X(P) = X(P) + DEL: GOSUB 1ØØØ
1ØØ  FOR J = 1 TO N: LET H(J) = F(J): NEXT J
11Ø  LET X(P) = X(P) - 2 * DEL: GOSUB 1ØØØ
12Ø  FOR J = 1 TO N: LET Y(J,P) = (H(J) - F(J))/DEL/2: NEXT J
13Ø  LET X(P) = X(P) + DEL: NEXT P: GOSUB 5ØØ
14Ø  FOR P = 1 TO N: LET D(P) = Ø
15Ø  FOR J = 1 TO N: LET D(P) = D(P) + I(P,J) * G(J): NEXT J
16Ø  LET X(P) = X(P) - D(P): NEXT P: GOTO 6Ø
5ØØ  REM MATRIZENINVERSION
51Ø  IF Y(1,1) = Ø THEN PRINT "GLEICHUNGEN VERTAUSCHEN !": END
52Ø  FOR I = 1 TO N: FOR J = 1 TO N: LET E(I,J) = Ø: NEXT J: LET E(I,I) = 1: NEXT I
53Ø  FOR K = 1 TO N - 1: FOR I = K + 1 TO N: IF Y(I,K) = Ø GOTO 56Ø
54Ø  LET Z(I) = Y(K,K)/Y(I,K): FOR L = 1 TO N: LET E(I,L) = E(K,L) - Z(I) * E(I,L)
55Ø  NEXT L: FOR J = K TO N: LET Y(I,J) = Y(K,J) - Z(I) * Y(I,J): NEXT J
56Ø  NEXT I: NEXT K: IF Y(N,N) = Ø THEN PRINT "UNLOESBAR !": END
57Ø  FOR L = 1 TO N: LET I(N,L) = E(N,L)/Y(N,N)
58Ø  FOR J = N - 1 TO 1 STEP -1: LET S = Ø
59Ø  FOR I = J + 1 TO N: LET S = S + Y(J,I) * I(I,L): NEXT I
 6ØØ  LET I(J,L) = (E(J,L) - S)/Y(J,J): NEXT J: NEXT L: RETURN
1ØØØ  REM PARAMETERUEBERGABE
1Ø1Ø  LET PG = X(1): LET U = X(2): LET P1 = X(3): LET P2 = X(4)
1Ø2Ø  REM FUNKTIONSGLEICHUNGEN
1Ø3Ø  LET F(1) = 2.1 * FN Q(U) - PG + Ø.1
1Ø4Ø  LET F(2) = Ø.7 * PG - Ø.7 * P1 - Ø.4 * SQR((P1 - P2) * P2)
1Ø5Ø  LET F(3) = Ø.4 * SQR((P1 - P2) * P2) - Ø.25 * P2
1Ø6Ø  LET F(4) = Ø.2 - Ø.4 * SQR((P1 - P2) * P2)
1Ø7Ø  RETURN
```

Bei einer Umsetzung in das Rechenprogramm NEWT ist wieder zu beachten, daß die Funktionen f_i jeweils problemspezifisch programmiert werden müssen. Sie sind deshalb im Unterprogramm (Zeilen 1Ø3Ø bis 1Ø6Ø) untergebracht. In der vorangehenden Zeile 1Ø1Ø werden die allgemeingültigen Variablennamen X(1) bis X(N) aus dem Hauptprogramm den technologischen Variablennamen in den Funktionsgleichungen zugeordnet.
Im Hauptprogramm sind folgende Detailprobleme zu lösen:

1. Zeile '4': Berechnung der Matrix Y (Programmname: Y(N,N))
Das Element Y(J,P) dieses Feldes stellt die Ableitung der Funktion F(J) nach der Veränderlichen X(P) dar:

$$Y(J,P) = \frac{\partial F(J)}{\partial X(P)}; \qquad J,P = 1,\ldots,N. \tag{21}$$

Diese Ableitungen werden numerisch gebildet. Dazu wird X(P) = X(P) + Delta gesetzt. Daraus wird der Wert der Funktion $F^+(J)$ berechnet (Programmname: H(J)) (Zeilen 9Ø, 1ØØ). Als nächstes wird X(P) vom Startwert aus um Delta vermindert; daraus berechnet man den Funktionswert $F^-(J)$ (Zeile 11Ø).
Die gesuchte Ableitung (23) berechnet man dann aus

$$Y(J,P) = \frac{F^+(J) - F^-(J)}{2 * \text{Delta}} \qquad \text{(Zeile 12Ø).}$$

2. Zeile '5': Berechnung der inversen Matrix $\mathbf{I} = \mathbf{Y}^{-1}$
Das ist ein Standardproblem der numerischen Mathematik, das in vielen Darstellungen beschrieben wird [6]. Das hierzu im Unterprogramm in den Zeilen 51Ø bis 6ØØ enthaltene Inversionsprogramm verlangt der Einfachheit halber, daß das Element Y(1,1) der zu invertierenden Matrix von null verschieden ist (Zeile 51Ø). Das bedeutet: In der Funktion F(1) des Unterprogramms (Zeile 1Ø3Ø) muß die Variable X(1) enthalten sein. Da die Numerierung der Funktionen völlig beliebig ist, läßt sich diese Bedingung auf jeden Fall erfüllen.
Es kann sein, daß die Matrix **Y** nicht invertierbar ist. Dann erfolgt Programmabbruch in Zeile 56Ø. Möglicherweise hilft dann ein Neustart mit anderen Näherungswerten weiter.
Die Bildung des Vektors **d** (Zeile '6') erfolgt in den Zeilen 14Ø und 15Ø. Zeile '7' wird durch die Laufanweisung in Zeile 16Ø realisiert.
Im Dialog mit dem Rechner wird vor der Eingabe der Dimension N ein Zahlenwert für Delta verlangt. Er sollte klein gegenüber den Beträgen der X(P) sein. Die einzugebenden Startwerte (Zeilen 4Ø, 5Ø) sollten in der Nähe der vermuteten Lösungswerte liegen, die oftmals aus technologischen Erwägungen abgeschätzt werden können.
Im Verlauf der Rechnung werden jeweils die N Werte der Funktionen F(J) ausgegeben (Zeile 6Ø). Daran kann man den Fortschritt der Rechnung verfolgen; die Werte müssen betragsmäßig kleiner werden als die Abbruchschranke eps, die im Programm in Zeile 8Ø mit 10^{-3} festgesetzt worden ist.

1.3.3. Anwendungsbeispiel

Im Programm NEWT wurden die Gleichungen des problemspezifischen Unterprogramms ab Zeile 1ØØØ für folgende Aufgabe aus dem Beispiel nach Bild 1 formuliert: Der Sollwert des Durchflusses $\dot{V}_2$ soll 0.2 betragen. Gesucht sind die dazugehörigen stationären Werte von p_G, $\tilde{u}$, p_1 und p_2. Die Gleichungen zur Bestimmung dieser vier Unbekannten ergeben sich aus den rechten Seiten des Zustandsgleichungssystems (10) bis (12a), jeweils multipliziert mit konstanten Faktoren (12.5; 4; 3) (Funktionen F(1) bis F(3)). Die Funktion F(4) ergibt sich aus der Forderung $\dot{V}_2 = 0.2$ mit (18) für $\dot{V}_2 = x_2$. Zu beachten ist, daß alle Gleichungen im Unterprogramm in der Form F(J) = Ø geschrieben werden müssen.
Die Lösung der Aufgabe ergab mit den Eingaben DELTA = Ø.Ø1 und den Startwerten X(1) = 2, X(2) = 1, X(3) = 1.5, X(4) = 1 nach drei Iterationen die in Tafel 3 zusammengestellten Werte für den stationären Arbeitspunkt des Systems.

Tafel 3. Stationärer Arbeitspunkt

$\tilde{u} = 0.7865$; $\tilde{k}_{VA} = 1$; $p_G = 1.3982$; $p_1 = 1.1125$; $p_2 = 0.8$

1.4. Linearisierung des Zustandsmodells; Programm LIN

Viele Methoden zur Untersuchung der Systemdynamik basieren auf linearen Modellen. Die i. allg. nichtlinearen Funktionen f_i in (13) und g_j in (15) werden deshalb durch lineare ersetzt, die in der Umgebung eines vorgegebenen Arbeitspunktes mit

den Koordinaten $\mathbf{q_0}$, $\mathbf{u_0}$, $\mathbf{x_0}$ gelten. Meist wird als Arbeitspunkt der stationäre Punkt nach Abschn. 1.3. gewählt.
Die linearisierte Form der Gleichungen (13) lautet

$$\begin{aligned} \frac{dq_1}{dt} &= a_{11}q_1 + a_{12}q_2 + \ldots + a_{1n}q_n + b_{11}u_1 + \ldots + b_{1m}u_m \\ &\vdots \\ \frac{dq_n}{dt} &= a_{n1}q_1 + a_{n2}q_2 + \ldots + a_{nn}q_n + b_{n1}u_1 + \ldots + b_{nm}u_m \end{aligned} \tag{22}$$

und die der Gleichungen (15)

$$\begin{aligned} x_1 &= c_{11}q_1 + c_{12}q_2 + \ldots + c_{1n}q_n + d_{11}u_1 + \ldots + d_{1m}u_m \\ &\vdots \\ x_p &= c_{p1}q_1 + c_{p2}q_2 + \ldots + c_{pn}q_n + d_{p1}u_1 + \ldots + d_{pm}u_m \,. \end{aligned} \tag{23}$$

Faßt man die Koeffizienten a_{ij}, b_{ij}, c_{ij}, d_{ij} in (22) und (23) zu Matrizen **A, B, C, D** zusammen, dann läßt sich für (14) und (15) kurz schreiben:

Zustandsgleichung

$$\frac{d\mathbf{q}}{dt} = \mathbf{A\,q} + \mathbf{B\,u} \tag{24}$$

Ausgabegleichung

$$\mathbf{x} = \mathbf{C\,q} + \mathbf{D\,u}\,. \tag{25}$$

Die Veränderlichen q, u, x in (22) bis (25) stellen dabei die Abweichungen von den Arbeitspunkten dar. **A** heißt Systemmatrix, **B** Steuermatrix, **C** Beobachtungsmatrix, **D** Durchgangsmatrix. Die Formate der Matrizen sind aus (22) und (23) ablesbar. Im BASIC-Programm LIN werden die Matrizen durch die Felder A(N,N), B(N,M), C(P,N), D(P,M) repräsentiert. Die Elemente dieser Felder, das sind die Koeffizienten der linearisierten Gleichungssysteme (22), (23), berechnen sich nach (26) bis (29):

$$A(I,J) = \frac{\partial F(I)}{\partial Q(J)}; \quad I,J = 1,\ldots,N \tag{26}$$

$$B(I,J) = \frac{\partial F(I)}{\partial U(J)}; \quad I = 1,\ldots,N,\ J = 1,\ldots,M \tag{27}$$

$$C(I,J) = \frac{\partial G(I)}{\partial Q(J)}; \quad I = 1,\ldots,P,\ J = 1,\ldots,N \tag{28}$$

$$D(I,J) = \frac{\partial G(I)}{\partial U(J)}; \quad I = 1,\ldots,P,\ J = 1,\ldots,M\,. \tag{29}$$

Die Werte der Ableitungen sind dabei an den jeweiligen Arbeitspunkten zu bilden. Die Gln. (26) bis (29) entsprechen völlig (21). Die Ableitungen werden numerisch deshalb genau so ermittelt, wie es im Abschn. 1.3.2. im Anschluß an (21) beschrieben worden ist.

Programm LIN

```
1Ø  REM BERECHNUNG ZUSTANDSMATRIZEN A, B, C, D
2Ø  INPUT "N, M, P "; N,M,P : DEFFN Q(X)=X * X:PRINT
3Ø  DIM A(N,N), B(N,M), C(P,N), D(P,M), F(N + P), H(N + P), V(N + M)
4Ø  INPUT "DELTA "; DEL : PRINT : PRINT
5Ø  PRINT "ARBEITSPUNKTKOORDINATEN 1 ..."; N + M
6Ø  FOR I = 1 TO N + M: INPUT V(I): NEXT I:CLS
7Ø  FOR I = 1 TO N + M: LET V(I) = V(I) + DEL: GOSUB 5ØØ
8Ø  FOR J = 1 TO N + P: LET H(J) = F(J): NEXT J
9Ø  LET V(I) = V(I) - 2 * DEL: GOSUB 5ØØ
1ØØ FOR J = 1 TO N + P: IF I > N GOTO 13Ø
11Ø IF J > N GOTO 15Ø
12Ø LET A(J,I) = (H(J) - F(J))/DEL/2: GOTO 17Ø
13Ø IF J > N GOTO 16Ø
14Ø LET B(J,I - N) = (H(J) - F(J))/DEL/2: GOTO 17Ø
15Ø LET C(J - N,I) = (H(J) - F(J))/DEL/2: GOTO 17Ø
16Ø LET D(J - N,I - N) = (H(J) - F(J))/DEL/2
17Ø NEXT J: LET V(I) = V(I) + DEL: NEXT I
18Ø PRINT "ZUSTANDSMATRIX A": PRINT
19Ø FOR I = 1 TO N: FOR J = 1 TO N: PRINT TAB(11 * (J - 1));A(I,J);
2ØØ NEXT J: PRINT: NEXT I
21Ø PRINT "STEUERMATRIX B": PRINT
22Ø FOR I = 1 TO N: FOR J = 1 TO M: PRINT TAB(11 * (J - 1));B(I,J);
23Ø NEXT J: PRINT: NEXT I: PAUSE (Ø): CLS
24Ø PRINT: PRINT: PRINT "BEOBACHTUNGSMATRIX C": PRINT
25Ø FOR I = 1 TO P: FOR J = 1 TO N: PRINT TAB(11 * (J - 1));C(I,J);
26Ø NEXT J: PRINT: NEXT I
27Ø PRINT: PRINT: PRINT "DURCHGANGSMATRIX D": PRINT
28Ø FOR I = 1 TO P: FOR J = 1 TO M: PRINT TAB(11 * (J - 1));D(I,J);
29Ø NEXT J: PRINT: NEXT I: END
5ØØ REM PARAMETERUEBERGABE
51Ø LET PG = V(1): LET P1 = V(2): LET P(2) = V(3): LET U = V(4): LET KVA = V(5)
52Ø REM RECHTE SEITEN DER ZUSTANDS- UND AUSGABEGLEICHUNGEN
53Ø LET F(1) = Ø.168 * FN Q(U) - Ø.Ø8 * PG + Ø.ØØ8
54Ø LET F(2) = Ø.25 * (Ø.7 * PG - Ø.7 * P1 - Ø.4 * SQR((P1 - P2) * P2))
55Ø LET F(3) = Ø.333 * (Ø.4 * SQR((P1 - P2) * P2 - Ø.25 * KVA * P2)
56Ø LET F(4) = P2
57Ø LET F(5) = Ø.4 * SQR((P1 - P2) * P2)
58Ø RETURN
```

Das Programm LIN besteht wie alle bisher behandelten Programme aus einem allgemeingültigen Teil (Zeilen 1Ø bis 29Ø) und aus dem problemspezifischen jeweils neu zu programmierenden Unterprogramm, das die Funktionen f_i und g_i des jeweils zu linearisierenden Zustandsmodells darstellt (Zeilen 52Ø bis 58Ø). Diesen vorangestellt ist die Zeile 51Ø, die die allgemeingültigen Namen der Programmvariablen aus dem Hauptprogramm mit den technologischen Variablennamen im Unterprogramm verknüpft. Im Hauptprogramm wird nicht zwischen Zustandsgrößen q und Steuergrößen u unterschieden; beide werden vielmehr zu einem Feld V(N + M) zusammengefaßt. Die ersten N Elemente dieses Feldes repräsentieren die Zustandsgrößen, die restlichen M die Steuergrößen (vgl. Zeile 51Ø).
Ebenso werden die rechten Seiten von (13) und (15) zu einem Funktionenvektor

F(N + P) zusammengefaßt, dessen erste N Gleichungen die Funktionen f_i und die restlichen P die Funktionen g_i darstellen.
Im Programm LIN wurden die Zeilen 5ØØ bis 58Ø für das Beispielsystem mit N = 3, M = 2, P = 2 geschrieben. F(1) bis F(3) sind deshalb die rechten Seiten von (10) bis (12a); F(4) und F(5) entsprechen (17) und (18).
Im Programm werden nacheinander alle (N + P) * (N + M) Ableitungen gebildet (Zeilen 7Ø bis 9Ø). Durch entsprechende Indexverschiebungen werden die berechneten Werte in die Felder A, B, C, D eingeordnet (Zeilen 12Ø bis 16Ø).
Bei den Eingaben wird zuerst ein Wert für Delta verlangt, dann die Koordinaten des Arbeitspunkts in der Reihenfolge der technologischen Variablen, wie sie aus Zeile 51Ø ablesbar ist.
Für das Beispiel ergaben sich mit Delta = Ø.Ø1 und den Arbeitspunktkoordinaten nach Tafel 3 die in Tafel 4 zusammengestellten Matrizen. Gegenüber der Rechnerausgabe wurden die Zahlenwerte auf 4 Stellen hinter dem Dezimalpunkt gerundet.

Tafel 4. Matrizen des im stationären Arbeitspunkt linearisierten Zustandsmodells

$$\mathbf{A} = \begin{pmatrix} -0.08 & 0 & 0 \\ 0.175 & -0.255 & 0.04876 \\ 0 & 0.1066 & -0.1482 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} 0.2643 & 0 \\ 0 & 0 \\ 0 & -0.066 \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0.32 & -0.195 \end{pmatrix} \qquad \mathbf{D} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

1.5. Zustandsmodell des geschlossenen Regelkreises

Oftmals bestehen komplizierte Systeme aus Teilsystemen, die jeweils durch ihre Zustandsmodelle beschrieben werden. Dann kann für das Gesamtsystem ein einziges Zustandsmodell konstruiert werden, das die Teilmodelle zusammenfaßt.
Dieser Fall liegt beispielsweise beim Regelkreis nach Bild 9 vor. Für die Regelstrecke existiere ein Zustandsmodell, das mit dem des Reglers zu einem Gesamtmodell zusammengebaut werden kann. Aus dem Gesamtmodell kann das dynamische Verhalten des Regelkreises berechnet werden (s. Abschn. 3.2.).
Für den Regler kann ein allgemeines Zustandsmodell angesetzt werden. Hier sollen aber nur die üblichen PID-Regler betrachtet werden. Hinsichtlich der Einsatzmöglichkeiten eines D-Anteils gibt es Einschränkungen zu beachten.

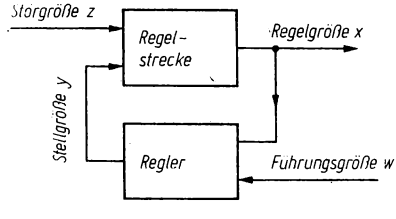


Bild 9. Blockdiagramm eines Regelkreises

Zustandsmodell der Regelstrecke

Die Regelstrecke wird beschrieben durch einen Zustandsvektor $\mathbf{q}$ (n Komponenten), ihre Ausgangsgröße x (Regelgröße) und die beiden Eingangsgrößen y (Stellgröße) und z (Störgröße).
Im allgemeinen linearen Zustandsmodell (24) wird die Steuermatrix $\mathbf{B}$ deshalb aufgeteilt in die beiden Spaltenvektoren $\mathbf{b}_y$ und $\mathbf{b}_z$, die zu den Eingangsgrößen y und z gehören:

$$\frac{d\mathbf{q}}{dt} = \mathbf{A}\,\mathbf{q} + \mathbf{b}_y\,y + \mathbf{b}_z\,z\,. \tag{30}$$

Es existiert eine einzige Ausgabegleichung (25) für die Ausgangsgröße x. Aus der Beobachtungsmatrix $\mathbf{C}$ wird deshalb ein Zeilenvektor $\mathbf{c}^T$; aus der Durchgangsmatrix $\mathbf{D}$ werden die beiden Durchgangskoeffizienten d_y und d_z. Damit erhält die Ausgabegleichung die Form

$$x = \mathbf{c}^T\,\mathbf{q} + d_y\,y + d_z\,z\,. \tag{31}$$

Sind die Koeffizienten d_y, d_z von null verschieden, dann ist das System *sprungfähig*, d. h., ein Sprung von y oder z tritt ebenfalls als Sprung am Ausgang x in Erscheinung. In diesem Fall ist ein D-Anteil im Regler nicht angebracht.
Ist $d_y = d_z = 0$, dann zeigt die Regelstrecke reines Verzögerungsverhalten, und ein D-Anteil kann vorteilhaft sein.

1.5.1. Sprungfähiges System; Regler ohne D-Anteil

Der Regler (Bild 9) hat die beiden Eingangsgrößen Istwert x und Führungsgröße w. Sein Ausgangssignal ist y.
Ein PI-Regler wird durch ein Zustandsmodell mit einer Zustandsgröße q_R beschrieben:

$$\frac{dq_R}{dt} = x - w\,. \tag{32}$$

Seine Ausgabegleichung lautet

$$y = -K_P x - K_I q_R + K_P w\,. \tag{33}$$

K_P, K_I sind die Reglerparameter. Für einen reinen P-Regler entfällt (32), und in (33) wird $K_I = 0$.
Durch Einsetzen von x gemäß (31) in (33) erhält man

$$y = -K_P(\mathbf{c}^T\mathbf{q} + d_y y + d_z z) - K_I q_R + K_P w$$

bzw.

$$y = -K_P c_0 \mathbf{c}^T\mathbf{q} - K_I c_0 q_R - K_P c_0 d_z z + K_P c_0 w\,. \tag{34}$$

Dabei gilt die Abkürzung $c_0 = 1/(1 + K_P d_y)$.
Setzt man y gemäß (34) in (30) ein, dann ergibt sich

$$\frac{d\mathbf{q}}{dt} = (\mathbf{A} - K_P c_0 \mathbf{b}_y \mathbf{c}^T)\,\mathbf{q} - K_I c_0 \mathbf{b}_y q_R + K_P c_0 \mathbf{b}_y w + (\mathbf{b}_z - K_P c_0 d_z \mathbf{b}_y)\,z\,. \tag{35}$$

Durch Einsetzen von x in die Zustandsgleichung des Reglers (32) erhält man

$$\frac{dq_R}{dt} = (1 - K_P c_0 d_y)\, \mathbf{c}^T \mathbf{q} - K_I c_0 d_y q_R + (K_P c_0 d_y - 1)\, w + (1 - K_P c_0 d_y)\, z\,. \quad (36)$$

Schließlich kann man noch y – s. Gl. (33) – in die Ausgabegleichung (31) einsetzen und bekommt

$$x = c_0 \mathbf{c}^T \mathbf{q} - K_I c_0 d_y q_R + K_P c_0 d_y w + c_0 d_z z\,. \quad (37)$$

Wegen der Gültigkeit der Beziehung $1 - K_P c_0 d_y = c_0$ lassen sich (36) und (37) noch vereinfachen. Man erhält schließlich

$$\frac{d\mathbf{q}}{dt} = (\mathbf{A} - K_P c_0 \mathbf{b}_y \mathbf{c}^T)\, \mathbf{q} - K_I c_0 \mathbf{b}_y q_R + K_P c_0 \mathbf{b}_y\, w + (\mathbf{b}_z - K_P c_0 d_z \mathbf{b}_y)\, z$$

$$\frac{dq_R}{dt} = c_0 \mathbf{c}^T \mathbf{q} - K_I c_0 d_y q_R - c_0\, w + c_0 d_z\, z \quad (38)$$

$$x = c_0 \mathbf{c}^T \mathbf{q} - K_I c_0 d_y q_R + (1 - c_0)\, w + c_0 d_z\, z\,. \quad (39)$$

Die beiden Gleichungen (38) sind das Zustandsmodell für den Regelkreis mit dem Regelkreiszustandsvektor

$$\mathbf{q}^* = \begin{pmatrix} \mathbf{q} \\ q_R \end{pmatrix},$$

der die Dimension $n + 1$ hat. Die Ausgangsgröße ist die Regelgröße x; die Regelkreiseingangsgrößen sind die Führungsgröße w und die Störgröße z. Gl. (39) ist die Ausgabegleichung für den Regelkreis.
Für die Matrizen des Regelkreises läßt sich aus (38) und (39) ablesen:

$$\mathbf{A}^* = \begin{pmatrix} \mathbf{A} - K_P c_0 \mathbf{b}_y \mathbf{c}^T & -K_I c_0 \mathbf{b}_y \\ c_0 \mathbf{c}^T & -K_I c_0 d_y \end{pmatrix} \quad (40)$$

$$\mathbf{b}_w^* = \begin{pmatrix} K_P c_0 \mathbf{b}_y \\ -c_0 \end{pmatrix}, \qquad \mathbf{b}_z^* = \begin{pmatrix} \mathbf{b}_z - K_P c_0 d_z \mathbf{b}_y \\ c_0 d_z \end{pmatrix} \quad (41)$$

$$\mathbf{c}^{*T} = (c_0 \mathbf{c}^T \quad -K_I c_0 d_y) \quad (42)$$

$$d_w^* = 1 - c_0\,, \quad d_z^* = c_0 d_z \quad (43)$$

1.5.2. Regler mit D-Anteil

Bedingung ist $d_y = d_z = 0$. Damit lautet die Ausgabegleichung der Regelstrecke einfach

$$x = \mathbf{c}^T \mathbf{q}\,, \quad (44)$$

während die Zustandsgleichung durch (30) gegeben ist. Für einen PID-Regler gilt

ebenfalls die Reglerzustandsgleichung (32), während der D-Anteil in seiner Ausgabegleichung zum Ausdruck kommt:

$$y = -K_P x - K_I q_R - K_D \dot{x} + K_P w \,. \tag{45}$$

Das D-Verhalten soll sich nur auf die Regelgröße, nicht dagegen auf den Sollwert erstrecken, was sonst bei Führungsgrößensprüngen zu Schwierigkeiten führen könnte.

Für $\dot{x}$ erhält man mit (44)

$$\dot{x} = \frac{dx}{dt} = \mathbf{c}^T \frac{d\mathbf{q}}{dt} \,.$$

Setzt man diesen Ausdruck in (45) und die damit erhaltene Beziehung für y wiederum in (30) ein, dann erhält man

$$\frac{d\mathbf{q}}{dt} = \mathbf{A}\mathbf{q} + b_y \left(- K_P \mathbf{c}^T \mathbf{q} - K_I q_R - K_D \mathbf{c}^T \frac{d\mathbf{q}}{dt} \right) + K_P \mathbf{b}_y w + \mathbf{b}_z z \,.$$

Bringt man das Glied mit $^{dq}/_{dt}$ auf die linke Seite und klammert aus, dann bekommt man mit E für die Einheitsmatrix

$$(\mathbf{E} + K_D \mathbf{b}_y \mathbf{c}^T) \frac{d\mathbf{q}}{dt} = (\mathbf{A} - K_P \mathbf{b}_y \mathbf{c}^T)\mathbf{q} - K_I \mathbf{b}_y q_R + K_P \mathbf{b}_y w + \mathbf{b}_z z$$

$$\frac{dq_R}{dt} = \mathbf{c}^T \mathbf{q} - w \,. \tag{46}$$

Mit der Matrix

$$\mathbf{M} = \begin{pmatrix} \mathbf{E} + K_D \mathbf{b}_y \mathbf{c}^T & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix} \tag{47}$$

läßt sich das System (46) in folgender Form schreiben:

$$\mathbf{M} \begin{pmatrix} \frac{d\mathbf{q}}{dt} \\ \frac{dq_R}{dt} \end{pmatrix} = \begin{pmatrix} \mathbf{A} - K_P \mathbf{b}_y \mathbf{c}^T & -K_I \mathbf{b}_y \\ \mathbf{c}^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ q_R \end{pmatrix} + \begin{pmatrix} K_P \mathbf{b}_y \\ -1 \end{pmatrix} w + \begin{pmatrix} \mathbf{b}_z \\ 0 \end{pmatrix} z \,.$$

Werden alle Glieder dieser Gleichung von links mit der Inversen $\mathbf{M}^{-1}$ multipliziert, dann ergibt sich auf der linken Seite

$$\begin{pmatrix} \frac{d\mathbf{q}}{dt} \\ \frac{dq_R}{dt} \end{pmatrix} = \frac{d\mathbf{q}^*}{dt} \,,$$

und man erkennt die Bildungsgesetze für die Matrizen des Regelkreises mit dem Zustandsmodell

$$\begin{aligned} \frac{d\mathbf{q}^*}{dt} &= \mathbf{A}^* \mathbf{q}^* + \mathbf{b}_w^* w + \mathbf{b}_z^* z \\ x &= \mathbf{c}^{*T} \mathbf{q}^* \end{aligned} \tag{48}$$

$$\mathbf{A}^* = \mathbf{M}^{-1}\begin{pmatrix} \mathbf{A} - \mathbf{K}_P\mathbf{b}_y\mathbf{c}^T - \mathbf{K}_I\mathbf{b}_y \\ \mathbf{c}^T \qquad\qquad 0 \end{pmatrix} \tag{49}$$

$$\mathbf{b}_w^* = \mathbf{M}^{-1}\begin{pmatrix} \mathbf{K}_P\mathbf{b}_y \\ -1 \end{pmatrix}, \quad \mathbf{b}_z^* = \mathbf{M}^{-1}\begin{pmatrix} \mathbf{b}_z \\ 0 \end{pmatrix} \tag{50}$$

$$\mathbf{c}^{*T} = (\mathbf{c}^T \quad 0)\,. \tag{51}$$

1.5.3. Rechenprogramm KRSMAT

Die in den Abschnitten 1.5.1. und 1.5.2. behandelten Fälle schließen sich gegenseitig aus. Daher kann man die Beziehungen zur Berechnung der Matrizen auch zusammenfassen
– aus (40) und (49)

$$\mathbf{A}^* = \mathbf{M}^{-1}\begin{pmatrix} \mathbf{A} - c_0\mathbf{K}_P\mathbf{b}_y\mathbf{c}^T - c_0\mathbf{K}_I\mathbf{b}_y \\ c_0\mathbf{c}^T \qquad\qquad - c_0\mathbf{K}_I d_y \end{pmatrix} \tag{52}$$

– aus (41) und (50)

$$\mathbf{b}_w^* = \mathbf{M}^{-1}\begin{pmatrix} c_0\mathbf{K}_P\mathbf{b}_y \\ -c_0 \end{pmatrix}, \quad \mathbf{b}_z^* = \mathbf{M}^{-1}\begin{pmatrix} \mathbf{b}_z - c_0\mathbf{K}_P d_z\mathbf{b}_y \\ c_0 d_z \end{pmatrix} \tag{53}$$

– aus (42) und (51)

$$\mathbf{c}^{*T} = (c_0\mathbf{c}^T \quad -c_0\mathbf{K}_I d_y)\,. \tag{54}$$

Schließlich gilt

$$d_w^* = 1 - c_0\,, \quad d_z^* = c_0 d_z\,.$$

Dabei gilt wechselseitig entweder

$$d_y, d_z \not\equiv 0, \quad \mathbf{K}_D = 0, \quad \mathbf{M} = \mathbf{E} \qquad \text{oder}$$
$$d_y = d_z = 0, \quad \mathbf{K}_D \not\equiv 0, \quad c_0 = 1\,.$$

Programm KRSMAT

```
1Ø  REM BERECHNUNG DER MATRIZEN DES GESCHLOSSENEN REGELKREISES
2Ø  REM STRECKENEINGANGSGROESSE 1: STELLGROESSE, 2: STOERGROESSE
3Ø  REM KREISEINGANGSGROESSE 1: FUEHRUNGSGROESSE, 2: STOER-
    GROESSE
4Ø  INPUT "DIMENSION N"; N: LET KD = Ø
5Ø  DIM AE(N,N), BE(N,2), CE(N), DE(2), A(N + 1,N + 1), B(N + 1,2), C(N + 1), D(2)
6Ø  DIM E(N + 1, N + 1), I(N + 1,N + 1), R(N + 1), M(N + 1,N + 1)
7Ø  PRINT "EINGABEN": PRINT
8Ø  FOR I = 1 TO N: FOR J = 1 TO N:PRINT"A(";I;J;")";:INPUT AE(I,J)
9Ø  NEXT J: NEXT I: PRINT
1ØØ FOR I = 1 TO N: FOR J = 1 TO 2: PRINT "B(";I;J;")";: INPUT BE(I,J)
11Ø NEXT J: NEXT I: PRINT
12Ø FOR I = 1 TO N: PRINT "CT";I;: INPUT CE(I): NEXT I: PRINT
13Ø FOR I = 1 TO 2: PRINT "D";I;: INPUT DE(I): NEXT I: CLS
```

```
14Ø  IF DE(1) = Ø AND DE(2) = Ø THEN LET SD = 1
15Ø  PRINT "REGLERPARAMETER": INPUT "KP";KP: INPUT "KI";KI
16Ø  REM D-ANTEIL NUR MOEGLICH, WENN DURCHGANGSMATRIX = Ø
17Ø  IF SD THEN INPUT "KD";KD
18Ø  LET CO = 1/(1 + KP * DE(1))
19Ø  FOR I = 1 TO N: FOR J = 1 TO N: LET A(I,J) = AE(I,J) - KP * CO * BE(I,1) * CE(J):
     NEXT J
2ØØ  LET A(I,N + 1) = -KI * CO * BE(I,1): LET A(N + 1,I) = CO * CE(I)
21Ø  LET B(I,1) = KP * CO * BE(I,1): LET B(I,2) = BE(I,2) - KP * CO * DE(2)
22Ø  LET C(I) = CO * CE(I): NEXT I
23Ø  LET A(N + 1,N + 1) = -KI * CO * DE(1): LET B(N + 1,1) = -CO: LET B(N + 1,2)
     = CO * DE(2)
24Ø  LET C(N + 1) = -KI * CO * DE(1): LET D(1) = KP * CO * DE(1): LET D(2)
     = CO * DE(2)
25Ø  IF KD = Ø GOTO 47Ø
26Ø  FOR I = 1 TO N: LET E(I,I) = 1: FOR J = 1 TO N
27Ø  LET M(I,J) = E(I,J) + KD * BE(I,1) * CE(J): NEXT J
28Ø  LET M(I,N + 1) = Ø: LET M(N + 1,I) = Ø: NEXT I
29Ø  LET M(N + 1,N + 1) = 1: LET E(N + 1,N + 1) = 1
3ØØ  REM INVERSION
31Ø  FOR K = 1 TO N: FOR I = K + 1 TO N + 1: IF M(I,K) = Ø GOTO 35Ø
32Ø  LET R(I) = M(K,K)/M(I,K)
33Ø  FOR L = 1 TO N + 1: LET E(I,L) = E(K,L) - R(I) * E(I,L): NEXT L
34Ø  FOR J = K TO N + 1: LET M(I,J) = M(K,J) - R(I) * M(I,J): NEXT J
35Ø  NEXT I: NEXT K
36Ø  FOR L = 1 TO N + 1: LET I(N + 1,L) = E(N + 1,L)/M(N + 1,N + 1)
37Ø  FOR J = N TO 1 STEP -1: LET S = Ø: FOR I = J + 1 TO N + 1
38Ø  LET S = S + M(J,I) * I(I,L): NEXT I
39Ø  LET I(J,L) = (E(J,L) - S)/M(J,J): NEXT J: NEXT L
4ØØ  FOR I = 1 TO N + 1: FOR J = 1 TO N + 1: LET M(I,J) = Ø
41Ø  FOR L = 1 TO N + 1: LET M(I,J) = M(I,J) + I(I,L) * A(L,J): NEXT L
42Ø  NEXT J: NEXT I: FOR I = 1 TO N + 1: FOR J = 1 TO N + 1
43Ø  LET A(I,J) = M(I,J): NEXT J: NEXT I
44Ø  FOR I = 1 TO N + 1: FOR J = 1 TO 2: LET M(I,J) = Ø: FOR L = 1 TO N + 1
45Ø  LET M(I,J) = M(I,J) + I(I,L) * B(L,J): NEXT L: NEXT J: NEXT I
46Ø  FOR I = 1 TO N + 1: FOR J = 1 TO 2: LET B(I,J) = M(I,J): NEXT J: NEXT I
47Ø  CLS: PRINT "ZUSTANDSMATRIX A *": PRINT
48Ø  FOR I = 1 TO N + 1: FOR J = 1 TO N + 1: PRINT TAB(11 * (J - 1));A(I,J);
49Ø  NEXT J: PRINT: NEXT I: PRINT: PAUSE(5Ø)
5ØØ  PRINT "STEUERMATRIX B *": PRINT
51Ø  FOR I = 1 TO N + 1: FOR J = 1 TO 2: PRINT TAB(11 * (J - 1));B(I,J);
52Ø  NEXT J: PRINT: NEXT I: PRINT: PAUSE (5Ø)
53Ø  PRINT "BEOBACHTUNGSVEKTOR CT *": PRINT
54Ø  FOR I = 1 TO N + 1: PRINT TAB(11 * (I - 1));C(I);: NEXT I: PRINT
55Ø  IF SD THEN PRINT "DURCHGANGSMATRIX = Ø": END
56Ø  PRINT "DURCHGANGSVEKTOR DT *": PRINT
57Ø  FOR I = 1 TO 2: PRINT TAB(11 * (I - 1));D(I);: NEXT I: END
```

Im Programm KRSMAT wird von den Beziehungen (47), (52) bis (54) zur Berechnung der Matrizen $\mathbf{A}^*$ (Programmname A(N + 1,N + 1)), $\mathbf{B}^* = (\mathbf{b}_w^* \; \mathbf{b}_z^*)$ (Programmname B(N + 1,2)), $\mathbf{c}^{*T}$ (Programmname C(N + 1)) und $\mathbf{d}^{*T} = (d_w^* \; d_z^*)$ (Programmname D(2)) ausgegangen. Die Berechnung geschieht in den Programmzeilen 18Ø

bis 24Ø. Für Regler ohne I-Anteil benötigt die Matrix **A*** nur n Zeilen und Spalten; die Ordnung des Systems erhöht sich nicht. Auf diese Fallunterscheidung wurde jedoch verzichtet. Wird das Programm KRSMAT auf Regelkreise mit Reglern ohne I-Anteile angewendet, dann besteht die letzte Spalte von **A*** nur aus Nullelementen. Die in Zeile 8Ø einzugebenden Matrizen des Streckenmodells tragen die Programmnamen AE, BE, CE, DE. Durch die Fallunterscheidung $d_y = d_z = 0$ wird die Eingabe eines D-Anteils in die Reglergleichung nur dann ermöglicht, wenn diese Frage positiv beantwortet wurde (Zeilen 14Ø, 17Ø).
Die in den Zeilen 18Ø bis 24Ø u. a. berechneten Matrizen **A*** und **B*** müssen noch mit der Matrix $\mathbf{M}^{-1}$ multipliziert werden. Setzt man bei der Berechnung von **M** (Programmname M(N + 1,N + 1)) in Zeilen 26Ø bis 29Ø nach (47) $K_D = 0$, dann erhält man die Einheitsmatrix, deren Inversion in den Zeilen 3ØØ bis 39Ø wiederum die Einheitsmatrix ergibt. (Das Inversionsprogramm ist gleich dem im Programm NEWT verwendeten.) Die Inverse hat den Programmnamen I(N + 1,N + 1). Da Berechnung von **M** und Inversion jedoch bei $K_D = 0$ unnötig sind, werden die entsprechenden Programmzeilen durch die Fallunterscheidung in Zeile 25Ø ggf. übergangen. Ansonsten erfolgt in den Zeilen 4ØØ bis 46Ø die Multiplikation mit der Inversen **I**.
Die Anwendung des Programms KRSMAT auf das Beispiel mit den Systemmatrizen nach Tafel 4 ergab die in Tafel 5 zusammengestellten Ergebnisse.

Tafel 5. Matrizen des geschlossenen Regelkreises

a) P-Regler, $K_P = 8$

$$\mathbf{A}^* = \begin{pmatrix} -0.08 & -0.6766 & -0.4123 & 0 \\ 0.175 & -0.255 & 0.04876 & 0 \\ 0 & 0.1066 & -0.1482 & 0 \\ 0 & 0.32 & -0.195 & 0 \end{pmatrix} \qquad \mathbf{B}^* = \begin{pmatrix} 2.1144 & 0 \\ 0 & 0 \\ 0 & -0.066 \\ -1 & 0 \end{pmatrix}$$

$$\mathbf{c}^{T*} = (0 \quad 0.32 \quad -0.195 \quad 0) \qquad \mathbf{d}^* = 0$$

b) PI-Regler, $K_P = 7$, $K_I = 0.75$

$$\mathbf{A}^* = \begin{pmatrix} -0.08 & -0.592 & 0.3608 & -0.1982 \\ 0.175 & -0.255 & 0.04876 & 0 \\ 0 & 0.1066 & -0.1482 & 0 \\ 0 & 0.32 & -0.195 & 0 \end{pmatrix} \qquad \mathbf{B}^* = \begin{pmatrix} 1.8501 & 0 \\ 0 & 0 \\ 0 & -0.06 \\ -1 & 0 \end{pmatrix}$$

$$\mathbf{c}^{T*} = (0 \quad 0.32 \quad -0.195 \quad 0) \qquad \mathbf{d}^* = 0$$

1.6. Rauschvorgänge

Stochastische, vom Zufall abhängende Einflußgrößen eines dynamischen Systems kommen sehr oft vor. Zustandsmodelle geben eine günstige Möglichkeit, solche Signale zu beschreiben. Ausgegangen wird von der Modellvorstellung eines „weißen

Rauschens" w. Das ist ein Rauschprozeß, bei dem zwischen zwei zeitlich beliebig eng benachbarten Signalwerten absolut kein Zusammenhang, keinerlei Korrelation existiert. Physikalisch realisierbar ist ein solcher Prozeß im kontinuierlichen Fall allerdings nicht, wohl aber im diskontinuierlichen (s. Abschn. 3.2.2.). Von Interesse ist aber vor allem das sog. „farbige Rauschen", das aus dem weißen Rauschen durch Signalfilterung hervorgebracht werden kann. Als einfachste Signalfilter kommen die aus der Regelungstechnik bekannten Verzögerungsglieder 1. und höherer Ordnung, aber auch Bandpässe und andere Filter in Betracht. Für jedes dieser Filter kann eine Zustandsgleichung geschrieben werden mit $\mathbf{q}_R$ als dem Zustandsvektor, w, dem weißen Rauschsignal, als der Eingangsgröße und x_R, dem gefilterten Rauschsignal, als der Ausgangsgröße (55).

$$\frac{d\mathbf{q}_R}{dt} = \mathbf{A}_R\,\mathbf{q}_R + \mathbf{b}_R\,w$$
$$x_R = \mathbf{c}_R^T\,\mathbf{q}_R \qquad (55)$$

Tafel 6 gibt für einige einfachste Rauschfilter die Werte der Modellparameter.

Tafel 6. Modelldaten für Rauschfilter

Filtertyp	$\mathbf{A}_R$	$\mathbf{b}_R$	$\mathbf{c}_R^T$
T_1 Zeitkonstante T	$-1/T$	$\sqrt{\frac{2}{T}}$	1
T_2 Zeitkonstanten T	$\begin{pmatrix} -1/T & 0 \\ 1/T & -1/T \end{pmatrix}$	$\begin{pmatrix} \frac{2}{\sqrt{T}} \\ 0 \end{pmatrix}$	(0 1)
Schmalband Mittenfrequenz ω_0 Bandbreite $B\omega_0$	$\begin{pmatrix} 0 & 1 \\ -\omega_0^2 & -2B\omega_0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2\sqrt{B\omega_0} \end{pmatrix}$	(0 1)
Tiefpaß Grenzfrequenz ω_0 $D_1 = 0.9239$ $D_2 = 0.3827$	$\begin{pmatrix} 0 & 1 & 0 & 0 \\ -\omega_0^2 & -2D_1\omega_0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & -\omega_0^2 & -2D_2\omega_0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2.378\,\omega_0^3 \\ 0 \\ 0 \end{pmatrix}$	(0 0 1 0)

Soll die Reaktion eines dynamischen Systems mit dem Zustandsmodell (24), (25) auf dieses Rauschsignal x_R untersucht werden, dann ist für u, das Eingangssignal des Systems, das Rauschsignal x_R zu setzen. Da das System nur unter der Wirkung des Eingangssignals u untersucht werden soll, tritt der Steuervektor $\mathbf{b}_u$ an die Stelle der Matrix **B**.
Die Gl. (24) wird noch einmal abgeschrieben. Darunter wird die Zustandsgleichung für das Rauschfilter (55) gesetzt:

$$\frac{d\mathbf{q}}{dt} = \mathbf{A}\,\mathbf{q} + \mathbf{b}_u\,u = \mathbf{A}\,\mathbf{q} + \mathbf{b}_u\,x_R$$

bzw.

$$\frac{d\mathbf{q}}{dt} = \mathbf{A}\,\mathbf{q} + \mathbf{b}_u\,\mathbf{c}_R^T\,\mathbf{q}_R$$

$$\frac{d\mathbf{q}_R}{dt} = \mathbf{A}_R\,\mathbf{q}_R + \mathbf{b}_R\,w\,. \tag{55}$$

Aus der Ausgabegleichung (25)

$$\mathbf{x} = \mathbf{C}\,\mathbf{q} + \mathbf{d}\,u$$

erhält man dann

$$\mathbf{x} = \mathbf{C}\,\mathbf{q} + \mathbf{d}\,\mathbf{c}_R^T\,\mathbf{q}_R\,.$$

Systemzustandsvektor $\mathbf{q}$ und Rauschfiltervektor $\mathbf{q}_R$ können zu einem einzigen Vektor $\mathbf{q}^{*T} = (\mathbf{q}^T\,\mathbf{q}_R^T)$ zusammengefaßt werden. Damit läßt sich das Gesamtmodell schreiben:

$$\frac{d\mathbf{q}^*}{dt} = \mathbf{A}^*\,\mathbf{q}^* + \mathbf{b}^*\,w$$

$$\mathbf{x} = \mathbf{C}^*\,\mathbf{q}^*\,.$$

Die Matrizen des Gesamtmodells setzen sich aus denen des Systemmodells und des Filtermodells zusammen:

$$\mathbf{A}^* = \begin{pmatrix} \mathbf{A} & \mathbf{b}_u\,\mathbf{c}_R^T \\ \mathbf{0} & \mathbf{A}_R \end{pmatrix},\quad \mathbf{b}^* = \begin{pmatrix} \mathbf{0} \\ \mathbf{b}_R \end{pmatrix}$$

$$\mathbf{C}^* = \begin{pmatrix} \mathbf{C} & \mathbf{d}\mathbf{c}_R^T \end{pmatrix}. \tag{56}$$

Wegen der einfachen Form der Rauschfiltermatrizen (s. Tafel 6) können die Matrizen nach (56) ohne Schwierigkeiten auch von Hand berechnet werden, so daß sich ein besonderes Rechenprogramm dafür erübrigt.

2. Kontinuierliche Klemmenmodelle

2.1. Herleitung aus dem Zustandsmodell

2.1.1. Klemmenmodell

Ein System mit einer Eingangsgröße u sowie einer Ausgangsgröße x kann n innere Zustandsgrößen $q_1 \ldots q_n$ aufweisen (Bild 10). Das Zustandsmodell (s. Abschnitte 1.1.3. und 1.1.4.) stellt den Zusammenhang zwischen diesen Größen her. Die Zustandsgrößen q_i sind innere Größen des Systems. Nach außen, d. h. an den „Klemmen" des Systems, treten nur die Eingangsgröße und die Ausgangsgröße in Erscheinung. Ein Modell, das nur diese beiden Größen verknüpft, mithin keine Zustandsgrößen enthält, heißt Klemmenmodell.

Bild 10. Allgemeines System

Hat ein System m Eingangsgrößen und p Ausgangsgrößen, dann lassen sich mp Klemmenmodelle von jeder Eingangsgröße zu jeder der Ausgangsgrößen definieren.
Betrachtet werden sollen nur lineare, ggf. linearisierte Systeme. Die Klemmenmodelle können dann als lineare gewöhnliche Differentialgleichungen oder als Übertragungsfunktionen G(p) formuliert werden, die als unabhängige Variable den Differentiationsoperator p enthalten [7; 8].
Die allgemeine Form einer Übertragungsfunktion lautet

$$G(p) = \frac{b_0 + b_1 p + \ldots + b_n p^n}{a_0 + a_1 p + \ldots + a_n p^n} . \tag{57}$$

Mit dem Zustandsmodell bestehen folgende allgemeine Zusammenhänge: Der Grad n des Nennerpolynoms in (57) ist gleich der Anzahl der Zustandsgrößen (Dimension des Zustandsvektors $\mathbf{q}$). Der Koeffizient b_n im Zähler ist nur dann von null verschieden, wenn der Durchgangskoeffizient d im Zustandsmodell von null verschieden ist. Dann ist das System sprungfähig (s. Abschn. 1.5.1.).
Im einzelnen kann das Klemmenmodell aus der linearen Zustandsgleichung (24) und der Ausgabegleichung (25) hergeleitet werden. Da nur eine einzige Eingangsgröße u und Ausgangsgröße x auftreten sollen, lauten diese Gleichungen jetzt

$$\frac{d\mathbf{q}}{dt} = \mathbf{A}\,\mathbf{q} + \mathbf{b}\,u$$

$$x = \mathbf{c}^T\mathbf{q} + d\,u .$$

Laplace-Transformation dieser beiden Gleichungen ergibt, wenn der Anfangszustand $\mathbf{q}_0 = \mathbf{0}$ gesetzt wird, die Gln. (58) und (59), in denen die Laplace-Transformierten durch Großbuchstaben bezeichnet werden:

$$p\,\mathbf{Q}(p) = \mathbf{A}\,\mathbf{Q}(p) + \mathbf{b}\,U(p) \tag{58}$$

$$X(p) = \mathbf{c}^T\mathbf{Q}(p) + d\,U(p) . \tag{59}$$

Aus Gl. (58) folgt

$$p\mathbf{Q}(p) - \mathbf{A}\mathbf{Q}(p) = (p\mathbf{E} - \mathbf{A})\mathbf{Q}(p) = \mathbf{b}\,U(p)$$

und daraus

$$\mathbf{Q}(p) = (p\mathbf{E} - \mathbf{A})^{-1}\,\mathbf{b}\,U(p) .$$

Einsetzen in (59) ergibt

$$X(p) = \mathbf{c}^T\,(p\mathbf{E} - \mathbf{A})^{-1}\,\mathbf{b}\,U(p) + d\,U(p) = [\mathbf{c}^T(p\mathbf{E} - \mathbf{A})^{-1}\mathbf{b} + d]\,U(p) .$$

Da die Übertragungsfunktion G(p) als Quotient der laplacetransformierten Ausgangsgröße und Eingangsgröße definiert ist, erhält man sofort

$$G(p) = \frac{X(p)}{U(p)} = \mathbf{c}^T(p\mathbf{E} - \mathbf{A})^{-1}\,\mathbf{b} + d . \tag{60}$$

Gl. (60) ist der gesuchte Zusammenhang zwischen dem Klemmenmodell als Übertragungsfunktion und dem Zustandsmodell mit den Bestimmungsstücken **A**, **b**, $\mathbf{c}^T$, d.
Zur Auswertung von (60) sind Matrizenoperationen erforderlich. Am kompliziertesten ist die Bildung der inversen Matrix $(p\mathbf{E} - \mathbf{A})^{-1}$, die eine Funktion von p ist. Ein normales Inversionsverfahren für Zahlenmatrizen kann deshalb nicht verwendet werden. Daher muß ein besonderer Algorithmus angewendet werden.

2.1.2. Faddejew-Algorithmus

Der Faddejew-Algorithmus [9] dient zur Berechnung der Inversen $(p\mathbf{E} - \mathbf{A})^{-1}$. Das Ergebnis seiner Anwendung hat die Form

$$(p\mathbf{E} - \mathbf{A})^{-1} = \frac{\mathbf{E}p^{n-1} + \mathbf{R}_1 p^{n-2} + \mathbf{R}_2 p^{n-3} + \ldots + \mathbf{R}_{n-1}}{s_1 p^n + s_2 p^{n-1} + \ldots + s_n p + s_{n+1}} . \tag{61}$$

Die Matrizen $\mathbf{R}_i$ und Koeffizienten s_i werden nach folgendem Schema über die Hilfsmatrizen $\mathbf{D}_i$ gebildet:

$$\begin{aligned}
s_1 &= 1 & \mathbf{R}_0 &= \mathbf{E} & \mathbf{D}_1 &= \mathbf{A} \\
s_2 &= -\mathrm{sp}(\mathbf{D}_1) & \mathbf{R}_1 &= \mathbf{D}_1 + s_2\mathbf{E} & \mathbf{D}_2 &= \mathbf{A}\,\mathbf{R}_1 \\
s_3 &= -\frac{\mathrm{sp}(\mathbf{D}_2)}{2} & \mathbf{R}_2 &= \mathbf{D}_2 + s_3\mathbf{E} & \mathbf{D}_3 &= \mathbf{A}\,\mathbf{R}_2 \\
& \ldots & & \ldots & & \ldots \\
s_n &= -\frac{\mathrm{sp}(\mathbf{D}_{n-1})}{n-1} & \mathbf{R}_{n-1} &= \mathbf{D}_{n-1} + s_n\mathbf{E} & \mathbf{D}_n &= \mathbf{A}\,\mathbf{R}_{n-1} \\
s_{n+1} &= -\frac{\mathrm{sp}(\mathbf{D}_n)}{n} & \mathbf{R}_n &= \mathbf{D}_n + s_{n+1}\mathbf{E} = \mathbf{0} .
\end{aligned} \tag{62}$$

Dabei bedeutet sp(**D**) die Spur der Matrix **D**, das ist die Summe ihrer Hauptdiagonalelemente:

$$\mathrm{sp}(\mathbf{D}) = d_{11} + d_{22} + \ldots + d_{nn} .$$

Der Koeffizient s_1 im Nenner von (61) hat also immer den Wert $s_1 = 1$. Die Matrix $\mathbf{R}_n$ wird eigentlich nicht benötigt. Sie muß gleich der Nullmatrix **0** sein. Wird sie berechnet, dann können die Werte ihrer Elemente zur Einschätzung der Rechengenauigkeit dienen.
Aus (60) und (61) erhält man für die gesuchte Übertragungsfunktion

$$G(p) = \mathbf{c}^T \frac{\mathbf{E}p^{n-1} + \mathbf{R}_1 p^{n-2} + \ldots + \mathbf{R}_{n-1}}{s_1 p^n + s_2 p^{n-1} + \ldots + s_{n+1}} \mathbf{b} + d .$$

Bringt man diesen Ausdruck auf einen Nenner, dann bekommt man

$$\begin{aligned}
G(p) = [&\mathbf{c}^T\mathbf{b}\, p^{n-1} + \mathbf{c}^T\mathbf{R}_1\mathbf{b}\, p^{n-2} + \ldots + \mathbf{c}^T\mathbf{R}_{n-1}\mathbf{b} \\
&+ d(s_1 p^n + s_2 p^{n-1} + \ldots + s_{n+1})] : (s_1 p^n + s_2 p^{n-1} + \ldots + s_{n+1}) .
\end{aligned} \tag{63}$$

Die Ausdrücke $\mathbf{c}^T\mathbf{R}_i\mathbf{b}$ sind skalare Koeffizienten, für die h_i gesetzt werden soll:

$$h_i = \mathbf{c}^T\mathbf{R}_{n-i}\mathbf{b} . \tag{64}$$

Faßt man die Glieder mit gleichen Potenzen von p im Zähler von (63) zusammen, dann erhält man für die Koeffizienten der Übertragungsfunktion (57)

	Zählerkoeffizient	Nennerkoeffizient
p^0	: $b_0 = h_1 + d\, s_{n+1}$	$a_0 = s_{n+1}$
p^1	: $b_1 = h_2 + d\, s_n$	$a_1 = s_n$
	⋮	
p^{n-1}	: $b_{n-1} = h_n + d\, s_2$	$a_{n-1} = s_2$
p^n	: $b_n = d\, s_1 = d$	$a_n = s_1 = 1$

(65)

Die Beziehungen lassen sich im Programmablaufplan Bild 11 zusammenfassen, der den Kern des Rechenprogramms bildet. $\mathbf{f}^T$ in den Zeilen '3' und '4' ist ein Hilfsvektor, der dazu dient, die Produkte $\mathbf{c}^T\mathbf{Rb}$ in zwei Schritten zu berechnen.

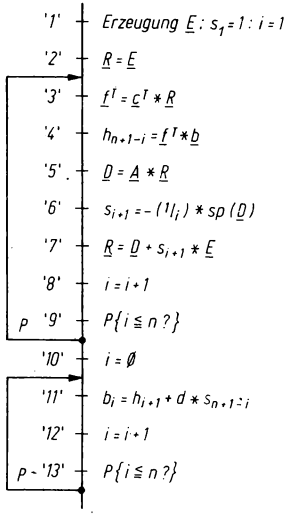


Bild 11. Berechnung der Koeffizienten von Zähler- und Nennerpolynom der Übertragungsfunktion G(p) nach dem Faddejew-Algorithmus

2.1.3. Rechenprogramm

Der PAP nach Bild 11 bildet die Grundlage eines Teiles des Rechenprogramms FREQ, mit dem Frequenzgänge berechnet werden können (s. Abschn. 2.2.). Als Eingaben für dieses Programm sind auch Zustandsmodelle möglich, die dann in Übertragungsfunktionen umgerechnet werden.

Die dabei zu berechnenden Koeffizienten $b_0 \ldots b_n$ der Übertragungsfunktion (57) werden durch die Feldelemente H (1,1) bis H(N + 1,1) dargestellt. Die Nennerkoeffizienten $a_0 \ldots a_n$, im Algorithmus (62) und Bild 11 durch $s_{n+1} \ldots s_1$ ausgedrückt, sind in den Feldelementen H(1,2)...H(N+1,2) untergebracht. Der Programmname für den Durchgangskoeffizienten d lautet DD. Alle anderen Variablenbezeichnungen im PAP und Programm FREQ sind identisch; Matrizen werden durch gleichbezeichnete Felder dargestellt.

Die Hauptschleife '3' bis '9' im PAP Bild 11 wird durch die Laufanweisung über I in den Zeilen 23Ø bis 36Ø gebildet. Innerhalb dieser Schleife werden die Matrizenoperationen durch weitere Laufanweisungen verwirklicht.

Zur Kontrolle der Rechengenauigkeit werden in den Zeilen 4ØØ bis 45Ø die Elemente der Matrix $\mathbf{R}_n$, die nahe an null sein sollen, ausgedruckt. Die Ergebnisse, also die Zählerkoeffizienten der Übertragungsfunktion H(1,1) ... H(N + 1,1) und Nennerkoeffizienten H(1,2) ... H(N + 1,2), werden in den Zeilen 54Ø bis 62Ø in übersichtlicher Tabellenform ausgegeben.

2.1.4. Anwendungsbeispiel

Tafel 7 zeigt die mit dem Programm FREQ aus den Matrizen der Tafeln 4 und 5 berechneten Zähler- und Nennerkoeffizienten der verschiedenen Übertragungsfunktionen.

Tafel 7. Koeffizienten verschiedener Übertragungsfunktionen G(p)

Koeffizient	Stellstrecke	Störstrecke	Regelkreis mit PI-Regler Störverhalten
b_0	1.232 E-3	1.8 E-4	0
b_1	.0148	3.282 E-3	1.8 E-4
b_2	0	.01287	3.28 E-3
b_3	0	0	.01287
a_0	2.607 E-3	2.607 E-3	9.24 E-4
a_1	.06485	.06485	.02233
a_2	.4832	.4832	.16845
a_3	1	1	.4832
a_4	–	–	1

Die Ordnung der Regelstrecke (Tafel 4) ist n = 3, ebenso die des Regelkreises mit P-Regler (Tafel 5a). Mit PI- und PID-Regler erhöht sich die Ordnung auf n = 4. Wendet man das Programm auf das folgende konstruierte Beispiel an:

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{pmatrix}; \quad \mathbf{b} = \begin{pmatrix} 0 \\ 0 \\ 3 \end{pmatrix}; \quad \mathbf{c}^T = (2 \;\; 1 \;\; 0); \quad d = 0,$$

so erhält man die Übertragungsfunktion

$$G(p) = \frac{3p + 6}{p^3 + 6p^2 + 11p + 6}.$$

Die Wurzeln des Nennerpolynoms von G(p) sind −1, −2, −3. Damit kann man schreiben

$$G(p) = \frac{3(p + 2)}{(p + 1)(p + 2)(p + 3)} = \frac{3}{(p + 1)(p + 3)}.$$

Die Möglichkeit, G(p) durch Herauskürzen eines Faktors – im Beispiel: (p + 2) – zu vereinfachen, wird jedoch vom Rechenprogramm nicht automatisch wahrgenommen. Hier muß ggf. eine Handrechnung einsetzen.

Dabei ist es notwendig, die jeweiligen Wurzeln des Zählerpolynoms

$$b_0 + b_1 p + \ldots + b_n p^n = 0$$

sowie des Nennerpolynoms

$$a_0 + a_1 p + \ldots + a_n p^n = 0$$

zu bestimmen. Dazu kann das Programm EWEPOL verwendet werden (s. Abschn. 2.3.).

2.2. Frequenzgänge

2.2.1. Herleitung

Den Frequenzgang $G(j\omega)$ erhält man, indem man in der Übertragungsfunktion (57) an die Stelle des Laplace-Operators p die imaginäre Größe $j\omega$ setzt, wobei ω die Kreisfrequenz bedeutet. Bei der Zahlenrechnung in diesem Band trägt ω die Einheit s^{-1}.
Mit den Rechenregeln für komplexe Größen lassen sich die Ausdrücke im Zähler und Nenner von $G(j\omega)$ in Real- und Imaginärteil aufspalten:

$$G(j\omega) = \frac{b_0 + b_1 j\omega + b_2 (j\omega)^2 + \ldots + b_n (j\omega)^n}{a_0 + a_1 j\omega + a_2 (j\omega)^2 + \ldots + a_n (j\omega)^n}$$

$$= \frac{b_0 - b_2\omega^2 + b_4\omega^4 + \ldots + j\omega(b_1 - b_3\omega^2 + \ldots)}{a_0 - a_2\omega^2 + a_4\omega^4 + \ldots + j\omega(a_1 - a_3\omega^2 + \ldots)} \,. \tag{66}$$

Der Frequenzgang $G(j\omega)$ kann beschrieben werden durch den Amplitudengang $|G(j\omega)|$ und den Phasengang $\varphi(\omega) = \arg\{G(j\omega)\}$. Um diese Funktionen zu berechnen, wird (66) in folgender Form dargestellt:

$$G(j\omega) = \frac{U(\omega) + j\,V(\omega)}{W(\omega) + j\,Y(\omega)} \,.$$

Dabei ist

$$\begin{aligned} U(\omega) &= b_0 - b_2\omega^2 + b_4\omega^4 - + \ldots \\ V(\omega) &= \omega(b_1 - b_3\omega^2 + - \ldots) \\ W(\omega) &= a_0 - a_2\omega^2 + a_4\omega^4 - + \ldots \\ Y(\omega) &= \omega(a_1 - a_3\omega^2 + - \ldots) \,. \end{aligned} \tag{67}$$

Damit kann man für den Amplitudengang schreiben

$$|G(j\omega)| = \sqrt{\frac{U^2 + V^2}{W^2 + Y^2}} \,. \tag{68}$$

Führt man die Phasenwinkel für Zähler und Nenner P_Z, P_N ein:

$$P_Z = \arctan\frac{V}{U}, \quad P_N = \arctan\frac{Y}{W}, \tag{69}$$

so kann man für den Phasengang $\varphi(\omega)$ schreiben

$$\varphi(\omega) = P_Z - P_N .$$

Tritt zur Übertragungsfunktion (57) bzw. dem Frequenzgang (66) noch ein Laufzeitanteil T_L mit der Übertragungsfunktion e^{-pT_L} hinzu, dann ist die Beziehung für den Phasengang zu ergänzen und endgültig zu schreiben

$$\varphi(\omega) = P_Z - P_N - \omega T_L . \tag{70}$$

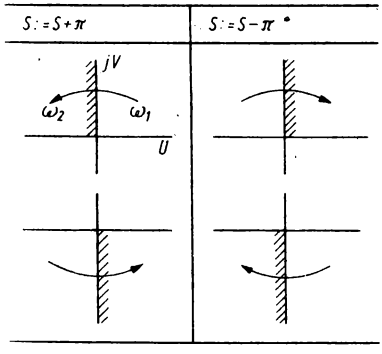
Die Beziehungen (69) müssen vor Umsetzung in ein Rechenprogramm überarbeitet werden, weil bei direkter Programmierung

- für $U = 0$ sowie $W = 0$ Fehlermeldung erfolgen würde
- mit der BASIC-Funktion ATN nur die Hauptwerte der arctan-Funktion berechnet werden können
- bei höheren Systemordnungen $n > 4$ der Phasenwinkel den Wert 360° (2π) überschreitet, was jedoch aus U, V bzw. W, Y allein nicht erkennbar ist.

Die auf der Basis von (69) errechenbaren Werte müssen also noch ergänzt werden. Bezeichnet Arctan ... den mit der BASIC-Funktion ATN(...) berechenbaren Hauptwert, dann berechnen sich die Phasenwinkel zu

$$P_Z = \text{Arctan}\,\frac{V}{U} + S_Z , \quad P_N = \text{Arctan}\,\frac{Y}{W} + S_N . \tag{71}$$

Die Summanden S_Z, S_N beinhalten die Bewegung des Phasenwinkels. Sie vergrößern/vermindern sich jeweils um π, wenn der durch V, U bzw. Y, W darstellbare Zeiger die V-Achse der U,V-Ebene im Gegenuhrzeigersinn/Uhrzeigersinn überschreitet (Tafel 8). Hat der Zeiger eine volle Umdrehung durchgemacht, also die V-Achse zweimal überschritten, dann hat sich S um 2π vergrößert bzw. verringert.



Tafel 8. Zur Ermittlung des Korrekturterms S bei der Phasenwinkelberechnung

Um das Überschreiten der V-Achse und seine Richtung zu erkennen, müssen die Werte U_i, V_i, die zur Frequenz ω_i gehören, mit den Werten U_{i-1}, V_{i-1}, gehörend zu ω_{i-1}, verglichen werden. Für $\omega \to 0$ kann S_Z bzw. S_N schon einen bestimmten Wert haben. Das hängt von der niedrigsten p-Potenz ab, die im Zähler bzw. Nenner von (65) auftritt. Ist beispielsweise $b_0 = b_1 = 0$, $b_2 > 0$, dann ist $S_{Z0} = \pi$, während sich für $b_2 < 0$ ein Wert $S_{Z0} = 2\pi$ ergibt.

Bei einer grafischen Darstellung des Amplitudengangs als Amplitudenkennlinie

verwendet man üblicherweise logarithmische Abszissen- und Ordinatenmaßstäbe. Die Phasenkennlinie als grafische Darstellung des Phasengangs verwendet eine logarithmische Teilung der Abszissen- (Frequenz-)Achse und eine linear geteilte Ordinatenachse.

2.2.2. Rechenprogramm FREQ

Das Rechenprogramm FREQ berechnet Amplituden- und Phasengänge als Funktion der Kreisfrequenz ω (Programmname O) und stellt sie wahlweise tabellarisch oder grafisch dar.

Programm FREQ

```
10  REM FREQUENZGANGBERECHNUNG
20  LET P$="PRESS ANY KEY !": INPUT "ORDNUNG N";N: PRINT
30  INPUT "LAUFZEIT TL";T: PRINT
40  INPUT "ZUSTANDSMODELL (Z) / KLEMMENMODELL (K)";I$: CLS
50  DIM S(2), H(N + 1,2), R(N,N), D(N,N), F(N), G(2), P(2), UA(2); VA(2)
60  IF I$="Z" THEN PRINT "ZUSTANDSMODELL": PRINT
70  IF I$="K" THEN PRINT "KLEMMENMODELL": PRINT: GOTO 460
80  DIM A(N,N),B(N),C(N)
90  REM PROGRAMMBEGINN
100 PRINT "ZUSTANDSMATRIX A(";N;",";N;")": PRINT
110 FOR I = 1 TO N: FOR J = 1 TO N
120 PRINT "A(";I;J;")";: INPUT A(I,J)
130 NEXT J: NEXT I: PRINT
140 PRINT "STEUERVEKTOR B(";N;")": PRINT
150 FOR I = 1 TO N: PRINT "B(";I;")";
160 INPUT B(I): NEXT I: PRINT
170 PRINT "BEOBACHTUNGSVEKTOR CT(";N;")": PRINT
180 FOR I = 1 TO N: PRINT "CT(";I;")";
190 INPUT C(I): NEXT I: PRINT
200 PRINT "DURCHGANGSKOEFFIZIENT D": PRINT: INPUT DD
210 REM BEGINN ALGORITHMUS
220 LET H(N + 1,2)=1: FOR I = 1 TO N: LET R(I,I) = 1: NEXT I
230 FOR I = 1 TO N: FOR J = 1 TO N: LET F(J) = 0
240 FOR K = 1 TO N
250 LET F(J) = F(J) + C(K) * R(K,J)
260 NEXT K: NEXT J
270 LET H(N + 1 - I,1) = 0: FOR J = 1 TO N
280 LET H(N + 1 - I,1) = H(N + 1 - I,1) + F(J) * B(J): NEXT J
290 FOR J = 1 TO N: FOR K = 1 TO N: LET D(J,K) = 0
300 FOR L = 1 TO N: LET D(J,K) = A(J,L) * R(L,K) + D(J,K)
310 NEXT L: NEXT K: NEXT J
320 LET H(N + 1 - I,2) = 0: FOR J = 1 TO N
330 LET H(N + 1 - I,2) = H(N + 1 - I,2) - D(J,J): NEXT J: LET H(N + 1 - I,2)
    = H(N + 1 - I,2)/I
340 FOR J = 1 TO N: FOR K = 1 TO N
350 LET R(J,K) = D(J,K): NEXT K
360 LET R(J,J) = D(J,J) + H(N + 1 - I,2): NEXT J: NEXT I
370 FOR I = 1 TO N + 1: LET H(I,1) = H(I,1) + DD * H(I,2)
```

```
38Ø  NEXT I: CLS
39Ø  REM AUSGABE
4ØØ  PRINT "KONTROLLE DER RECHENGENAUIGKEIT"
41Ø  PRINT "DIE ELEMENTE DER FOLGENDEN"
42Ø  PRINT "MATRIX SOLLTEN NAHE AN Ø SEIN"
43Ø  PRINT: FOR I = 1 TO N: FOR J = 1 TO N
44Ø  PRINT TAB(1Ø * (J - 1));R(I,J);
45Ø  NEXT J: PRINT: NEXT I: PAUSE(1ØØ): CLS
46Ø  PRINT "PARAMETER DER UEBERTRAGUNGSFUNKTION"
47Ø  IF I$ = "Z" GOTO 54Ø
48Ø  FOR J = 1 TO 2
49Ø  IF J = 1 THEN PRINT "ZAEHLERKOEFFIZIENTEN"
5ØØ  IF J = 2 THEN PRINT "NENNERKOEFFIZIENTEN"
51Ø  PRINT: FOR I = 1 TO N + 1
52Ø  PRINT "P↑";I - 1;
53Ø  INPUT H(I,J): NEXT I: PRINT: NEXT J: CLS
54Ø  PRINT AT(3,7);"ZAEHLER:   |   NENNER:"
55Ø  LINE 12,22Ø,272,22Ø
56Ø  FOR I = 1 TO N + 1: IF H(I,1) = Ø GOTO 6ØØ
57Ø  PRINT AT(I + 5,Ø);H(I,1)
58Ø  PRINT AT(I + 5,13);"P↑";I - 1
59Ø  PRINT AT(I + 5,17);" | "
6ØØ  IF H(I,2) = Ø GOTO 62Ø
61Ø  PRINT AT(I + 5,18);H(I,2): PRINT AT(I + 5,31);"P↑";I - 1
62Ø  NEXT I: PRINT: PRINT
63Ø  IF T < > Ø THEN PRINT AT(15,Ø);"LAUFZEITFAKTOR :E↑(-";T;"P)"
64Ø  PAUSE(2Ø): PRINT AT(3Ø,1Ø);P$
65Ø  IF INKEY$ = "" GOTO 65Ø
66Ø  CLS: INPUT "NIEDRIGSTER OMEGA-WERT OMU";OM
67Ø  IF OM = Ø THEN PRINT "???" : PAUSE(2Ø): GOTO 66Ø
68Ø  INPUT "WIEVIEL WERTE JE DEKADE ?";DE
69Ø  LET K = 1Ø↑(1/DE)
7ØØ  INPUT "WIEVIEL DEKADEN ?";AD
71Ø  LET L = DE * AD
72Ø  INPUT "TABELLE (T) / GRAFIK (G) ?";C$: CLS
73Ø  IF C$ = "G" GOTO 76Ø
74Ø  PRINT TAB(3);"OMEGA";TAB(11);"|    |G|";TAB(22);"|    PHI"
75Ø  PRINT "----------------------------------"
76Ø  LET A$ = " ": IF C$ = "T" GOTO 8ØØ
77Ø  INPUT "AMPLITUDENGANG (A) / PHASENGANG (P) ?";A$: CLS
78Ø  IF A$ = "P" THEN GOSUB 144Ø
79Ø  IF A$ = "A" THEN GOSUB 137Ø
8ØØ  FOR J = 1 TO 2: FOR I = Ø TO N: IF H(I + 1,J) = Ø THEN NEXT I
81Ø  IF H(I + 1,J) > Ø THEN LET S(J) = PI * (INT(I/2) - (INT((I + 1)/4) = (I + 1)/4)):
     GOTO 83Ø
82Ø  LET S(J) = PI * (INT((I + 2)/2) - (INT((I + 3)/4) = (I + 3)/4))
83Ø  LET F = SGN(H(I + 1,J)): FOR S = Ø TO I/2: LET F = -F: NEXT S
84Ø  LET UA(J) = F * (INT(I/2) = I/2): LET VA(J) = F * (INT(I/2) < > I/2): NEXT J
85Ø  LET O = OM: FOR H = 1 TO L + 1: FOR J = 1 TO 2: GOSUB 158Ø
86Ø  IF U < Ø AND UA(J) > = Ø AND VA(J) > = Ø THEN LET S(J) = S(J) + PI
87Ø  IF U > = Ø AND UA(J) < = Ø AND VA(J) < Ø THEN LET S(J) = S(J) + PI
88Ø  IF U > = Ø AND UA(J) < Ø AND VA(J) > Ø THEN LET S(J) = S(J) - PI
89Ø  IF U < Ø AND UA(J) > = Ø AND VA(J) < Ø THEN LET S(J) = S(J) - PI
```

```
9ØØ  LET G(J) = U * U + V * V
91Ø  IF U = Ø AND V > Ø THEN LET P(J) = PI/2: GOTO 94Ø
92Ø  IF U = Ø AND V < Ø THEN LET P(J) = -PI/2: GOTO 94Ø
93Ø  LET P(J) = ATN(V/U)
94Ø  LET P(J) = P(J) + S(J)
95Ø  LET UA(J) = U: LET VA(J) = V
96Ø  NEXT J: LET G = SQR(G(1)/G(2))
97Ø  LET M = P(1) - P(2) - O * T
98Ø  IF C$ = "T" GOTO 1Ø8Ø
99Ø  LET MG = 123 + M * 4Ø/PI
1ØØØ  LET ZG = 123 + 13.897423 * LN(G)
1Ø1Ø  IF A$ = "P" AND MG < = Ø GOTO 115Ø
1Ø2Ø  IF A$ = "A" AND ZG > 25Ø GOTO 115Ø
1Ø3Ø  IF A$ = "A" AND ZG < = Ø GOTO 115Ø
1Ø4Ø  LET BS = 35 + (H - 1) * 18Ø/L
1Ø5Ø  IF A$ = "A" THEN PSET BS,ZG
1Ø6Ø  IF A$ = "P" THEN PSET BS,MG
1Ø7Ø  GOTO 115Ø
1Ø8Ø  FOR S = -5 TO 3
1Ø9Ø  IF OM < = 1Ø↑S THEN LET FK = 1Ø↑(1 - S): GOTO 111Ø
11ØØ  NEXT S
111Ø  LET O1 = INT((O + .5)/FK) * FK)/FK
112Ø  LET G = INT((G + .ØØØ5) * 1ØØØ)/1ØØØ
113Ø  LET M = INT((M + .ØØØ5) * 1ØØØ)/1ØØØ
114Ø  PRINT TAB(2);O1;TAB(12);G;TAB(24);M
115Ø  LET O = O * K
116Ø  NEXT H
117Ø  PAUSE(5Ø): PRINT AT(3Ø, 1Ø);P$
118Ø  IF INKEY$ = "" GOTO 118Ø
119Ø  CLS: PRINT "EINGABEN": PRINT
121Ø  IF C$ = "T" GOTO 124Ø
122Ø  IF A$ = "A" THEN PRINT "PHASENGANG:        P": GOTO 125Ø
123Ø  IF A$ = "P" THEN PRINT "AMPLITUDENGANG:A": GOTO 125Ø
124Ø  PRINT "DIAGRAMME :      D": GOTO 126Ø
125Ø  PRINT "TABELLEN :       T"
126Ø  PRINT "NEUES OMEGA :    O"
127Ø  PRINT "NEUEINGABE :     N"
128Ø  PRINT "ENDE :           E": PRINT
129Ø  INPUT G$: CLS
13ØØ  IF G$ = "N" THEN RUN
131Ø  IF G$ = "O" GOTO 66Ø
132Ø  IF G$ = "D" THEN LET C$ = "G": GOTO 76Ø
133Ø  IF G$ = "T" THEN LET C$ = "T": GOTO 74Ø
134Ø  IF G$ = "A" THEN LET A$ = "A": GOTO 78Ø
135Ø  IF G$ = "P" THEN LET A$ = "P": GOTO 78Ø
136Ø  IF G$ = "E" THEN END
137Ø  GOSUB 15ØØ
138Ø  PRINT AT(3,1);" |G| "
139Ø  FOR I = 59 TO 187 STEP 32
14ØØ  PSET 22,I: PSET 24,I: NEXT I
141Ø  PRINT AT(12,Ø);"1Ø": PRINT AT(8,Ø);"1ØØ": PRINT AT(2Ø,Ø);".1"
142Ø  PRINT AT(24,Ø);".Ø1": PRINT AT(16,Ø);"1"
143Ø  RETURN
```

```
144Ø  GOSUB 15ØØ
145Ø  PRINT AT(3,1);"PHI"
146Ø  FOR I = 43 TO 2Ø3 STEP 2Ø
147Ø  PSET 22,I: PSET 24,I: NEXT I
148Ø  PRINT AT(11,1);"PI": PRINT AT(21,Ø);"-PI"
149Ø  RETURN
15ØØ  LINE 2Ø,123,25Ø,123: LINE 23,2Ø,23,22Ø
151Ø  PRINT AT(19,26);"OM/OMU": PRINT AT(1,24);"OMU = ";OM
152Ø  FOR I = Ø TO AD
153Ø  PSET I * 18Ø/AD + 35, 124: PSET I * 18Ø/AD + 35, 121: NEXT I
154Ø  FOR I = Ø TO AD: IF I < = 2 GOTO 156Ø
155Ø  PRINT AT(17,3 + 21.914 * I/AD);E↑";I;: GOTO 157Ø
156Ø  PRINT AT(17,3 + 21.914 * I/AD);1Ø↑I
157Ø  NEXT I: RETURN
158Ø  LET U = Ø: LET V = Ø: LET X = 1
159Ø  FOR M = 1 TO N + 1 STEP 2
16ØØ  LET U = U + X * H(M,J): LET X = -X * O * O: NEXT M
161Ø  LET X = O: FOR M = 1 TO N STEP 2
162Ø  LET V = V + X * H(M + 1,J): LET X = -X * O * O: NEXT M
163Ø  RETURN
```

Eingabewerte sind die Zählerkoeffizienten $b_0 \ldots b_n$ sowie Nennerkoeffizienten $a_0 \ldots a_n$ der Übertragungsfunktion. (Programmnamen H(1,1)...H(N + 1,1) sowie H(1,2)...H(N + 1,2)).
Vor Eingabe dieser Parameterwerte in den Programmzeilen 48Ø bis 53Ø werden eingangs die Ordnung N und der Wert einer eventuell auftretenden Laufzeit T_L eingegeben (Zeilen 2Ø, 3Ø).
Das Programm FREQ verarbeitet auch Zustandsmodelle. Diese werden in dem im Abschn. 2.1. beschriebenen Programmteil in die Parameter eines Klemmenmodells umgerechnet (Zeilen 8Ø bis 45Ø). Das Programm berechnet dann eine Folge von Frequenzgangpunkten für diskrete Frequenzen $\omega_i > Ø$, die nach einer geometrischen Folge gestuft sind. Bei einer logarithmischen Teilung der ω-Achse ergeben sich dadurch auf dieser Achse gleiche Abstände der Frequenzgangpunkte. Für die ω_i gilt demnach

$$\omega_i = \omega_u \, k^i ; \qquad i = 0, 1, 2, \ldots$$

Der Faktor k ergibt sich aus der gewünschten Anzahl d_E von Rechenwerten je Dekade (Frequenzverhältnis 1 : 10).

$$k = \sqrt[d_E]{10} \qquad \text{(Programmzeile 69Ø).}$$

Unganzzahlige d_E sind möglich. Der niedrigste Wert ω_u (Programmname OM) ist in Zeile 66Ø einzugeben; die Anzahl der Dekaden a_D, für die Frequenzgangwerte berechnet werden sollen, in Zeile 7ØØ. Auch hier sind unganzzahlige Werte möglich.
Die Berechnung von $d_E \cdot a_D + 1$ Amplitudengangwerten $|G(j\omega_i)|$ (Programmname G) erfolgt in einer großen Laufanweisung über H, die die Zeilen 85Ø bis 116Ø umfaßt. Dabei sind die Funktionen U und V, W und Y aus den Koeffizienten b_i und a_i der Übertragungsfunktion zu berechnen. Da die dazu verwendeten Beziehungen (67) hinsichtlich der b_i und a_i völlig gleich aufgebaut sind, werden sie in einem Unterprogramm (Zeilen 158Ø bis 163Ø) verwirklicht, in dem (H(I + 1,1) für

b_i und H(I + 1,2) für die a_i steht. In den Zeilen 8ØØ bis 84Ø werden die Werte S_Z bzw. S_N sowie U und V aus dem Index I des niedrigsten von Null verschiedenen Zähler- bzw. Nennerkoeffizienten b_i bzw. a_i sowie aus dessen Vorzeichen berechnet. Die Summanden S_Z, S_N sind im Feld S(J) mit J = 1 für Zähler und J = 2 für Nenner zusammengefaßt, ebenso wie die Phasenwinkel P_Z und P_N im Feld P(J) und die zur Berechnung von G (Zeile 96Ø) verwendeten Zähler- und Nennerausdrücke $U^2 + V^2$ bzw. $W^2 + Y^2$ im Feld G(J).

Bei den weiteren Aufrufen des Unterprogramms für die einzelnen Frequenzwerte ω_i (Zeile 85Ø) werden die Phasenwinkelkorrektursummanden S(J) jeweils nach den Bedingungen der Tafel 8 aktualisiert. Dazu werden herangezogen: die aktuellen im Unterprogramm mit der Frequenz ω_i berechneten Werte U und V sowie die zum davorliegenden Frequenzwert ω_{i-1} gehörenden „alten" Werte, gespeichert in den Feldern UA(J) und VA(J).

Der Phasenwinkel φ (Programmname M) wird in Zeile 97Ø entsprechend (70) berechnet.

Für einen Tabellendruck ist die Ausgabe im normalen 8stelligen BASIC-Format unschön. Daher erfolgt in den Zeilen 1Ø8Ø bis 113Ø eine Formatierung der auszudruckenden Werte von ω, | G | und φ, während für die Diagrammausgabe der Amplitudenkennlinie in Zeile 1ØØØ der G-Wert logarithmiert wird.

Bild 12 zeigt als Anwendungsbeispiel die mit dem Programm FREQ berechneten Frequenzgangkennlinien für die Stellstrecke des Beispielsystems sowie für das Störverhalten des mit einem PI-Regler (Einstellwerte nach Tafel 5b) geschlossenen Regelkreises.

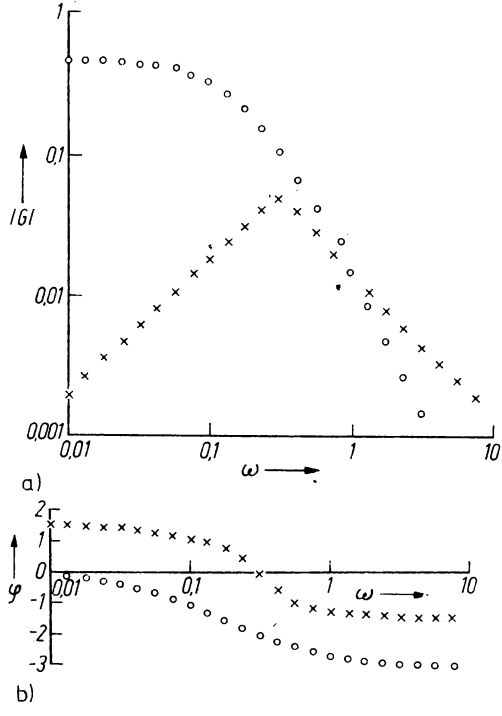


Bild 12. Frequenzgangkennlinien für den offenen und geschlossenen Regelkreis
a) Amplitudenkennlinie; b) Phasenkennlinie
ooo Stellstrecke
xxx Störfrequenzgang geschlossener Regelkreis

2.3. Eigenwertberechnung

2.3.1. Bedeutung der Eigenwerte

Sehr wichtige Größen zur Beurteilung des dynamischen Verhaltens eines durch sein Zustandsmodell beschriebenen linearen Systems sind seine Eigenwerte p_i. Ein System mit der Ordnung n (Format der Systemmatrix **A**: n,n) hat genau n Eigenwerte, die einfach oder mehrfach, reell oder komplex sein können. Mit jedem Eigenwert p_i hängt ein Anteil im Zeitverhalten des Systems zusammen, wobei die im Bild 13 gezeigten Zusammenhänge gelten.
Reelle Eigenwerte bewirken monoton verlaufende Anteile in der Zeitfunktion (Bild 13 a, b, c). Komplexe Eigenwerte treten immer paarweise auf (Bild 13 d, e, f); sie bewirken oszillatorische (schwingende) Anteile. Negativ reelle (Bild a) oder komplexe Eigenwerte mit negativem Realteil (Bild d) ergeben abklingende Zeitvorgänge (asymptotisch stabile Vorgänge). Positiv reelle Eigenwerte oder solche mit positivem Realteil (Bild c, f) bewirken aufklingende Vorgänge (Instabilität). Die Vorgänge nach Bild b und e entsprechen der Stabilitätsgrenze.

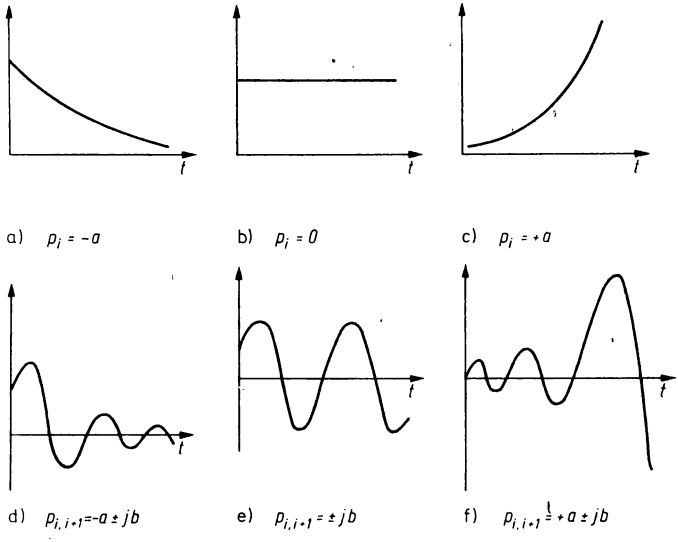


Bild 13. Zusammenhänge zwischen Eigenwerten und Anteilen im Zeitverhalten

Die Eigenwerte eines Systems ergeben sich aus seiner Zustandsmatrix **A** nach der Eigenwertgleichung

$$|\mathbf{pE} - \mathbf{A}| = 0. \tag{72}$$

Liegt ein Klemmenmodell vor in Form einer Übertragungsfunktion G(p) – s. Gl. (57) –, dann lautet die Eigenwertgleichung

$$F(p) = a_0 + a_1 p + \ldots a_n p^n = 0\,. \tag{73}$$

Es sind also die Nullstellen (Wurzeln) des Nennerpolynoms der Übertragungsfunktion zu bestimmen.
Die Beziehung (72) zur Berechnung der Eigenwerte einer quadratischen Matrix A läßt sich ebenfalls auf die Lösung eines Polynoms n-ten Grades der Form (73) zurückführen, bei dem die Koeffizienten a_i aus den Matrizenelementen a_{ij} gebildet werden. F(p) heißt dann das *charakteristische Polynom* der Matrix A. Die Bestimmung der Eigenwerte eines dynamischen Systems verlangt also die beiden folgenden Schritte:

1. Bildung des charakteristischen Polynoms;
2. Bestimmung der Nullstellen (Wurzeln) des charakteristischen Polynoms.

Liegt das Modell als Übertragungsfunktion vor, dann entfällt Schritt 1.

2.3.2. Ermittlung des charakteristischen Polynoms

Gl. (72) lautet ausgeschrieben

$$\begin{vmatrix} p - a_{11} & - a_{12} & \dots & - a_{1n} \\ - a_{21} & p - a_{22} & \dots & \\ & & & \\ - a_{n1} & & \dots & p - a_{nn} \end{vmatrix} = 0 .$$

Die direkte Berechnung dieser Determinante ist aber höchstens für $n = 2$ oder $n = 3$ zumutbar. Für größere n wird der Algorithmus von Leverrier verwendet [10], der die Koeffizienten des charakteristischen Polynoms nach einem numerischen Verfahren bildet, dem auch der Faddejew-Algorithmus (s. Abschn. 2.1.2.) zugrunde liegt. Das charakteristische Polynom (73) wird dazu in folgender Form mit den Koeffizienten $k_2 \dots k_{n+1}$ geschrieben:

$$F(p) = p^n + k_2 p^{n-1} + k_3 p^{n-2} + \dots + k_{n+1} = 0 .$$

Dabei werden die k_i mit einer Hilfsmatrix S(n,n) wie folgt berechnet:

$$\begin{aligned} &\mathbf{T}_1 = \mathbf{A} , \qquad k_2 = -\mathrm{sp}(\mathbf{T}_1) , \qquad \mathbf{S}_1 = \mathbf{T}_1 + k_2 \mathbf{E} \\ &\mathbf{T}_2 = \mathbf{A}\mathbf{S}_1 , \qquad k_3 = -\frac{\mathrm{sp}(\mathbf{T}_2)}{2} , \qquad \mathbf{S}_2 = \mathbf{T}_2 + k_3 \mathbf{E} \\ &\mathbf{T}_n = \mathbf{A}\mathbf{S}_{n-1} , \qquad k_{n+1} = -\frac{\mathrm{sp}(\mathbf{T}_n)}{n} . \end{aligned} \tag{74}$$

2.3.3. Bestimmung der Polynomwurzeln

Die Aufgabe besteht darin, die n Nullstellen $p_1 \dots p_n$ eines Polynoms der Form (73) zu ermitteln. Wenn das für $n = 2$ oder $n = 3$ zwar auch in geschlossener Form möglich ist, so soll doch ein allgemein verwendbares numerisches Verfahren entwickelt werden. Die wichtigsten Schritte werden durch den Programmablaufplan Bild 14

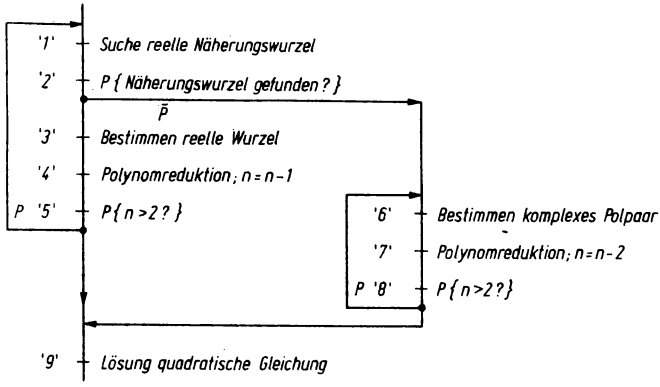


Bild 14. Programmablaufplan zur Bestimmung der Polynomwurzeln

dargestellt. Für die Unbekannte wird nicht p, sondern wie in der Mathematik üblich das Symbol x verwendet.
Nach Finden eines Näherungswerts für eine reelle Nullstelle (Zeile '1') – ist der Grad n des Polynoms ungerade, so existiert auf alle Fälle mindestens eine reelle Nullstelle – wird deren Wert x_1 in Zeile '3' genau bestimmt.
Durch Division des Polynoms F(x) durch den Wurzelfaktor $(x - x_1)$ erhält man ein reduziertes Polynom vom Grad $n-1$, mit dem das Verfahren so lange wiederholt wird, bis sich eine quadratische Gleichung ($n = 2$, Zeile '5') ergibt, deren Lösung in geschlossener Form angegeben werden kann (Zeile '9'). Läßt sich kein Näherungswert für eine reelle Wurzel finden (Zeile '2'), dann besitzt das Polynom (n gerade) nur komplexe Wurzelpaare, evtl. auch reelle Doppelwurzeln.
In Zeile '6' wird ein Wurzelpaar x_i, $\bar{x}_i$ gesucht ($\bar{x}_i$ konjugiert komplexer Wert). Die Division des Ursprungspolynoms durch den Wurzelpaarfaktor $(x - x_i)(x - \bar{x}_i)$ ergibt ein reduziertes Polynom vom Grad $n-2$ (Zeile '7'), mit dem das Verfahren so lange wiederholt wird, bis $n = 2$ ist (Zeile '8'). Um diesen Algorithmus in ein Programm umzusetzen, müssen folgende Einzelprobleme gelöst werden:

'1': Suche einer rellen Näherungswurzel (Bild 15)

Nachdem ein Intervall $x_{min} \ldots x_{max}$ festgelegt worden ist, in dem eine Nullstelle vermutet wird, werden, beginnend mit $x_1 = x_{min}$, nacheinander eine Folge von Funktionswerten $F_1, F_2, \ldots$ berechnet. Ein Näherungswert ist gefunden, wenn zwei auf-

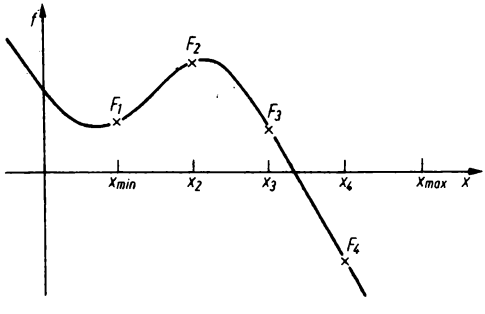


Bild 15. Zur Suche nach einer reellen Näherungswurzel

einanderfolgende Funktionswerte unterschiedliches Vorzeichen haben (im Bild 15 F_3, F_4).

'3': Bestimmen einer reellen Nullstelle

Der gefundene Näherungswert (im Bild 15 x_4) wird mit dem Newtonschen Verfahren iterativ verbessert. Dieses Verfahren wurde schon im Abschn. 1.3.2. – s. Gl. (21), Bild 7 – beschrieben.
Zu seiner Anwendung werden die Werte der Funktion f und ihrer Ableitung f' benötigt.

'4': Polynomreduktion

Die Funktion f(x) ist ein Polynom der Form (73) vom Grad n. Als algorithmisches Verfahren, mit dem sowohl der Funktionswert $f(x_0)$, der ihrer Ableitung $f'(x_0)$ sowie die Koeffizienten des durch den Wurzelfaktor $(x - x_i)$ dividierten, im Grad um 1 reduzierten Polynoms bestimmt werden können, wird das Horner-Schema verwendet. Tafel 9 erläutert den Algorithmus. Als Ergebnis seiner Anwendung erhält man Funktionswert und Ableitung. Ist $x_0 = x_i$ eine Nullstelle des Polynoms, dann ist $f(x_i) = 0$, und die Werte des Schemas $a'_n, a'_{n-1}, \ldots, a'_1$ sind die Koeffizienten des reduzierten Polynoms vom Grad $n - 1$:

$$a'_n x^{n-1} + a'_{n-1} x^{n-2} + \ldots + a'_2 x + a'_1 . \tag{75}$$

Tafel 9. Horner-Schema zur Berechnung von $f(x_0)$, $f'(x_0)$ und zur Ermittlung der Koeffizienten des reduzierten Polynoms

	a_n	a_{n-1}	a_{n-2}	a_1	a_0
x_0		$+ a'_n x_0$	$a'_{n-1} x_0$	$a'_2 x_0$	$a'_1 x_0$
	a'_n	a'_{n-1}	a'_{n-2}	a'_1	$\underline{f(x_0)}$
x_0		$+ a''_n x_0$	$a''_{n-1} x_0$	$a''_2 x_0$	
	a''_n	a''_{n-1}	a''_{n-2}	$\underline{\underline{f'(x_0)}}$	

'6': Bestimmen eines komplexen Polpaars

Das Newtonsche Näherungsverfahren nach (19) läßt sich auch für komplexe x anwenden. f und f' werden dann ebenfalls komplex; also $f = f_r + jf_i$; $f' = f'_r + jf'_i$.
Nach den Rechenregeln für komplexe Zahlen erhält man damit aus (19) für den verbesserten Näherungswert

$$x_1 = x_0 - \frac{f_r + jf_i}{f'_r + jf'_i} = x_0 - \frac{f_r f'_r + f_i f'_i + j(f_i f'_r - f_r f'_i)}{f'^2_r + f'^2_i} . \tag{76}$$

Abbruchkriterium für die Iteration ist jetzt

$$|f|^2 = f_r^2 + f_i^2 \leqq \text{eps} .$$

'7': Polynomreduktion

Die Funktionen f_r, f_i, f'_r, f'_i sowie die Koeffizienten des reduzierten Polynoms können algorithmisch mit dem sog. Collatz-Schema berechnet werden [11]. Tafel 10 zeigt den Aufbau des Schemas.

Ist $x_0 = p + jq$ eine Wurzel des Polynoms, dann ist $a'_1 = a'_0 = 0$. In diesem Fall stellen $a'_n \ldots a'_2$ die Koeffizienten des im Grad um 2 reduzierten Polynoms dar:

$$a'_n x^{n-2} + a'_{n-1} x^{n-3} + \ldots + a'_3 x + a'_2 . \tag{77}$$

Tafel 10. Collatz-Schema zur Berechnung von $f(x_0) = f_r + jf_i$, $f'(x_0) = f'_r + jf'_i$ bei komplexem $x_0 = p + jq$

Hilfsgrößen: $t = p^2 + q^2$; $s = -2p$

	a_n	a_{n-1}	a_{n-2}	a_3	a_2	a_1	a_0
t			$-ta'_n$	$-ta'_5$	$-ta'_4$	$-ta'_3$	$-ta'_2$
s		$-sa'_n$	$-sa'_{n-1}$	$-sa'_4$	$-sa'_3$	$-sa'_2$	
	a'_n	a'_{n-1}	a'_{n-2}	a'_3	a'_2	$\underline{\underline{a'_1}}$	$\underline{\underline{a'_0}}$
t			$-ta''_n$	$\ldots -ta''_5$	$-ta''_4$		
s		$-sa''_n$	$-sa''_{n-1}$	$-sa''_4$			
	a''_n	a''_{n-1}	a''_{n-2}	$\underline{\underline{a''_3}}$	$\underline{\underline{a''_2}}$		

Ergebnis: $f(x_0) = a'_1 x_0 + a'_0 = a'_1 p + a'_0 + j a'_1 q$

$f'(x_0) = b_1 x_0 + b_0$

mit $b_1 = 2a''_2 - sa''_3$; $b_0 = sa''_2 - 2ta''_3 + a'_1$

'9': Lösung der quadratischen Gleichung

Bleibt als Ergebnis der Reduktionen nur eine quadratische Gleichung übrig

$$a'_n x^2 + a'_{n-1} x + a'_{n-2} = 0 ,$$

dann kann deren Lösung sofort hingeschrieben werden:

$$x_{1,2} = \begin{cases} -\dfrac{a'_{n-1}}{2a'_n} \pm \sqrt{\left(\dfrac{a'_{n-1}}{2a'_n}\right)^2 - \dfrac{a'_{n-2}}{a'_n}} & \text{für } \left(\dfrac{a'_{n-1}}{2a'_n}\right)^2 \geqq \dfrac{a'_{n-2}}{a'_n} \\ -\dfrac{a'_{n-1}}{2a'_n} \pm j\sqrt{\dfrac{a'_{n-2}}{a'_n} - \left(\dfrac{a'_{n-1}}{2a'_n}\right)^2} & \text{für } \dfrac{a'_{n-2}}{a'_n} > \left(\dfrac{a'_{n-1}}{2a'_n}\right)^2 . \end{cases} \tag{78}$$

2.3.4. Rechenprogramm EWEPOL

Die Zeilennumerierung des Programms EWEPOL beginnt bei 95ØØ. Damit kann man das Programm zusammen mit anderen Programmen abspeichern, um mit seiner Hilfe die charakteristischen Polynome und Eigenwerte von Matrizen zu ermitteln, die in anderen Programmen berechnet worden sind.

Dazu wäre eine Format- und Namensanpassung zu programmieren. Das Programm EWEPOL ist selbstverständlich auch für die Direkteingabe vorgesehen.

Programm EWEPOL

```
9500  REM BERECHNUNG CHARAKTERISTISCHES POLYNOM
9505  INPUT "POLYNOMLOESUNG: P ; EIGENWERTE: E "; I$
9510  INPUT "GRAD N "; N: LET M = N + 1: DIM K(M,3): DEFFN Q(X) = X * X: CLS
9515  IF I$ = "P" GOTO 9590
9520  DIM A(N,N), S(N,N), T(N,N)
9525  FOR I = 1 TO N: FOR J = 1 TO N
9530  PRINT "A(";I;J;")";: INPUT A(I,J): LET T(I,J) = A(I,J)
9535  NEXT J: NEXT I: LET K(1,1) = 1: FOR L = 1 TO N: FOR I = 1 TO N
9540  LET K(L + 1,1) = K(L + 1,1) - T(I,I): NEXT I: LET K(L + 1,1) = K(L + 1,1)/L
9545  FOR I = 1 TO N: FOR J = 1 TO N: LET S(I,J) = T(I,J): NEXT J
9550  LET S(I,I) = T(I,I) + K(L + 1,1): NEXT I
9555  FOR I = 1 TO N: FOR J = 1 TO N: LET T(I,J) = 0
9560  FOR K = 1 TO N: LET T(I,J) = T(I,J) + A(I,K) * S(K,J): NEXT K
9565  NEXT J: NEXT I: NEXT L
9570  CLS: PRINT "CHARAKTERISTISCHES POLYNOM"
9575  FOR I = 1 TO M:PRINT "P↑";M - I,K(I,1): NEXT I
9580  GOTO 9610
9590  REM NULLSTELLENBESTIMMUNG POLYNOM GRAD N
9595  PRINT "POLYNOMKOEFFIZIENT": PRINT: PRINT
9600  FOR I = 1 TO M: PRINT "X↑";M - I;: INPUT K(I,1): NEXT I
9610  PRINT N;" NULLSTELLEN"
9615  FOR I = 1 TO M: LET K(I,2) = K(I,1): LET K(I,3) = K(I,1): NEXT I: PRINT
9620  IF M< = 3 GOTO 9710
9625  LET ZA = 0: LET IP = 0: LET EPS = 1E - 6
9630  INPUT "SUCHBEREICH XMIN ... XMAX ";XN,XX: CLS

9635  LET XS = (XX - XN)/30: LET X = XN: GOSUB 9745
9640  LET F1 = F: LET X = X + XS: IF X< = XX GOTO 9660
9645  PRINT: PRINT "KEINE REELLE NULLSTELLE ZWISCHEN ";XN;"...";XX:
      PRINT
9650  PRINT "ANDERER SUCHBEREICH ? EINGABE 1 / KOMPLEXE WURZELN ?
      EINGABE 2"
9655  INPUT IP: CLS: ON IP GOTO 9630, 9770
9660  GOSUB 9745: IF SGN(F) = SGN(F1) GOTO 9640
9665  LET X = X - F/FS: LET ZA = ZA + 1: IF ZA< = 50 GOTO 9680
9670  PRINT "ABBRUCHSCHRANKE ";EPS;" ZU KLEIN. NEUEINGABE !"
9675  INPUT EPS: LET ZA = 0: GOTO 9635
9680  GOSUB 9745: IF ABS(F) > EPS GOTO 9665
9685  PRINT "NULLSTELLE": PRINT: PRINT X
9690  LET M = M - 1: REM REDUZIERTES POLYNOM
9695  FOR I = 1 TO M: LET K(I,1) = K(I,2): NEXT I
9700  IF M > 3 AND IP< >2 GOTO 9635
9705  IF M > 3 AND IP = 2 GOTO 9770
9710  LET R = FN Q(K(2,2)/(2 * K(1,2)) - K(3,2)/K(1,2)
9715  REM LOESUNG QUADRATISCHE GLEICHUNG
9720  IF R < 0 GOTO 9735
9725  PRINT -K(2,2)/(2 * K(1,2)) + SQR(R): PRINT -K(2,2)/(2 * K(1,2) - SQR(R)
9730  END
9735  PRINT -K(2,2)/(2 * K(1,2));" + J";SQR(-R)
9740  PRINT -K(2,2)/(2 * K(1,2));" -J";SQR(-R): END
9745  REM HORNERSCHEMA
9750  LET K(1,2) = K(1,1): LET K(1,3) = K(1,1)
```

```
9755  FOR I = 1 TO M - 1: LET K(I + 1,2) = K(I,2) * X + K(I + 1,1)
976Ø  LET K(I + 1,3) = K(I,3) * X + K(I + 1,2): NEXT I
9765  LET F = K(M,2): LET FS = K(M - 1,3): RETURN
977Ø  REM NEWTON KOMPLEXE WURZELN
9775  LET P = X: LET Q = X: REM STARTWERTE
978Ø  GOSUB 98Ø5
9785  IF FN Q(FR) + FN Q(FI) < 1E - 8 GOTO 986Ø
979Ø  LET NR = FN Q(SR) + FN Q(SI): LET P = P - (FR * SR + FI * SI)/NR
98ØØ  LET Q = Q + (FR * SI - FI * SR)/NR: GOTO 978Ø
98Ø5  REM COLLATZ-SCHEMA
981Ø  LET T = FN Q(P) + FN Q(Q): LET S = -2 * P: LET K(1,2) = K(1,1)
9815  LET K(2,2) = K(2,1) - S * K(1,2): FOR I = 3 TO M - 1
982Ø  LET K(I,2) = K(I,1) - S * K(I - 1,2) - T * K(I - 2,2): NEXT I
9825  LET K(M,2) = K(M,1) - T * K(M - 2,2): LET FR = K(M - 1,2) * P + K(M,2)
983Ø  LET FI = K(M - 1,2) * Q: LET K(1,3) = K(1,1): LET K(2,3) = K(2,2) - S * K(1,3)
9835  IF M < = 5 GOTO 9845
984Ø  FOR I = 3 TO M - 3: LET K(I,3) = K(I,2) - S * K(I - 1,3) - T * K(I-2,3): NEXT I
9845  LET K(M - 2,3) = K(M - 2,2) - T * K(M - 4,3): LET B1 = 2 * K(M - 2,3) - S
      * K(M - 3,3)
985Ø  LET BØ = S * K(M - 2,3) - 2 * T * K(M - 3,3) + K(M - 1,2)
9855  LET SR = B1 * P + BØ: LET SI = B1 * Q: RETURN
986Ø  PRINT: PRINT "NULLSTELLEN": PRINT
9865  PRINT P;" + J";ABS(Q): PRINT P;" - J";ABS(Q)
987Ø  LET M = M - 2: GOTO 9695
```

Zur Bestimmung des charakteristischen Polynoms einer Matrix A (Programmname A(N,N)) sind deren Elemente ab Zeile 952Ø einzugeben. Sollen die Nullstellen eines beliebigen Polynoms vom Grade n gesucht werden, dann werden dessen reelle Koeffizienten ab Zeile 959Ø eingegeben.
Bei Abarbeitung des Horner- bzw. Collatz-Schemas nach Tafeln 9 und 10 treten Koeffizienten mit den Bezeichnungen a_i, a_i', a_i'' auf. Diese Koeffizienten werden im Programm durch die Elemente eines Feldes K(N+1,3) nach folgendem Schema dargestellt:

$a_n \ldots a_0$: K(1,1) ... K(N + 1,1)
$a_n' \ldots a_0'$: K(1,2) ... K(N + 1,2)
$a_n'' \ldots a_0''$: K(1,3) ... K(N + 1,3) .

Diese Indexzuordnung ist zweckmäßig, weil bei den verschiedenen Reduktionen des Polynomgrads (N := N - 1, N := N - 2) die Koeffizienten $a_n = a_n' = a_n''$ jeweils unverändert bleiben und nur die Koeffizienten mit den niedrigen Indizes (a_0, a_1, ...) in Wegfall kommen; vgl. (75) und (77).
Die Indizierung des charakteristischen Polynoms und des Leverrier-Algorithmus (74) folgt bereits diesem Schema. Der Leverrier-Algorithmus selber ist in den Zeilen 9535 bis 9565 programmiert.
Zur Suche einer reellen Näherungswurzel (Zeile '1') im Suchbereich x_{min} bis x_{max} (Programmnamen XN, XX) (Eingabe: Zeile 963Ø) wird dieser in 30 Abschnitte unterteilt (vgl. Bild 15, in dem drei Abschnitte gewählt wurden). Bei Anwendung des Programms sollte der Suchbereich $x_{max} - x_{min}$ nicht unnötig groß gewählt werden, um zu vermeiden, daß zwei nahe beieinander liegende Nullstellen nicht erkannt werden (Bild 16).

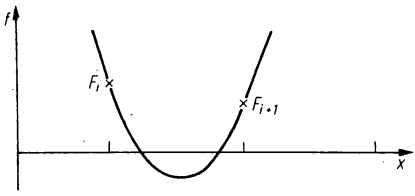


Bild 16. Bei zu großem Suchbereich werden nahe beieinander liegende Nullstellen übersehen

Zur Funktionswertberechnung bei der Nullstellensuche wird das Horner-Schema benutzt, das als Unterprogramm in den Zeilen 9745 bis 9765 niedergelegt ist. F ist der Programmname für den Funktionswert f; FS steht für die Ableitung f'.
Wird im ursprünglich gewählten Suchbereich keine reelle Nullstelle gefunden, kann entweder ein anderer Suchbereich gewählt oder zur Bestimmung eines komplexen Polpaars nach Zeile '6' des PAP übergegangen werden (Zeilen 9645 bis 9655).
Der Newton-Algorithmus zur Nullstellenverbesserung nach (19) erstreckt sich auf die Zeilen 9665 bis 968Ø. Als Abbruchschranke eps ist im Programm (Zeile 9625) der Wert 10^{-6} gewählt worden. Dieser Wert hat sich bewährt. Allerdings kann es vorkommen, daß die Funktion f im Bereich der Nullstelle sehr steil verläuft (Bild 17). Da x nur eine beschränkte Stellenzahl hat, kann f im Nullstellenbereich nur die beiden Werte F^+ und F^- annehmen, die beide betragsmäßig größer sind als die Abbruchschranke eps. In diesem Fall kommt es zu keinem Abbruch; der Algorithmus pendelt beliebig lange um die Nullstelle herum. Daher wird in Zeile 9665 mit ZA die Anzahl der Iterationen gezählt. Überschreitet ZA die Zahl 50, ohne daß es zu einem Abbruch gekommen ist, kann man annehmen, daß eps zu klein gewählt worden war, und in den Zeilen 967Ø, 9675 ist ein größerer Wert einzugeben.

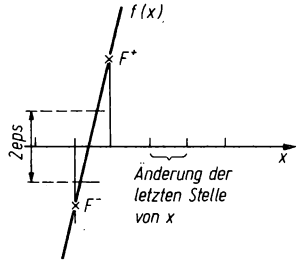


Bild 17. Infolge beschränkter Stellenzahl von x kann f in der Umgebung der Nullstelle nur die Werte F^+ und F^- annehmen

Wurde eine Nullstelle gefunden und in Zeile 9685 ausgedruckt, erfolgt in den Zeilen 969Ø, 9695 die Reduktion des Polynomgrads um 1 sowie die Umspeicherung der Koeffizienten a', die im Horner-Schema berechnet worden waren. Die Suche einer weiteren Nullstelle wird im vorher eingegebenen Suchbereich fortgesetzt.
Die Lösungen der quadratischen Gleichung gemäß (78) werden in den Zeilen 9715 bis 974Ø ermittelt.
Das Newton-Verfahren für komplexe Nullstellen (Zeilen 977Ø bis 98ØØ) verwendet die Variablennamen FR für f_r, FI für f_i, SR für f'_r, SI für f'_i. Da komplexe Nullstellen immer konjugiert komplex anfallen, kann nach Finden einer Nullstelle $p + j\,q$ der zugehörige konjugiert komplexe Wert $p - j\,q$ sofort ohne weitere Rechnung ausgedruckt werden (Zeile 9865).

Mit q = 0 findet das Programm als Sonderfall auch reelle Doppelwurzeln, für die in Zeile '1' des Algorithmus keine Näherungswerte gefunden werden können und die sich überdies einer Verbesserung nach dem einfachen Newton-Verfahren nach Zeile '3' entziehen, wie man aus Bild 18 erkennt.

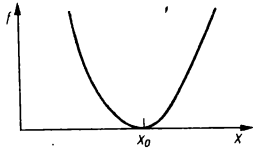


Bild 18. Funktionsverlauf in der Umgebung einer reellen Doppelwurzel x_0

In Anwendung des Programms EWEPOL auf die bisher behandelten Beispiele wurden folgende Ergebnisse erhalten:

Eigenwerte

Systemmatrix **A** (Tafel 4):

$$-0.08,\ -0.1118,\ -0.2913.$$

Regelkreis mit PI-Regler (Tafel 5):

$$-0.081,\ -0.1089,\ -0.1466 \pm \mathrm{j}\ 0.287.$$

Diese Eigenwerte sind die Lösungen der charakteristischen Polynome der entsprechenden Matrizen, deren Koeffizienten a_i auch aus Tafel 7, Spalten 1 und 3 abgelesen werden können. Die b_i in dieser Tafel sind die Koeffizienten der entsprechenden Zählerpolynome. Bestimmt man auch deren Nullstellen, dann lassen sich die Übertragungsfunktionen des Systems in folgender Form darstellen:

Stellübertragungsfunktion der Regelstrecke

$$G_u(p) = \frac{0.0148\ (p + 0.0832)}{(p + 0.08)\ (p + 0.1118)\ (p + 0.2913)}$$

$$\approx \frac{0.0148}{(p + 0.1118)\ (p + 0.2913)} \tag{79}$$

Störübertragungsfunktion des Regelkreises

$$G_z^*(p) = \frac{0.01287p\,(p + 0.08)\ (p + 0.1749)}{(p + 0.081)\ (p + 0.1089)\ [(p + 0.1466)^2 + 0.287^2]}$$

$$\approx \frac{0.01287p\,(p + 0.1749)}{(p + 0.1089)\ [(p + 0.1466)^2 + 0.287^2]} \tag{80}$$

In beiden Übertragungsfunktionen läßt sich also, zumindest im Rahmen der numerischen Genauigkeit, ein gemeinsamer Faktor (p + 0.08) aus Zähler und Nenner herauskürzen.

3. Diskontinuierliche lineare Zustandsmodelle

3.1. Ermittlung von zeitdiskreten Modellen

3.1.1. Diskretisierung der Zeit

In diskontinuierlichen Modellen wird die Zeit t, $t \geqq 0$, die in der Natur kontinuierlich verläuft, in diskrete Abschnitte zerhackt, d. h. in eine Folge diskreter Abschnitte kT, $k = 0, 1, 2, \ldots$, unterteilt. Man spricht deshalb auch von zeitdiskreten Modellen [12]. Diskontinuierliche Modelle kommen der Behandlung auf Digitalrechnern besonders entgegen, weil diese ohnehin mit diskreten Zahlenwerten rechnen, die einer an sich kontinuierlichen Zeitfunktion niemals lückenlos dicht, sondern eben nur in gewissen Zeitabständen T entnommen werden können (Bild 19). Aus der kontinuierlichen Zeitfunktion x(t) wird so eine Folge einzelner Funktionswerte $x(kT) = x_k$, die numerisch verarbeitet werden kann.

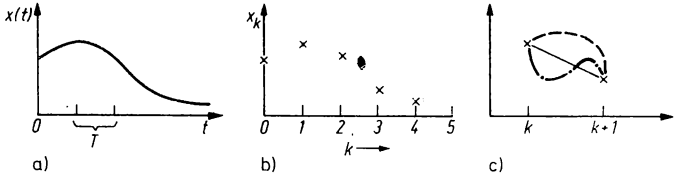


Bild 19. Diskretisierung einer Zeitfunktion
a) kontinuierliche Funktion x(t); b) Folge zeitdiskreter Abtastwerte x_k; c) zwischen zwei Abtastwerten sind keine Aussagen über den Verlauf möglich

Natürlich wünscht man, daß die Ergebnisse dieser numerischen Rechnungen dem Vorgang im Originalsystem, in dem die Zeit kontinuierlich verläuft, möglichst ähnlich sein sollen. Auf alle Fälle sollen die Lösungen des diskontinuierlichen Systems in den Abtastpunkten mit denen des kontinuierlichen übereinstimmen. Allerdings läßt sich aus den Ergebnissen des diskontinuierlichen Systems nicht erkennen, wie der Verlauf zwischen den Abtastpunkten aussieht (Bild 19c).
Deshalb ist es günstig, den zeitlichen Diskretisierungsabstand T, die Abtastperiode, möglichst klein zu wählen. Dabei ist „klein" immer im Verhältnis zur Systemdynamik, zu den Werten der Zeitkonstanten im System, zu verstehen.
Die Zeitkonstanten T_k eines Systems sind die negativen Kehrwerte seiner reellen Eigenwerte: $T_k = -1/p_k$.
Die Regelstrecke des Beispielsystems hat demzufolge die drei Zeitkonstanten $T_1 = 1/0.08 = 12.5$, $T_2 = 1/0.1118 = 8.94$, $T_3 = 1/0.2913 = 3.43$.

Zu dem reellen Eigenwert p_k gehört ein Anteil in der Zeitfunktion der Systembewegung

$$e^{p_k t} = e^{-t/T_k}.$$

Diskretisiert man die Zeitachse und setzt $t = kT$, dann ergibt sich daraus eine Folge e^{-kT/T_k}. Es sollen genügend viele Glieder dieser Folge vorhanden sein.
Wählt man im Beispielsystem $T = 1$, dann ergibt sich für die kleinste Zeitkonstante $T_3 = 3.43$ die Folge $e^{-k/3.43} = e^{-0.291k}$. Für $k = 10$ ist dieser Wert auf 5.4% abgesun-

ken, d. h., der Teilvorgang mit der kleinsten Zeitkonstante wird durch etwa zehn Punkte bis zu seinem Abklingen beschrieben, diejenigen mit den größeren Zeitkonstanten (T_1, T_2) durch entsprechend mehr. T = 1 ist für das Beispielsystem also völlig ausreichend.

3.1.2. Diskretisierung des Zustandsmodells

Die Zustandsgleichungen mit kontinuierlich ablaufender Zeit waren durch die Beziehungen

$$\frac{d\mathbf{q}(t)}{dt} = \mathbf{f}(\mathbf{q}(t), \mathbf{u}(t)) \tag{14}$$

$$\mathbf{x}(t) = \mathbf{g}(\mathbf{q}(t), \mathbf{u}(t)) \tag{16}$$

bzw. durch die linearen Gleichungen

$$\frac{d\mathbf{q}(t)}{dt} = \mathbf{A}\,\mathbf{q}(t) + \mathbf{B}\,\mathbf{u}(t) \tag{24}$$

$$\mathbf{x}(t) = \mathbf{C}\,\mathbf{q}(t) + \mathbf{D}\mathbf{u}(t) \tag{25}$$

gegeben. In den diskontinuierlichen Modellen treten an die Stelle der Funktionen $\mathbf{q}(t)$, $\mathbf{u}(t)$, $\mathbf{x}(t)$ die entsprechenden Folgen zeitdiskreter Werte $\mathbf{q}_k$, $\mathbf{u}_k$, $\mathbf{x}_k$.
Die dazugehörigen Modelle lauten im allgemeinen nichtlinearen Fall

$$\mathbf{q}_{k+1} = \mathbf{f}^*(\mathbf{q}_k, \mathbf{u}_k)$$
$$\mathbf{x}_k = \mathbf{g}(\mathbf{q}_k, \mathbf{u}_k)\,.$$

Im linearen Fall, der zukünftig ausschließlich betrachtet werden soll, lautet die zeitdiskrete Zustandsgleichung jetzt

$$\mathbf{q}_{k+1} = \mathbf{P}\,\mathbf{q}_k + \mathbf{R}\,\mathbf{u}_k\,, \tag{81}$$

während die Ausgabegleichung lautet

$$\mathbf{x}_k = \mathbf{C}\,\mathbf{q}_k + \mathbf{D}\,\mathbf{u}_k\,. \tag{82}$$

An den Matrizen der Ausgabegleichung ändert sich also durch die Diskretisierung der Zeit nichts, während in der Zustandsgleichung die neuen Matrizen **P** (Fundamentalmatrix) und **R** (Steuermatrix) auftauchen. Diese Matrizen berechnen sich aus den Matrizen **A** und **B** des kontinuierlichen Modells nach folgenden Beziehungen [8]:

$$\mathbf{P} = e^{\mathbf{A}T} = \mathbf{E} + \mathbf{A}T + \mathbf{A}^2\frac{T^2}{2} + \mathbf{A}^3\frac{T^3}{3!} + \ldots + \mathbf{A}^k\frac{T^k}{k!} + \ldots \tag{83}$$

$$\mathbf{R} = [(e^{\mathbf{A}T} - \mathbf{E})\mathbf{A}^{-1}]\mathbf{B} = \left[\mathbf{E}T + \mathbf{A}\frac{T^2}{2} + \mathbf{A}^2\frac{T^3}{3!} + \ldots + \mathbf{A}^k\frac{T^{k+1}}{(k+1)!} + \ldots\right]\mathbf{B}\,. \tag{84}$$

Die Beziehungen (83) und (84) stellen Potenzreihen von Matrizen dar. Dabei ist eine Matrizenpotenz genauso erklärt wie die Potenz einer skalaren Größe.

$$\left.\begin{aligned} a^k &= a \cdot a \cdot a \ldots a \\ A^k &= A \cdot A \cdot A \ldots A \end{aligned}\right\} \text{ (k Faktoren) .}$$

An sich sind (83) und (84) unendliche Reihen. Die Konvergenz dieser Reihen ist auf alle Fälle gesichert. Für eine praktische Rechnung wird man jedoch immer nur mit endlichen Anzahlen von Summanden arbeiten. Daher wird ein Abbruchkriterium benötigt. Es werden nur soviel Glieder berechnet und aufsummiert, bis folgende Abbruchbedingung erfüllt ist:

$$\| A^i \| \frac{T^i}{i!} \leqq \text{eps} . \tag{85}$$

Dabei bedeutet $\| A \|$ die Norm der Matrix **A**, die aus der Summe der Beträge aller ihrer Elemente gebildet werden kann:

$$\| A \| = \sum_{i=1}^{n} \sum_{j=1}^{n} | a_{ij} | . \tag{86}$$

Anstelle der durch (86) ausgedrückten Betragsnorm kann man auch eine quadratische Norm berechnen, bei der die Quadrate aller Matrizenelemente addiert werden.

Zur Berechnung von **P** und **R** auf dem Mikrorechner wird von den Beziehungen (83) bis (86) ausgegangen. Bild 20 zeigt den Programmablaufplan.

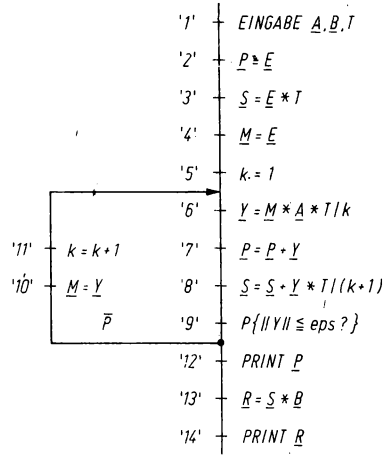


Bild 20. Programmablaufplan zur Berechnung von ***P*** *und* ***R***

M, **S** und **Y** sind Zwischenmatrizen. Dabei entspricht **Y** den Gliedern der Reihe (83). Im Rechenschritt k ist

$$Y_k = A^k \frac{T^k}{k!} .$$

M entspricht Y_{k-1} (Zeile '10'). Damit wird Y_k in Zeile '6' nach der rekursiven Beziehung

$$Y_k = Y_{k-1} \, A \, T/k$$

gebildet.

Die Steuermatrix **R** wird in Zeile '13' aus **S** * **B** berechnet. **S** entspricht der in (84) in eckigen Klammern stehenden Summe. Vergleicht man die einzelnen Glieder dieser Reihe mit denen der Reihe (83), dann ist jedes Glied mit einem Faktor T/(k + 1) multipliziert. Dies erfolgt in Zeile '8'. Die Norm ‖**Y**‖ in Zeile '9' muß nach (86) berechnet werden.

3.1.3. Rechenprogramm

Der Programmablaufplan nach Bild 20 kann ohne Schwierigkeiten in ein Rechenprogramm umgesetzt werden, das Bestandteil des Programms ZUST ist (s. Abschn. 3.2.).
Die Zeilen '2', '3', '4' des PAP sind in den Programmzeilen 17Ø, 18Ø zusammengefaßt. Die Matrizenoperationen der Zeilen '6' bis '8' werden durch Laufanweisungen über I und J in den Zeilen 2ØØ bis 23Ø realisiert. Als Abbruchschranke eps ist in Zeile 24Ø der Wert 0.01 fest vorgegeben. PAP-Zeilen '10' und '11' finden sich in den Programmzeilen 25Ø, 26Ø; Zeile '13' in den Zeilen 28Ø, 29Ø.
Die Anwendung dieses Rechenprogramms auf die Matrizen des kontinuierlichen Beispiels nach den Tafeln 4 und 5 ergab die in Tafel 11 zusammengestellten Ergeb-

Tafel 11. Matrizen des diskontinuierlichen Systems

Regelstrecke; n = 3

T = 1

$$\mathbf{P} = \begin{pmatrix} .9231 & 0 & 0 \\ .1483 & .777 & .04 \\ .008 & .0872 & .8644 \end{pmatrix} \qquad \mathbf{R} = \begin{pmatrix} .254 & 0 \\ .0207 & -.00141 \\ .00073 & -.06139 \end{pmatrix}$$

T = 2

$$\mathbf{P} = \begin{pmatrix} .8521 & 0 & 0 \\ .2525 & .6072 & .0655 \\ .02713 & .14325 & .7507 \end{pmatrix} \qquad \mathbf{R} = \begin{pmatrix} .4885 & 0 \\ .07453 & -.00495 \\ .0052 & -.1147 \end{pmatrix}$$

Regelkreis; n = 4

T = 1

$$\mathbf{P} = \begin{pmatrix} .8777 & -.5044 & .3219 & -.1874 \\ .1458 & .7341 & .06685 & -.0154 \\ .00788 & .0857 & .8654 & -.00054 \\ .02432 & .269 & -.1716 & .9983 \end{pmatrix} \qquad \mathbf{R} = \begin{pmatrix} 1.8448 & -.011 \\ .1491 & -.002026 \\ .005215 & -.0614 \\ -.984 & .005925 \end{pmatrix}$$

T = 2

$$\mathbf{P} = \begin{pmatrix} .69466 & -.8358 & .5596 & -.344 \\ .2351 & .4669 & .1566 & -.0541 \\ .02624 & .133 & .7572 & -.0038 \\ .08355 & .4388 & -.294 & .9881 \end{pmatrix} \qquad \mathbf{R} = \begin{pmatrix} 3.57 & -.0406 \\ .543 & -.00933 \\ .0376 & -.1148 \\ -1.8823 & .02157 \end{pmatrix}$$

nisse. Für T = 1 waren jeweils vier Glieder der Reihe (83) zu berechnen (k = 4); für T = 2 waren es fünf Glieder. Die Rechenzeiten betrugen

	T = 1	T = 2
n = 3	12.6/11.4	15.6/12.5
n = 4	25.3/20	32/24.

Die Zeitangaben gelten für ZX Spectrum bzw. KC 85/3.

3.2. Simulation des Zeitverhaltens

3.2.1. Lösung der diskontinuierlichen Zustandsgleichung

Die diskontinuierliche Zustandsgleichung (81) und die Ausgabegleichung (82) sind direkt für eine rekursive digitale Berechnung geeignet. Zur Lösung müssen dabei gegeben sein

- der Anfangszustand $\mathbf{q}(k = 0) = \mathbf{q}_0$
- die Folge der Eingangssignalwerte $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_k$.

Damit ergibt sich aus (82)

$$\mathbf{x}_0 = \mathbf{C}\,\mathbf{q}_0 + \mathbf{D}\,\mathbf{u}_0$$

und aus (81)

$$\mathbf{q}_1 = \mathbf{P}\,\mathbf{q}_0 + \mathbf{R}\,\mathbf{u}_0 .$$

Entsprechend folgt weiter

$$\mathbf{x}_1 = \mathbf{C}\,\mathbf{q}_1 + \mathbf{D}\,\mathbf{u}_1$$
$$\mathbf{q}_2 = \mathbf{P}\,\mathbf{q}_1 + \mathbf{R}\,\mathbf{u}_1 \text{ usw.}$$

Bild 21a zeigt den einfachen Programmablaufplan dazu.

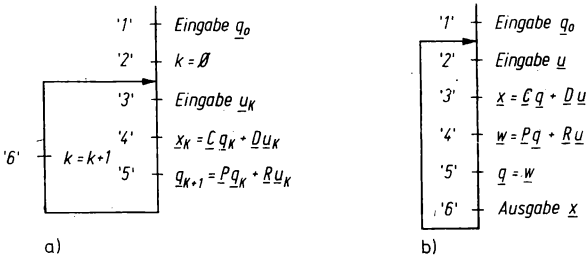


Bild 21. Programmablaufplan zur Lösung der diskontinuierlichen Zustandsgleichung

Nach diesem PAP werden in Zeile '5' die „neuen" Werte der Zustandkomponenten $q_{k+1,i}$ berechnet, und zwar aus den „alten" $q_{k,i}$. Dabei müssen die alten Komponenten $q_{k,1} \ldots q_{k,n}$ so lange verfügbar sein, bis die letzte neue Komponente $q_{k+1,n}$ berechnet worden ist.
Bei einer direkten Programmierung des PAP nach Bild 21a wäre das nicht gewährleistet, weil die Komponente $q_{k,i}$ sofort mit dem neu berechneten Wert $q_{k+1,i}$ über-

speichert werden würde. Dieses Überspeichern ist auszuschließen. Daher sind die neuen Komponenten $q_{k+1,i}$ zunächst in einem Zwischenvektor **w** zwischenzuspeichern, bis die Berechnung abgeschlossen ist.
Dagegen wird der Index k für die Rechnung nicht benötigt, wenn der jeweils berechnete Ausgangsvektor **x** unmittelbar ausgegeben wird. Das führt auf den PAP nach Bild 21b. Die Zeile '6', in der der in Zeile '3' berechnete **x**-Vektor ausgegeben wird, kann auch an anderer Stelle eingeordnet werden.
Für sekundäre Zwecke kann es allerdings angebracht sein, doch eine Größe k als Zählvariable mitzuführen, deren Wert bei jedem Durchlauf durch die Programmschleife um 1 erhöht wird – sei es, um nur eine bestimmte Anzahl von Werten zu berechnen, für Diagrammausgabe oder auch zur Erzeugung von Werten des Eingangssignals **u** im Rechenprogramm.

3.2.2. Eingangssignale

Nach Bild 21a und b wird in jedem Zyklus ein Wert des Eingangsvektors **u** (im einfachsten Fall: eine Eingangskomponente u) benötigt. Dies kann z.B. durch eine INPUT-Anweisung oder durch Lesen aus einer Datenliste erfolgen.
Für typische Eingangszeitverläufe können die Werte vom Rechner selbst erzeugt werden.

Sprungfunktion

Es ist $u_k = u_0 = \text{const}$ für alle k. u_0 ist die Sprungamplitude. Für diesen Fall liefert das Gleichungssystem (81), (82) eine exakte, d.h. in den Abtastpunkten mit der Lösung des kontinuierlichen Systems übereinstimmende Lösung – unabhängig davon, wie groß T gewählt wurde.

Sinusfunktion

Die kontinuierliche Funktion lautet

$$u(t) = u\hat{\varnothing} \sin(\omega t).$$

Durch Abtastung folgt daraus

$$u_k = u\hat{\varnothing} \sin(k\,\omega T).$$

Die Lösung von (81), die im Abschn. 3.2.1. beschrieben wurde, beruht auf der Voraussetzung, daß u zwischen den Abtastpunkten konstant ist, d. h. $u(t) = u_k$ für $kT \leqq t < (k+1)T$. Das bedeutet: Der Algorithmus berechnet die Lösung der Systemgleichung für einen Eingangszeitverlauf, der durch die im Bild 22 gezeigte Treppenfunktion gegeben ist, anstelle der ursprünglichen Sinusfunktion. Dadurch ergibt sich ein Fehler, der um so weniger ins Gewicht fällt, je kleiner die Abtastpe-

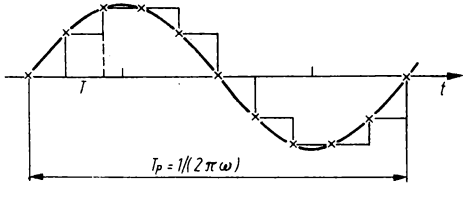


Bild 22. Abtastung einer Sinusfunktion

riode T im Verhältnis zur Periodendauer $T_p = 1/(2\pi\omega)$ des Sinussignals ist. Bei gegebener Kreisfrequenz ω ist demnach die Abtastperiodendauer T geeignet zu wählen. Im Programm ZUST (s. Abschn. 3.2.4.) wurde ein Grenzwert $T/T_p \leqq 1/8$, d. h. $\omega T \leqq \pi/4$ zugrunde gelegt.

Zufallsfunktion

Stochastische, vom Zufall abhängende Einflußgrößen treten oftmals auf. Die Nachbildung solcher Signale als Systemeingangsgrößen hat deshalb große Bedeutung.
Wie im Abschn. 1.6. gezeigt worden war, läßt sich ein beliebiges gefiltertes Rauschsignal x_R als Eingangssignal für ein dynamisches System dadurch gewinnen, daß man das Zustandsmodell des Systems ($\mathbf{A}$, $\mathbf{b}_u$, $\mathbf{C}$, $\mathbf{d}$) mit dem des Rauschfilters ($\mathbf{A}_R$, $\mathbf{b}_R$, $\mathbf{c}_R^T$) zu einem einzigen Modell zusammenfaßt (56).
Dieses Gesamtmodell ($\mathbf{A}^*$, $\mathbf{b}^*$, $\mathbf{C}^*$) ist in das Programm ZUST einzugeben.
Als Eingangssignal des Systems muß dann ein weißes Rauschsignal w verwendet werden (55).
Als kontinuierliches Signal läßt sich ein solches weißes Rauschen nicht darstellen. Für die hier betrachtete zeitdiskrete Lösung erhält man jedoch mit der BASIC-Funktion RND sehr leicht eine Folge von Werten, die als Abtastwerte eines weißen Rauschens aufgefaßt werden können.
So ergibt sich in der BASIC-Version des KC 85/3 bei jedem Aufruf

$$u\emptyset * (\mathrm{RND}(1) - \emptyset.5)$$

ein Wert aus einer zwischen $-u\emptyset/2$ und $+u\emptyset/2$ gleichverteilten, d. h. zentrierten Zufallszahlenfolge. (Die Varianz der Folge beträgt $\sigma_u^2 = u\emptyset^2/12$.)
Das so gewonnene Signal stellt das Äquivalent eines weißen Eingangs-Rauschsignals dar. Wird also der Befehl

$$\mathrm{LET\ U} = \mathrm{U}\emptyset * (\mathrm{RND}(1) - .5) \tag{87}$$

in der Zeile '2' des PAP nach Bild 21b angeordnet, dann wird damit die Systemreaktion auf ein gefiltertes Rauschsignal ermittelt, dessen Parameter durch das Rauschfilter gegeben sind.

3.2.3. Dialogführung; Programm ZUST

Ein anwenderfreundliches Programm verlangt oftmals viele Befehle, die nicht der Umsetzung der eigentlichen Algorithmen dienen, sondern dazu, den Dialog mit dem Nutzer möglichst bequem zu gestalten. Bereits die früher behandelten Programme FREQ (s. Abschn. 2.2.2.) und EWEPOL (s. Abschn. 2.3.4.) enthielten solche Dialogführungen, die bei der Beschreibung der Programme nicht weiter erläutert worden sind.
Hier soll als ein Beispiel etwas ausführlicher, aber keineswegs erschöpfend auf die Dialogführung im Rechenprogramm ZUST eingegangen werden. Dieses Programm soll folgende Anforderungen erfüllen:

1. Es sind auf der Basis eines kontinuierlichen Zustandsmodells die Matrizen **P** und **R** eines diskontinuierlichen Modells zu berechnen und auszugeben.
2. Bedarfsweise ist das dynamische Verhalten der durch das Zustandsmodell beschriebenen Systeme für verschiedene Eingangssignale zu ermitteln. Als Ein-

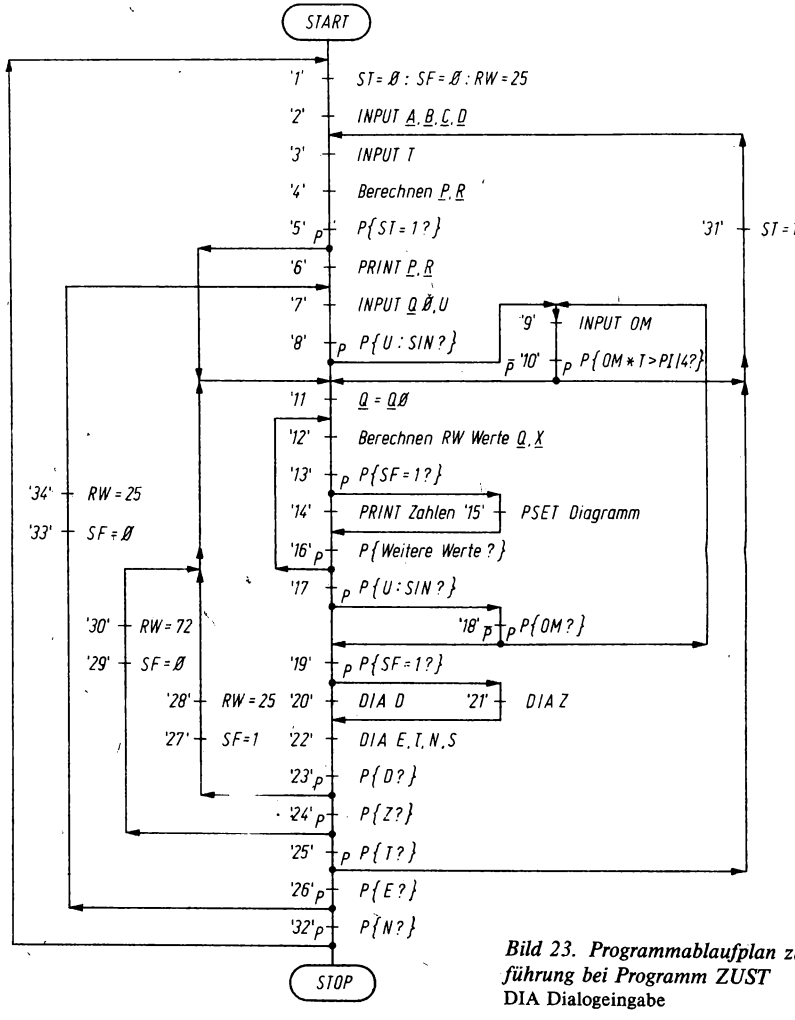


Bild 23. Programmablaufplan zur Dialogführung bei Programm ZUST
DIA Dialogeingabe

gangssignale sind vorzusehen: Sprungfunktionen, Sinusfunktionen, Zufallsfunktionen.

3. Die Rechenergebnisse sollen wahlweise als Tabellen oder als Diagramme der Zeitfunktionen dargestellt werden.
4. Der Programmnutzer soll vom Rechner über Bildschirm aufgefordert werden, die entsprechenden Eingaben zu machen.

Die algorithmische Lösung der Aufgaben ist durch die Abschnitte 3.1.2. und 3.2.1. beschrieben worden. Die dazu erforderlichen Operationen nehmen im Programmablaufplan (Bild 23) nur einen geringen Raum ein. Zeile '4' umfaßt die Operationen zur Berechnung von **P** und **R**, die im Abschn. 3.1.2. sowie im PAP Bild 20 dargestellt worden sind. Im Rechenprogramm ZUST entsprechen dem die Zeilen 15Ø bis 3ØØ.

```
 1Ø  REM EINGABEN
 2Ø  INPUT "FORMATE N, M, P ";N,M,P
 3Ø  LET ST = Ø: LET SF = Ø: LET RW = 25
 4Ø  DIM A(N,N), B(N,M), C(P,N), R(N,M), D(P,M), P(N,N), Y(N,N), S(N,N)
 5Ø  DIM M(N,N), Q(N), U(M), V(N), W(N), X(P)
 6Ø  FOR I = 1 TO N: FOR J = 1 TO N: PRINT"A(";I;J;")";
 7Ø  INPUT A(I,J): NEXT J: NEXT I: PRINT
 8Ø  FOR I = 1 TO N: FOR J = 1 TO M: PRINT"B(";I;J;")";
 9Ø  INPUT B(I,J): NEXT J: NEXT I
1ØØ  FOR I = 1 TO P: FOR J = 1 TO N: PRINT"C(";I;J;")";
11Ø  INPUT C(I,J): NEXT J: NEXT I: PRINT
12Ø  FOR I = 1 TO P: FOR J = 1 TO M: PRINT"D(";I;J;")";
13Ø  INPUT D(I,J): NEXT J: NEXT I
14Ø  INPUT "ZEITABSTAND T ";T
15Ø  REM BERECHNUNG FUNDAMENTALMATRIX
16Ø  LET K = 1
17Ø  FOR I = 1 TO N: FOR J = 1 TO N: LET P(I,J) = Ø: LET M(I,J) = Ø: LET S(I,J) = Ø:
     NEXT J
18Ø  LET P(I,I) = 1: LET M(I,I) = 1: LET S(I,I) = T: NEXT I
19Ø  LET NO = Ø
2ØØ  FOR I = 1 TO N: FOR J = 1 TO N: LET Y(I,J) = Ø
21Ø  FOR L = 1 TO N: LET Y(I,J) = Y(I,J) + M(I,L) * A(L,J) * T/K: NEXT L
22Ø  LET P(I,J) = P(I,J) + Y(I,J): LET S(I,J) = S(I,J) + Y(I,J) * T/(K + 1)
23Ø  LET NO = NO + ABS(Y(I,J)): NEXT J: NEXT I
24Ø  IF NO < = .Ø1 GOTO 28Ø
25Ø  LET K = K + 1: FOR I = 1 TO N: FOR J = 1 TO N
26Ø  LET M(I,J) = Y(I,J): NEXT J: NEXT I: GOTO 19Ø
27Ø  REM BERECHNUNG STEUERMATRIX
28Ø  FOR I = 1 TO N: FOR J = 1 TO M: LET R(I,J) = Ø
29Ø  FOR L = 1 TO N: LET R(I,J) = R(I,J) + S(I,L) * B(L,J): NEXT L
3ØØ  NEXT J: NEXT I: CLS: IF SF THEN GOSUB 1Ø5Ø
31Ø  IF ST GOTO 64Ø
32Ø  PRINT "FUNDAMENTALMATRIX P(T = ";T;")": PRINT
33Ø  FOR I = 1 TO N: FOR J = 1 TO N: PRINT TAB(11 * (J - 1));P(I,J);
34Ø  NEXT J: PRINT: NEXT I: PRINT
35Ø  PRINT "STEUERMATRIX R(T = ";T;")": PRINT
36Ø  FOR I = 1 TO N: FOR J = 1 TO M: PRINT TAB(11 * (J - 1));R(I,J);
37Ø  NEXT J: PRINT: NEXT I: PRINT
38Ø  PRINT "NEUER WERT FUER T?: 1"
39Ø  PRINT "ZEITVERLAUF ?       2"
4ØØ  PRINT "ENDE ?              3"
41Ø  INPUT AP: CLS
42Ø  ON AP GOTO 14Ø, 45Ø, 43Ø
43Ø  END
44Ø  REM EINGABEN FUER ZEITVERLAUF
45Ø  LET MAX = Ø: LET BP = Ø: IF P = 1 THEN LET V = 1: GOTO 47Ø
46Ø  PRINT "INDEX AUSGANGSGROESSE X(I), I = ?": INPUT V
47Ø  PRINT ANFANGSZUSTAND QØ = Ø: EINGABE Ø, SONST 1": INPUT AA
48Ø  IF AA = Ø THEN FOR I = 1 TO N: LET V(I) = Ø: NEXT I: GOTO 53Ø
49Ø  PRINT "ANFANGSZUSTAND";: FOR I = 1 TO N: PRINT "QØ(";I;")";
5ØØ  INPUT V(I): NEXT I: PRINT
```

```
51Ø  PRINT "EINGANGSGROESSEN = Ø: EINGABE Ø, SONST 1"
52Ø  INPUT BB: IF BB = Ø THEN LET R = 1: LET UØ = Ø: LET U(R) = Ø: CLS: GOTO 64Ø
53Ø  IF M = 1 THEN LET R = 1: GOTO 55Ø
54Ø  PRINT "INDEX EINGANGSGROESSE U(I)? I = ?": INPUT R
55Ø  PRINT "ZEITVERLAUF EINGANGSGROESSE U(";R;")": PRINT
56Ø  PRINT "SPRUNG : EINGABE 1"
57Ø  PRINT "SINUS :          2"
58Ø  PRINT "ZUFALL :         3"
59Ø  INPUT BP: CLS
6ØØ  INPUT "AMPLITUDE UØ ";UØ
61Ø  IF BP < > 2 GOTO 64Ø
62Ø  INPUT "OMEGA ";OM: CLS: IF T * OM < = PI/4 GOTO 64Ø
63Ø  PRINT "T = ";T;" ZU GROSS": LET ST = 1: GOTO 14Ø
64Ø  IF SF THEN GOSUB 1Ø5Ø: REM BERECHNUNG ZEITVERLAUF
65Ø  FOR I = TO N: LET Q(I) = V(I): NEXT I
66Ø  LET H = Ø
67Ø  FOR K = Ø TO RW: IF BP = 1 THEN LET U(R) = UØ: GOTO 7ØØ
68Ø  IF BP = 2 THEN LET U(R) = UØ * SIN(H * OM * T): GOTO 7ØØ
69Ø  IF BP = 3 THEN LET U(R) = 2 * UØ * (RND(1)-.5)
7ØØ  LET X(V) = Ø: FOR L = 1 TO N: LET X(V) = X(V) + C(V,L) * Q(L): NEXT L
71Ø  LET X(V) = X(V) + D(V,R) * U(R)
72Ø  IF ABS(X(V)) > MAX THEN LET MAX = ABS(X(V))
73Ø  IF SF = Ø THEN PRINT H * T, X(V): GOTO 76Ø
74Ø  LET XA = F * X(V) + 128: IF XA > 25Ø OR XA < 1Ø GOTO 93Ø
75Ø  PSET 4 * K + 25, XA
76Ø  FOR I = 1 TO N: LET W(I) = Ø: FOR J = 1 TO N
77Ø  LET W(I) = W(I) + P(I,J) * Q(J): NEXT J
78Ø  LET W(I) = W(I) + R(I,R) * U(R): NEXT I
79Ø  FOR I = 1 TO N: LET Q(I) = W(I): NEXT I
8ØØ  LET H = H + 1: NEXT K: IF SF = Ø THEN LET F = 75/MAX
81Ø  INPUT "WEITERE WERTE J/N ? ";E$: LET N$ = "N"
82Ø  IF E$ = "N" GOTO 85Ø
83Ø  CLS: IF SF THEN GOSUB 1Ø5Ø
84Ø  GOTO 67Ø
85Ø  CLS: IF BP = 2 THEN INPUT "NEUES OMEGA J/N ? ";N$
86Ø  IF N$ = "N" GOTO 88Ø
87Ø  GOTO 62Ø
88Ø  PRINT "EINGABEN": PRINT
89Ø  IF SF THEN PRINT "ZAHLEN :                                    Z"
9ØØ  IF SF = Ø THEN PRINT "DIAGRAMM :                                        D"
91Ø  PRINT "ANDERE EIN-/AUSGANGSGROESSEN :    E"
92Ø  PRINT "ANDERE ZEITABSTAENDE :            T"
93Ø  IF SF THEN PRINT "ANDERE SKALIERUNG :                 F"
94Ø  PRINT "NEUEINGABE :                      N"
95Ø  PRINT "SCHLUSS :                         S"
96Ø  INPUT B$: CLS
97Ø  IF B$ = "D" THEN LET SF = 1: LET RW = 72: GOTO 64Ø
98Ø  IF B$ = "Z" THEN LET SF = Ø: LET RW = 25: GOTO 64Ø
99Ø  IF B$ = "E" THEN LET SF = Ø: LET RW = 25: GOTO 45Ø
1ØØØ IF B$ = "T" THEN LET ST = 1: GOTO 14Ø
1Ø1Ø IF B$ = "F" THEN PRINT "SKALIERUNG F = ";F: PRINT "NEUER WERT ": IN-
     PUT F: CLS: GOTO 64Ø
```

```
1Ø2Ø  IF BØ = "N" THEN RUN
1Ø3Ø  END
1Ø4Ø  REM ZEICHNEN KOORDINATENSYSTEM
1Ø5Ø  LINE 25,128,315,128: LINE 25 28,25,23Ø
1Ø6Ø  FOR X = 65 TO 3Ø5 STEP 4Ø: FOR Y = 129 TO 131: PSET X,Y: NEXT Y: NEXT X
1Ø7Ø  FOR Y = 28 TO 228 STEP 2Ø: FOR X = 23 TO 26: PSET X,Y: NEXT X: NEXT Y
1Ø8Ø  FOR I = 1 TO 7: PRINT AT(16, 5*I + 1); 1Ø * I: NEXT I
1Ø9Ø  PRINT AT(2,1);1ØØ/F: PRINT AT(29,1);-1ØØ/F
11ØØ  PRINT AT(17,35);"T": PRINT AT(3,32);"T = ";T: PRINT AT(16,2);"Ø"
111Ø  RETURN
```

Zeile '12' des PAP Bild 23 umfaßt die Berechnung einer Folge von jeweils RW Werten des Zustandsvektors **q** und Ausgangsvektors **x** nach den Algorithmen von Abschn. 3.2.1.; PAP Bild 21b, Programm ZUST: Zeilen 66Ø bis 8ØØ.
Die Eingabe der Modellmatrizen (Zeile '2', Bild 23) wird in den Programmzeilen 6Ø bis 13Ø verwirklicht. Vorher hat der Nutzer auf Anforderung in Zeile 2Ø die Formate N (Anzahl der Zustandsgrößen), M (Anzahl der Eingangsgrößen) P (Anzahl der Ausgangsgrößen) einzugeben. Ist $M > 1$ bzw. $P > 1$, wird später vom Nutzer eine Festlegung gefordert, für welche der Ausgangsgrößen X(V) bzw. Eingangsgrößen U(R) die Berechnung durchgeführt bzw. Ergebnisausgabe vorgenommen werden sollen (Zeilen 46Ø und 54Ø). Ist $M = 1$ bzw. $P = 1$, werden diese Abfragen natürlich übergangen (Zeile 45Ø bzw. 53Ø).
Eine notwendige Eingabe ist die der Abtastzeit T (Zeile '3' bzw. 14Ø). Diese Eingabe kann zu verschiedenen Anlässen notwendig werden:

1. nach Start des Programms und Modelleingabe
2. bei Berechnung der Reaktion auf Sinus-Eingangssignale der Kreisfrequenz ω, wenn das ursprünglich gewählte T nicht der Bedingung $\omega T \leqq \pi/4$ genügt (vgl. Abschn. 3.2.2.)
   (Zeile '10' bzw. 62Ø); dann ist ein kleinerer Wert für T einzugeben
3. nach Berechnung eines Satzes von RW Werten, wenn danach der Wunsch nach einer größeren oder kleineren Abtastzeit aufkommt (Zeile '25' bzw. 1ØØØ).

Nach jeder Eingabe bzw. Änderung von T müssen die Matrizen **P** und **R** neu berechnet werden. Ihr Ausdruck soll jedoch nur erfolgen, wenn ein neues Modell eingegeben wurde (Fall 1) oder wenn das Programm ZUST dazu verwendet werden soll, **P** und **R** zu berechnen und auszudrucken (Dialogeingabe in Zeile 38Ø). Wird dagegen im Zuge der Berechnung von Zeitfunktionen die Abtastzeit T geändert (Fälle 2 und 3), dann besteht keine Notwendigkeit, **P** und **R** jedesmal auszudrukken. Daher wird die Ausgabe durch den Selektor ST gesteuert. $ST = Ø$ bedeutet Ausgabe. ST wird in den Zeilen '1' und '31' (3Ø, 63Ø, 1ØØØ) festgelegt bzw. verändert.
Ein weiterer Selektor SF legt die Form der Ergebnisdarstellung als Zahlentabelle ($SF = Ø$) oder Diagramm ($SF = 1$) fest. Beim ersten Durchlauf werden immer Zahlentabellen ausgegeben (Zeile '14'), und zwar jeweils $RW = 25$ Werte, um gerade einen Bildschirm zu füllen (s. Zeile '1'). Die Bildschirmtabelle gibt die Werte der Zeit kT und der ausgewählten Ausgangsgröße X(V) an (Zeile 73Ø).
Nach Darstellung von RW Werten besteht die Möglichkeit, einen weiteren Satz von wiederum RW Werten ausgeben zu lassen (Dialogeingabe Zeile '16' bzw. 81Ø) oder in den weiteren Dialog einzutreten (Zeile '22').

Ist SF = 1 gesetzt, werden RW = 72 Werte als Diagramm ausgegeben (Zeile '15' bzw. 74Ø, 75Ø). Dazu wird im Unterprogramm ab Zeile 1Ø5Ø ein Koordinatensystem gezeichnet und beschriftet. Der Maßstabsfaktor F, der den Ordinatenmaßstab der Diagrammdarstellung festlegt (Zeile 74Ø), wird automatisch aus dem größten Wert von X(V) berechnet, der in der jeweils vorangegangenen Tabellendarstellung aufgetreten ist (Zeilen 72Ø, 8ØØ). Deshalb erfolgt nach jeder wesentlichen Änderung der Programmabarbeitung, außer der von T, zunächst ein automatisches Umschalten auf Tabellenausgabe, SF = Ø (Zeilen '1', '29', '33'). Die Eingaben in Zeile '7' (Zeilen 49Ø ... 6ØØ) umfassen neben der schon beschriebenen Festlegung der Indizes von Ausgangs- und Eingangsgröße den Anfangszustand $\mathbf{q}_0$, abgespeichert für wiederholten Zugriff im Feld V(I), sowie Angaben zur gewünschten Eingangsgröße U(R) (Menüauswahl Sprung/Sinus/Zufall, Zeilen 56Ø bis 59Ø). Bei „Sinus" wird ein Selektor BP = 2 gesetzt, um die zusätzliche Eingabe der Kreisfrequenz ω zu ermöglichen (Zeile '9' (62Ø) bzw. Neueingabe '18' (85Ø)). Bei der Dialogeingabe von $\mathbf{q}_0$ und u werden dem Nutzer nur relevante Fragen gestellt. Soll beispielsweise nur die Systemreaktion auf einen von null verschiedenen Anfangszustand $\mathbf{q}_0$ ermittelt werden, aber keine Eingangsgröße auftreten, dann erübrigen sich Anfragen nach Signaltyp und -amplitude (Überspringen der Zeilen 53Ø bis 63Ø).

Nach Abarbeiten des Programms, beim ersten Durchlauf dem Ausdruck von 25 Tabellenwerten (ggf. wiederholt), werden in der Dialogeingabe '20' (später dann, bei vorangegangener Diagrammausgabe, SF = 1, in der Dialogeingabezeile '21') sowie den weiteren Dialogeingabemöglichkeiten '22' die Alternativen Diagrammdarstellung D (bei vorangegangener Zahlenausgabe) bzw. Zahlenausgabe Z (bei vorangegangener Diagrammausgabe), Änderung der Abtastzeit T, Änderung E von $\mathbf{q}_0$ bzw. u, Neueingabe N von Prozeßmodell und schließlich Abschluß S der Sitzung zur Auswahl gestellt, die zu den von den Zeilen '23', '24', '25', '26' und '32' ausgehenden Programmfortsetzungen führen. Im Programm entsprechen dem die Zeilen 88Ø bis 1Ø3Ø.

Eine weitere Möglichkeit, im Bild 23 nicht dargestellt, besteht in der Änderung des Ordinatenmaßstabs F (Zeile 93Ø), die auch vom Programm dann angesprungen wird, wenn die Kurve in der Diagrammdarstellung die Bildschirmfläche zu überschreiten droht (Zeile 74Ø).

Bild 24 zeigt als Anwendungsbeispiel einen Regelvorgang, der für das Beispiel nach Bild 1 berechnet worden ist.

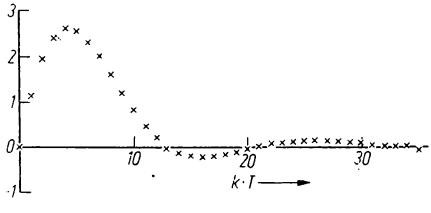


Bild 24. Werteverlauf der Regelgröße des Beispielsystems nach Sprungstörung und PI-Regler mit Einstellwerten nach Tafel 5

4. Diskontinuierliche Klemmenmodelle

4.1. Herleitung der Modelle

4.1.1. Die z-Übertragungsfunktion

Das meiste, was im Abschn. 2.1.1. zu den Klemmenmodellen kontinuierlicher Systeme gesagt worden ist, gilt auch sinngemäß für die Klemmenmodelle diskontinuierlicher Systeme.
Der Hauptunterschied wird durch den zeitdiskreten Charakter bestimmt. Klemmenmodelle kontinuierlicher Systeme stellen die Zusammenhänge zwischen den kontinuierlichen Zeitfunktionen der Eingangsgröße u(t) und der Ausgangsgröße x(t) her. Sie haben die Form von Differentialgleichungen mit Ableitungen nach der Zeit (in diesem Band nicht weiter betrachtet) oder von Übertragungsfunktionen G(p) (s. Gl. (57)). Dabei ist p der Laplace-Operator, der auch als Differentiationsoperator für d/dt aufgefaßt werden kann [7].
Klemmenmodelle diskontinuierlicher Systeme vermitteln die Zusammenhänge zwischen Wertefolgen der Eingangsgröße u_k und der Ausgangsgröße x_k. Die allgemeine Form eines solchen Klemmenmodells ist eine Differenzengleichung.

$$h_n x_{k-n} + h_{n-1} x_{k-n+1} + \ldots + h_1 x_{k-1} + h_0 x_k = g_0 u_k + g_1 u_{k-1} + \ldots + g_n u_{k-n}; \quad (88)$$

$k = 0, 1, 2, \ldots$ entspricht den Abschnitten kT auf der Zeitachse (s. Bild 19b).

x_{k-i} bedeutet demnach den Wert von x zu einem Abtastpunkt, der i Abschnitte vor dem gerade betrachteten Punkt k liegt. Allgemein wird vereinbart, daß in den Folgen keine negativen Indizes auftreten sollen, d. h., es ist $x_{k-i} = 0$ und $u_{k-i} = 0$ für $k < i$.
Gl. (88) ist direkt nach einem numerischen rekursiven Verfahren zu lösen (vgl. Abschn. 4.2.). Zur Lösung müssen gegeben sein

- die Werte der Eingangsfolge u_k; $k = 0, 1, 2, \ldots$ Das entspricht dem Zeitverlauf der Eingangsgröße u(t); $t \geqq 0$ bei einem kontinuierlichen System.
- die Anfangsglieder $x_0, x_1, \ldots, x_{n-1}$ der Ausgangsfolge. Dies entspricht den Anfangswerten $x(0), \dot{x}(0), \ldots, x^{(n-1)}(0)$ bei einem kontinuierlichen System.

Ein Klemmenmodell der Form (88) ist auch als Übertragungsfunktion G(z) darstellbar. Dazu wird die z-Transformation angewendet [8]. Für eine numerische Berechnung der Systemantworten wird von der z-Transformation jedoch nur eine einzige Eigenschaft benötigt: z ist ein Verschiebeoperator, der die Indizes in den Folgen erhöht bzw. erniedrigt. Eine Multiplikation mit z bedeutet eine Indexerhöhung um 1, eine Multiplikation mit z^i eine Indexerhöhung um i. Ebenso bedeutet eine Multiplikation mit z^{-1} eine Indexerniedrigung um 1, eine Multiplikation mit z^{-i} eine Indexverringerung um i.
Wenn symbolisch geschrieben werden soll

$$X(z) \triangleq \{x_k\} \qquad \text{(z-Transformierte der Folge } x_k),$$

dann gilt einfach

$$z^{-i} X(z) \triangleq \{x_{k-i}\}.$$

Damit kann man für (88) auch symbolisch schreiben

$$(h_n z^{-n} + h_{n-1} z^{n-1} + \ldots + h_1 z^{-1} + h_0)\, X(z) = (g_0 + g_1 z^{-1} + \ldots + g_n z^{-n})\, U(z).$$

Die Größen X(z) bzw. U(z) wurden dabei auf der linken bzw. rechten Seite der Gleichung jeweils ausgeklammert.
Für die Übertragungsfunktion G(z) läßt sich daraus ablesen

$$G(z) = \frac{X(z)}{U(z)} = \frac{g_0 + g_1 z^{-1} + \ldots + g_n z^{-n}}{h_0 + h_1 z^{-1} + \ldots + h_n z^{-n}}. \tag{89}$$

Durch Erweitern mit z^n erhält man eine gleichwertige Form, in der nur positive Potenzen von z auftreten:

$$G(z) = \frac{g_0 z^n + g_1 z^{n-1} + \ldots g_n}{h_0 z^n + h_1 z^{n-1} + \ldots h_n}. \tag{90}$$

4.1.2. Herleitung aus dem diskontinuierlichen Zustandsmodell

Das Zustandsmodell (81), (82) erhält für jeweils eine einzige Eingangsgröße u und Ausgangsgröße x die Form

$$\mathbf{q}_{k+1} = \mathbf{P}\,\mathbf{q}_k + \mathbf{r}\,u_k \tag{91}$$
$$x_k = \mathbf{c}^T \mathbf{q}_k + d\,u_k. \tag{92}$$

Werden die z-Transformierten der Folgen $\mathbf{q}_k$, u_k und x_k wieder mit Großbuchstaben $\mathbf{Q}(z)$, U(z), X(z) gekennzeichnet, dann erhält man, wenn der Anfangszustand $\mathbf{q}_0 = \mathbf{0}$ ist, aus (91) und (92) die beiden Gleichungen

$$z\,\mathbf{Q}(z) = \mathbf{P}\,\mathbf{Q}(z) + \mathbf{r}\,U(z) \tag{93}$$
$$X(z) = \mathbf{c}^T \mathbf{Q}(z) + d\,U(z). \tag{94}$$

In völliger Analogie zur Vorgehensweise im Abschn. 2.1.1., Gln. (58), (59) wird Gl. (93) nach $\mathbf{Q}(z)$ aufgelöst und das Ergebnis in Gl. (94) eingesetzt. Man erhält

$$z\,\mathbf{Q}(z) - \mathbf{P}\,\mathbf{Q}(z) = (z\,\mathbf{E} - \mathbf{P})\mathbf{Q}(z) = \mathbf{r}\,U(z)$$
$$\mathbf{Q}(z) = (z\,\mathbf{E} - \mathbf{P})^{-1}\,\mathbf{r}\,U(z)$$
$$X(z) = [\mathbf{c}^T (z\,\mathbf{E} - \mathbf{P})^{-1}\,\mathbf{r} + d]\,U(z)$$
$$G(z) = \frac{X(z)}{U(z)} = \mathbf{c}^T (z\,\mathbf{E} - \mathbf{P})^{-1}\,\mathbf{r} + d. \tag{95}$$

Gl. (95) entspricht völlig (60), mit der die p-Übertragungsfunktion aus **A**, **b**, $\mathbf{c}^T$ und d berechnet wurde. Folgende Entsprechungen bestehen:
- Differentialoperator p/Verschiebeoperator z
- Systemmatrix **A**/Fundamentalmatrix **P**
- Steuervektor **b**/Steuervektor **r**.

G(z) kann deshalb direkt durch Anwendung des Faddejew-Algorithmus (s. Abschn. 2.1.2.) bzw. des entsprechenden Teils des Programms FREQ (s. Abschn. 2.1.3.) mit den Eingaben **P** anstelle **A**, **r** anstelle **b** ermittelt werden. Ausgegeben werden die Zähler- und Nennerkoeffizienten der z-Übertragungsfunktion

in der Form (90). Natürlich ist bei der Bildschirmausgabe im Programm FREQ (Zeilen 54Ø bis 62Ø) in Gedanken „z↑" anstelle von „p↑" zu setzen. Auf diese Weise wurden aus dem diskontinuierlichen Zustandsmodell der Regelstrecke mit den Matrizen nach Tafel 10 die z-Übertragungsfunktionen G(z) erhalten, deren Koeffizientenwerte von der Abtastperiodendauer T abhängen. Mit dem Programm FREQ erhält man
für T = 1

$$G_u(z) = \frac{-0.005226 - 0.0002834\,z + 0.006482\,z^2}{-0.61677 + 2.1833\,z - 2.5645\,z^2 + z^3}$$

und für T = 2

$$G_u(z) = \frac{-0.01482 - 1.8329\,z + 0.02283\,z^2}{-0.3804 + 1.6035\,z - 2.21\,z^2 + z^3}.$$

(Bemerkt sei noch, daß die Anwendung der weiteren Teile des Programms FREQ auf die diskontinuierlichen Modelle keinen Sinn ergibt.)

4.1.3. Herleitung aus der kontinuierlichen Übertragungsfunktion

Darstellungen der Systemdynamik als Übertragungsfunktionen der Form (57) sind sehr verbreitet. Einfache Übertragungsglieder (Verzögerungsglied, Schwingungsglied, I-Glied, DT_1-Glied usw.) werden oftmals durch die Parameter ihrer Übertragungsfunktion charakterisiert.
Daher besteht ein Bedürfnis, das Zeitverhalten eines Systems, von dem die p-Übertragungsfunktion G(p) bekannt ist, digital, d.h. punktweise zu berechnen. Der Weg dazu führt über die z-Übertragungsfunktion G(z) auf eine Differenzengleichung der Form (88) und deren Lösung (s. Abschn. 4.2.).
Ein exakter Weg, G(z) aus G(p) zu gewinnen, ist leider etwas kompliziert und vor allem kaum algorithmierbar. Deshalb wird hier ein Näherungsansatz gemacht, der für kleine T (im Verhältnis zu den Zeitkonstanten des Systems; vgl. Abschn. 3.1.1.) gute Ergebnisse bringt [8]. Dabei erhält man die z-Übertragungsfunktion G(z), indem man einfach in G(p) anstelle p setzt

$$p = \frac{2}{T}\,\frac{z-1}{z+1}. \tag{96}$$

Der Anschaulichkeit halber sei die Rechnung für eine Übertragungsfunktion der Ordnung n = 2 durchgeführt. G(p) lautet hierfür

$$G(p) = \frac{b_0 + b_1 p + b_2 p^2}{a_0 + a_1 p + a_2 p^2}.$$

Einsetzen von (96) und Erweitern mit $(z+1)^2$ ergibt

$$G(z) = \frac{b_0(z+1)^2 + b_1\frac{2}{T}(z-1)(z+1) + b_2\frac{4}{T^2}(z-1)^2}{a_0(z+1)^2 + a_1\frac{2}{T}(z-1)(z+1) + a_2\frac{4}{T^2}(z-1)^2}.$$

Ausmultiplizieren und Ordnen nach Potenzen von z in Zähler und Nenner bringt

$$G(z) = \frac{b_0 - \frac{2b_1}{T} + \frac{4b_2}{T^2} + \left(2b_0 - \frac{8}{T^2} b_2\right) z + \left(b_0 + \frac{2}{T} b_1 + \frac{4}{T^2} b_2\right) z^2}{a_0 - \frac{2a_1}{T} + \frac{4a_2}{T^2} + \left(2a_0 - \frac{8}{T^2} a_2\right) z + \left(a_0 + \frac{2}{T} a_1 + \frac{4}{T^2} a_2\right) z^2}.$$

Wie man bemerkt, werden die Ausdrücke in Zähler und Nenner völlig analog gebildet. Dividiert man den erhaltenen Ausdruck durch z^2, dann läßt sich das Ergebnis in Form einer z-Übertragungsfunktion (89) schreiben:

$$G(z) = \frac{g_2 z^{-2} + g_1 z^{-1} + g_0}{h_2 z^{-2} + h_1 z^{-1} + h_0}, \tag{97}$$

und man erkennt das Bildungsgesetz für die Koeffizienten im Zähler

$$\begin{aligned} g_2 &= b_0 - 2\frac{b_1}{T} + 4\frac{b_2}{T^2} \\ g_1 &= 2b_0 - 8\frac{b_2}{T^2} \\ g_0 &= b_0 + 2\frac{b_1}{T} + 4\frac{b_2}{T^2}. \end{aligned} \tag{98}$$

Für die Nennerkoeffizienten $h_2 \ldots h_0$ gelten die gleichen Beziehungen, wobei a_i anstelle der b_i zu setzen ist.
Die beispielhaft für $n = 2$ abgeleiteten Beziehungen (98) lassen sich auf beliebige n ausdehnen. Allgemein läßt sich schreiben

$$g_i = \sum_{j=0}^{n} k_{ij} \frac{b_j}{T^j}; \qquad i = 0 \ldots n \tag{99}$$

$$h_i = \sum_{j=0}^{n} k_{ij} \frac{a_j}{T^j}. \tag{100}$$

Die Koeffizienten k_{ij} können zu einer Matrix zusammengefaßt werden, die für $n = 2$ folgende Gestalt hat:

$$\mathbf{K} = \begin{pmatrix} 1 & 2 & 4 \\ 2 & 0 & -8 \\ 1 & -2 & 4 \end{pmatrix}.$$

Tafel 12. Matrix der Koeffizienten zur Berechnung eines diskontinuierlichen aus einem kontinuierlichen Klemmenmodell ; n = 5

$$\mathbf{K} = \begin{pmatrix} 1 & 2 & 4 & 8 & 16 & 32 \\ 5 & 6 & 4 & -8 & -48 & -160 \\ 10 & 4 & -8 & -16 & 32 & 320 \\ 10 & -4 & -8 & 16 & 32 & -320 \\ 5 & -6 & 4 & 8 & -48 & 160 \\ 1 & -2 & 4 & -8 & 16 & -32 \end{pmatrix}$$

Für andere n lassen sich die Koeffizienten k_{ij} in analoger Weise berechnen. Tafel 12 zeigt das Ergebnis für n = 5.
Wendet man die Beziehungen (98) auf die Übertragungsfunktion eines PT_1-Glieds an mit

$$G(p) = \frac{10}{p+1} \quad \text{(Abtastperiode T = 0.2)},$$

dann erhält man folgende z-Übertragungsfunktion:

$$G(z) = \frac{10 + 20z^{-1} + 10z^{-2}}{11 + 2z^{-1} - 9z^{-2}}, \tag{101}$$

also eine z-Übertragungsfunktion der Ordnung n = 2, obwohl das Originalsystem nur 1. Ordnung war. Eine genauere Betrachtung von (101) zeigt aber, daß Zähler und Nenner eine gemeinsame Nullstelle bei $z^{-1} = -1$ haben. Die entsprechenden Wurzelfaktoren können herausgekürzt werden, und man erhält

$$G(z) = \frac{10 + 10z^{-1}}{11 - 9z^{-1}}. \tag{102}$$

Man kann also mit den Beziehungen, die für größere n gelten, auch die Koeffizienten von Übertragungsfunktionen korrekt berechnen, die geringerer Ordnung sind. Damit ist es nicht notwendig, etwa für jedes n einen Satz von Koeffizienten k_{ij} für die Beziehungen (99), (100) auszurechnen.
Allerdings kann der eben durchgeführte Kürzungsprozeß gemeinsamer Zähler- und Nennernullstellen vom Rechner nicht automatisch vorgenommen werden. Der Rechner würde im später zu beschreibenden Programm KLEMM somit nicht mit der gekürzten Form (102), sondern mit der Originalform (101) rechnen.
Dem Zeitverlauf der Simulationsergebnisse ist das zunächst überhaupt nicht anzusehen. Ist aber der Unterschied zwischen möglicher Systemordnung und in Anspruch genommener sehr groß (das Programm, mit dem man Systeme 5. Ordnung nachbilden könnte, wird benutzt, um ein PT_1-Glied zu simulieren), dann können sich für größere k numerische Instabilitäten bemerkbar machen. (Das Auftreten solcher Instabilitäten hängt auch von der im Rechner benutzten Arithmetik ab.)
Deshalb werden im später beschriebenen Programm KLEMM zwei Klassen von Systemordnungen unterschieden; zur Nachbildung von Systemen 1. und 2. Ordnung werden die Koeffizienten nach (98) verwendet, während für die 3. bis 5. Ordnung die Gln. (99), (100) mit den Koeffizienten nach Tafel 12 Anwendung finden.

4.2. Lösung der Modellgleichung

4.2.1. Rekursive Lösung der Differenzengleichung

Liegt eine z-Übertragungsfunktion der Form (89) und eine Folge von Eingangssignalwerten u_k vor (die Anfangswerte der Ausgangsfolge werden als 0 vorausgesetzt), dann wird zur numerischen Lösung auf die Differenzengleichung (88) zurückgegangen. Diese Gleichung kann nach x_k aufgelöst werden

$$x_k = \frac{1}{h_0}\left[(g_0u_k + g_1u_{k-1} + \ldots + g_nu_{k-n} - h_1x_{k-1} - h_2x_{k-2} - \ldots - h_nx_{k-n})\right].$$

Mit der Schreibweise der z-Transformation erhält diese Gleichung die folgende symbolische Form:

$$X(z) = \frac{1}{h_0}\left[(g_0 + g_1 z^{-1} + \ldots + g_n z^{-n})\, U(z) - (h_1 z^{-1} + h_2 z^{-2} + \ldots + h_n z^{-n})\, X(z)\right]. \tag{103}$$

Der Wert der Ausgangsfolge x_k im Tastzeitpunkt k ergibt sich demnach aus dem Wert der Eingangsfolge u_k im gleichen Zeitpunkt, den diesem gegenüber um 1, 2, ..., n Tastpunkte zurückliegenden Werten u_{k-1}, u_{k-2}, ..., u_{k-n} sowie aus den jeweils zurückliegenden Werten der Ausgangsfolge selber ($x_{k-1} \ldots x_{k-n}$).
Diese zurückliegenden Werte der Eingangs- und Ausgangsfolge müssen abgespeichert werden. Beim Übergang auf den folgenden Tastzeitpunkt k+1 muß umgespeichert werden.
Der Speicheraufwand läßt sich verringern, wenn man (103) anders ordnet. Schreibt man zunächst

$$X = \frac{1}{h_0}\left[g_0 U + z^{-1}(g_1 U - h_1 X) + z^{-2}(g_2 U - h_2 X) + \ldots + z^{-n}(g_n U - h_n X)\right],$$

dann kann man diese Gleichung auch folgendermaßen darstellen:

$$X = \frac{1}{h_0}\left[g_0 U + z^{-1}(g_1 U - h_1 X + z^{-1}(g_2 U - h_2 X + z^{-1}(g_3 U - h_3 X + \ldots + z^{-1}(g_n U - h_n X)\ldots)))\right] \tag{104}$$

Diese Gleichung (104) läßt sich durch einen Programmablaufplan nach Bild 25 darstellen, wenn man sich erinnert, daß eine Multiplikation mit z^{-1} jeweils eine Verzögerung um einen Takt bedeutet. Insgesamt treten n zu verzögernde Ausdrücke auf, die in Speicherzellen untergebracht werden müssen. Dafür werden Veränderliche $q_1 \ldots q_n$ festgelegt. Insgesamt benötigt man für ein System n-ter Ordnung n+1 Speicherplätze. Eine Indizierung der Glieder u_k und x_k ist nicht erforderlich (vgl. Bild 21a und b), wenn in jedem Rechenzyklus der jeweils aktuelle Wert von u_k eingegeben wird (INPUT, READ oder im Programm erzeugt) und der berechnete Wert von x durch PRINT, PSET oder einen ähnlichen Befehl ausgegeben wird.

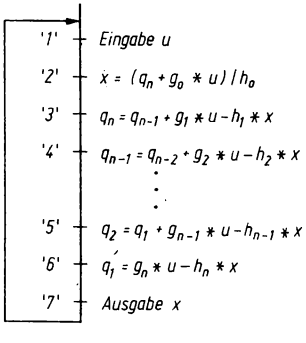


Bild 25. Programmablaufplan zur Lösung der Differenzengleichung (88)

4.2.2. Rechenprogramm KLEMM

Das Rechenprogramm KLEMM dient der Simulation des Zeitverhaltens (Reaktion auf Sprungfunktionen, Sinusfunktionen und Zufallsfunktionen) von Systemen, bei denen ein Klemmenmodell in Form einer Übertragungsfunktion G(p) gegeben ist. Als Eingaben erhält das Programm die Koeffizienten des Zähler- und Nennerpolynoms von G(p). Unterschieden werden zwei Klassen:

- Ordnung $n < 3$
- Ordnung $n < 6$.

Programm KLEMM

```
1Ø  REM BERECHNUNG ZEITVERHALTEN AUS KLEMMENMODELL
2Ø  INPUT "ORDNUNG N (< 6 !)";N
3Ø  IF N < = 2 THEN LET N = 3: RESTORE 9Ø: GOTO 5Ø
4Ø  LET N = 6
5Ø  CLS: DIM F(N,2),K(N,N),M(N,2),T(N),Q(N-1)
6Ø  FOR I = 1 TO N: FOR J = 1 TO N: READ K(I,J): NEXT J: NEXT I
7Ø  DATA 1,2,4,8,16,32,5,6,4,-8,-48,-16Ø,1Ø,4,-8,-16,32,32Ø
8Ø  DATA 10,-4,-8,16,32,-32Ø,5,-6,4,8,-48,16Ø,1,-2,4,-8,16,-32
9Ø  DATA 1,2,4,2,Ø,-8,1,-2,4
1ØØ LET ST = Ø: LET SF = Ø: LET RW = 25: PRINT "ZAEHLERKOEFFIZIENTEN":
    PRINT: PRINT
11Ø FOR I = Ø TO N-1: PRINT "P↑";I;: INPUT F(I + 1,1): NEXT I
12Ø PRINT: PRINT: PRINT "NENNERKOEFFIZIENTEN": PRINT: PRINT
13Ø FOR I = Ø TO N-1: PRINT "P↑";I;: INPUT F(I + 1,2): NEXT I
14Ø PRINT: PRINT: INPUT "TASTZEIT ";T
15Ø FOR I = Ø TO N-1: LET T(I + 1) = T↑I: NEXT I
16Ø FOR J = 1 TO 2: FOR I = 1 TO N: LET M(I,J) = Ø
17Ø FOR L = 1 TO N: LET M(I,J) = M(I,J) + K(I,L) * F(L,J)/T(L)
18Ø NEXT L: NEXT I: NEXT J: CLS
19Ø IF ST GOTO 29Ø
2ØØ PRINT "ZEITVERLAUF EINGANGSGROESSE": PRINT
21Ø PRINT "SPRUNGFUNKTION:   (1)": PRINT
22Ø PRINT "SINUS:            (2)": PRINT
23Ø PRINT "ZUFALL:           (3)": PRINT
24Ø INPUT "EINGABE →          ";BP: PRINT
25Ø INPUT "EINGANGSAMPLITUDE ";UØ: CLS
26Ø IF BP < > 2 GOTO 29Ø
27Ø INPUT "OMEGA";OM: CLS
28Ø IF OM * T > PI/4 THEN PRINT "T = ";T;" ZU GROSS": LET ST = 1: GOTO 14Ø
29Ø IF SF THEN GOSUB 74Ø
3ØØ FOR I = 1 TO N-1: LET Q(I) = Ø: NEXT I
31Ø LET H = Ø: LET XM = Ø
32Ø FOR K = Ø TO RW
33Ø IF BP = 1 THEN LET U = UØ: GOTO 36Ø
34Ø IF BP = 2 THEN LET U = UØ * SIN(H * OM * T): GOTO 36Ø
35Ø LET U = 2 * UØ * (RND(1) - .5)
36Ø LET X = (Q(N - 1) + M(1,1) * U)/M(1,2)
37Ø IF N = 6 GOTO 41Ø
38Ø LET Q(2) = Q(1) - M(2,2) * X + M(2,1) * U
39Ø LET Q(1) = -M(3,2) * X + M(3,1) * U
4ØØ GOTO 44Ø
```

```
41Ø  FOR I = 5 TO 2 STEP -1
42Ø  LET Q(I) = Q(I - 1) - M(7 - I,2) * X + M(7 - I,1) * U: NEXT I
43Ø  LET Q(1) = -M(6,2) * X + M(6,1) * U
44Ø  IF ABS(X) > XM THEN LET XM = ABS(X)
45Ø  IF SF = Ø THEN PRINT TAB(1Ø);H * T; TAB (2Ø);X: GOTO 48Ø
46Ø  LET XA = FX * X + 132: IF XA > 24Ø OR XA < 1Ø GOTO 61Ø
47Ø  PSET 4 * K + 25,XA
48Ø  LET H = H + 1: NEXT K
49Ø  IF SF = Ø THEN LET FX = 75/XM
5ØØ  PRINT: INPUT "WEITERE WERTE (J/N) ?";E$: LET N$ = "N": CLS
51Ø  IF E$ = "N" GOTO 54Ø
52Ø  IF SF THEN GOSUB 74Ø
53Ø  GOTO 32Ø
54Ø  IF BP = 2 THEN INPUT "NEUES OMEGA (J/N) ?";N$: CLS
55Ø  IF N$ = "N" GOTO 57Ø
56Ø  GOTO 27Ø
57Ø  IF SF THEN PRINT "ZAHLEN?:                    Z"
58Ø  IF SF = Ø THEN PRINT "DIAGRAMM:                    D"
59Ø  PRINT "ANDERE TASTZEIT:          T"
6ØØ  PRINT "ANDERE EINGANGSGROESSEN   E"
61Ø  IF SF THEN PRINT "ANDERE SKALIERUNG:          F"
62Ø  PRINT "NEUEINGABE:               N"
63Ø  PRINT "SCHLUSS:                  S": PRINT
64Ø  INPUT B$: CLS
65Ø  IF B$ = "D" THEN LET SF = 1: LET RW = 72: GOTO 29Ø
66Ø  IF B$ = "Z" THEN LET SF = Ø: LET RW = 25: GOTO 3ØØ
67Ø  IF B$ = "E" THEN LET SF = Ø: LET RW = 25: GOTO 21Ø
68Ø  IF B$ = "T" THEN LET ST = 1: GOTO 14Ø
69Ø  IF B$ = "F" THEN PRINT "SKALIERUNG F = ";FX: GOTO 72Ø
7ØØ  IF B$ = "N" THEN RUN
71Ø  END
72Ø  INPUT "NEUER WERT";FX: CLS: GOTO 29Ø
73Ø  REM ZEICHNEN KOORDINATENSYSTEM
74Ø  LINE 25,132,315,132: LINE 25,29,25,234
75Ø  FOR X = 65 TO 3Ø5 STEP 4Ø: FOR Y = 133 TO 135: PSET X,Y,7: NEXT Y: NEXT X
76Ø  FOR Y = 32 TO 232 STEP 2Ø: FOR X = 23 TO 26: PSET X,Y: NEXT X: NEXT Y
77Ø  FOR I = 1 TO 7: PRINT AT(16, 5*I + 1);1Ø * I: NEXT I
78Ø  PRINT AT(2,1);1ØØ/FX: PRINT AT(29,1); -1ØØ/FX
79Ø  PRINT AT(16,35);"T": PRINT AT(3,32);"T = ";T: PRINT AT(15,2); "Ø"
8ØØ  RETURN
```

Im ersten Fall sind die Zählerkoeffizienten $b_0 \ldots b_2$ einzugeben (Programmnamen F(1,1)...F(3,1)) sowie die Nennerkoeffizienten $a_0 \ldots a_2$ (Programmnamen F(1,2)...F(3,2)). Im zweiten Fall sind die einzugebenden $b_0 \ldots b_5$ bzw. $a_0 \ldots a_5$ in den Feldelementen F(1,J)...F(6,J) untergebracht, wobei wie schon im Programm FREQ J = 1 den Zähler und J = 2 den Nenner der Übertragungsfunktion kennzeichnet. Da die Beziehungen zur Verarbeitung der Zähler- und Nennerkoeffizienten völlig gleich aufgebaut sind – (vgl. Gln. (99) und (102)) –, läßt sich zur Berechnung der Koeffizienten $g_0 \ldots g_2(g_5)$ und $h_0 \ldots h_2(h_5)$ mit den Programmnamen M(1,J)...M(3,J) (M(6,J)) eine einfache Laufanweisung über J einsetzen (Zeilen 15Ø bis 18Ø). Der Programmablaufplan Bild 25 wird für n = 2 in den Zeilen 36Ø bis 39Ø und für n = 5 in den Zeilen 36Ø, 41Ø bis 43Ø abgearbeitet.

Der Dialog mit dem Nutzer ist weitgehend an den des Programms ZUST angelehnt (vgl. Abschn. 3.2.3.). Wie bei diesem Programm wird beim ersten Durchlauf eine Zahlentabelle (Zeit/Werte von X) mit jeweils RW = 25 Werten ausgegeben (SF = Ø), wobei gleich der Skalierungsfaktor FX für den Ordinatenmaßstab in der Diagrammausgabe ermittelt wird (Zeile 49Ø). Die nach Ausdruck der ersten 25 Werte möglichen Dialogeingaben haben dieselbe Bedeutung wie beim Programm ZUST (s. auch Bild 23).

Anwendungsbeispiel

Eine Simulation der Übertragungsfunktionen (79) oder (80) des Beispiels bringt keine anderen Kurvenverläufe hervor, als sie aus dem Zustandsmodell erhalten wurden (s. Bild 24).
Statt dessen wird im Bild 26 die mit dem Programm KLEMM ermittelte Sprungantwort eines Übertragungsglieds mit der Übertragungsfunktion

$$G(p) = \frac{12 - 6p + p^2}{12 + 6p + p^2}$$

gezeigt; das ist die Padé-Approximation 2. Ordnung eines Laufzeitglieds.

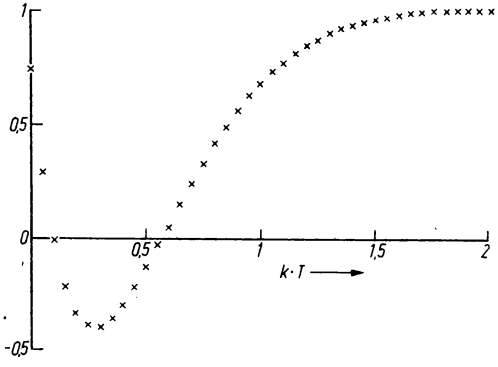


Bild 26. Sprungantwort eines Übertragungsglieds mit $G(p) = \frac{12 - 6p + p^2}{12 + 6p + p^2}$

4.3. Eigenwertberechnung

Auch für diskontinuierliche Systeme mit der Systemmatrix **P** existiert eine Eigenwertgleichung, die völlig der Gl. (72) entspricht:

$$|z\,\mathbf{E} - \mathbf{P}| = 0\,. \tag{105}$$

Genauso gibt es ein charakteristisches Polynom F(z), das sich durch Ausrechnen der Gl. (105) oder aus dem Nenner der z-Übertragungsfunktion G(z) – s. Gl. (90) – ergibt:

$$h_n + h_{n-1}z + \ \ldots\ + h_0 z^n = 0\,.$$

Die Verhältnisse entsprechen also vollständig den im Abschn. 2.3. behandelten. Die einzigen Unterschiede liegen in den Symbolen (z anstelle p, **P** anstelle **A**, h_i anstelle a_i sowie anderslaufende Indizierung der Koeffizienten des charakteristischen Polynoms).

Das im Abschn. 2.3. behandelte Programm EWEPOL läßt sich somit ohne Einschränkungen auch dazu verwenden, das charakteristische Polynom der Matrix **P** in einem diskontinuierlichen System zu ermitteln und seine Nullstellen zu suchen. Lediglich bei der Dialogführung muß der Nutzer auf die unterschiedliche Symbolik achten.
Auf diese Weise wurden die folgenden Eigenwerte für das diskontinuierliche System mit den Matrizen nach Tafel 10 erhalten:

Regelstrecke, $n = 3$
Abtastzeit $T = 1$: $z_1 = 0.7472$, $z_2 = 0.8943$, $z_3 = 0.923$
Abtastzeit $T = 2$: $z_1 = 0.5584$, $z_2 = 0.8$, $z_3 = 0.8521$

Regelkreis, $n = 4$
$T = 1$: $z_1 = 0.9$, $z_2 = 0.9129$, $z_{3,4} = 0.8283 \pm j\,0.2445$
$T = 2$: $z_1 = 0.8093$, $z_2 = 0.8452$, $z_{3,4} = 0.6262 \pm j\,0.405$.

Allerdings sind die Eigenwerte z_i diskontinuierlicher Systeme anders zu interpretieren als die p_i kontinuierlicher Systeme (vgl. Abschn. 2.3.1.).
Es gilt:

- Positiv reelle z_i bedeuten monoton verlaufende Anteile in den Folgen.
- Bedingung für Stabilität ist, daß die Beträge der z_i kleiner sind als 1

$$|z_i| < 1 .$$

Im Beispiel ergeben sich folgende Beträge der komplexen Eigenwerte für $n = 4$:

$$T = 1: \quad |z_{3,4}| = \sqrt{0.8283^2 + 0.2445^2} = 0.8636$$

$$T = 2: \quad |z_{3,4}| = \sqrt{0.6262^2 + 0.405^2} = 0.7458 .$$

Es handelt sich somit um stabile Folgen, wie ja auch schon aus Bild 24 zu erkennen war.

4.4. Regelstrecke mit Verzögerung und Laufzeit

4.4.1. Regelkreismodell

Aus theoretischen Prozeßanalysen lassen sich leicht Zustandsmodelle herleiten (s. Abschn. 1.1.). Kann man das Prozeßverhalten dagegen nur experimentell untersuchen, dann erfordert die Aufnahme von Sprungantworten den geringsten Aufwand, und zur Parametrisierung der dabei erhaltenen Kurven ist die Approximation durch ein dynamisches Verhalten mit Laufzeit T_L und Verzögerung 1. Ordnung T_1 am einfachsten (Bild 27). Insbesondere das Laufzeitverhalten läßt sich bei der digitalen Nachbildung besonders einfach berücksichtigen.
Bild 28 zeigt den nachzubildenden Regelkreis, in dem die Stellstrecke mit der Übertragungsfunktion G_u ein PT_LT_1-Verhalten aufweisen soll. Für die Störstrecke G_z hat die Annahme einer Laufzeit keine Bedeutung; deshalb reicht die Annäherung durch ein PT_1-Verhalten aus. Für die beiden Übertragungsfunktionen gilt daher

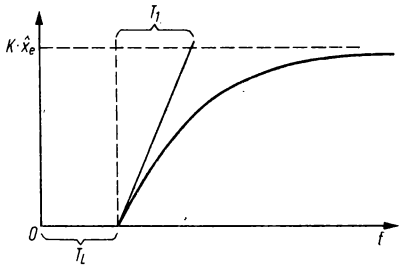


Bild 27. PT_LT_1-Sprungantwort

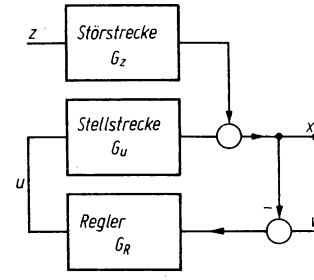


Bild 28. Regelkreis mit Störstrecke und Stellstrecke

Stellstrecke: $$G_u(p) = \frac{K_u e^{-pT_L}}{1 + pT_{1u}} \tag{106}$$

Störstrecke: $$G_z(p) = \frac{K_z}{1 + pT_{1z}} \tag{107}$$

PI-Regler: $$G_R(p) = K_P\left(1 + \frac{1}{pT_n}\right). \tag{108}$$

Das Blockdiagramm Bild 28 soll zur Grundlage eines diskontinuierlichen Klemmenmodells gemacht werden, mit dem die interessierenden Größen: Regelgröße x und Steuergröße (Stellgröße) u unter Einwirkung von veränderlichen Störgrößen z bzw. Führungsgrößen w berechnet werden sollen.
Durch Auswertung des Blockdiagramms erhält man sofort

– für das *Störverhalten* (W = 0)

$$X = \frac{G_z}{1 + G_u G_R} Z \tag{109}$$

$$U = -G_R X$$

– für das *Führungsverhalten* (Z = 0)

$$X = \frac{G_R G_u}{1 + G_u G_R} W \tag{110}$$

$$U = G_R(W - X).$$

Wenn die Regler im Abtastbetrieb arbeiten, was angenommen werden darf, dann gelten (109) und (110) sowohl für kontinuierliche Systeme, in denen G_u, G_z, G_R Übertragungsfunktionen als Funktionen von p darstellen (106) bis (108), als auch für diskontinuierliche Systeme, bei denen die z-Übertragungsfunktionen einzusetzen sind.
Mit T als der Abtastdauer erhält man für die z-Übertragungsfunktion eines Verzögerungsglieds 1. Ordnung, z. B. G_z nach Gl. (107) die Beziehung [8]

$$G_z(z) = \frac{K_z(1 - a_z)\, z^{-1}}{1 - a_z z^{-1}}. \tag{111}$$

Für den Laufzeitanteil mit der Laufzeit T_L tritt einfach eine Verzögerung um d Takte auf, wobei d angibt, wie oft die Abtastdauer T in der Laufzeit T_L enthalten ist:

$$d = \mathrm{INT}(T_L/T) .$$

Die Verzögerung um d Takte wird durch z^{-d} dargestellt; dadurch erhält die z-Übertragungsfunktion zu (106) die Form

$$G_u(z) = \frac{K_u(1 - a_u)z^{-(1+d)}}{1 - a_u z^{-1}} . \tag{112}$$

Dabei errechnen sich die Konstanten a_z und a_u gemäß

$$a_z = e^{-T/T_{1z}}, \quad a_u = e^{-T/T_{1u}} .$$

Für den PI-Regler wird p nach (96) in die Übertragungsfunktion (108) eingesetzt; dies entspricht einer Integration nach der Trapezregel. Man erhält

$$G_R(z) = K_P \frac{\bar{b} - bz^{-1}}{1 - z^{-1}} . \tag{113}$$

Dabei werden die Koeffizienten b und $\bar{b}$ wie folgt berechnet:

$$b = 1 - \frac{T}{2T_n}, \quad \bar{b} = 1 + \frac{T}{2T_n} .$$

Für $T_n \to \infty$ erhält man $b = \bar{b}$, und Gl. (113) entspricht einem reinen P-Regler.
Wenn man die Übertragungsfunktionen (111) bis (113) in (109) und (110) einsetzt, dann bekommt man für das Gesamtübertragungsverhalten:

Störverhalten

$$X = \frac{K_z(1 - a_z)\left[z^{-1} - (1 + a_u)z^{-2} + a_u z^{-3}\right]}{N_1(z)} Z \tag{114}$$

Dabei hat der Nenner $N_1(z)$ folgende Gestalt:

$$\begin{aligned} N_1(z) = {} & 1 - (1 + a_u + a_z)z^{-1} + (a_u + a_z + a_u a_z)z^{-2} - a_u a_z z^{-3} \\ & + K_P K_u(1 - a_u)bz^{-(d+1)} - K_P K_U(1 - a_u)(b + a_z\bar{b})z^{-(d+2)} \\ & + K_P K_u a_z b(1 - a_u)z^{-(d+3)} . \end{aligned}$$

Führungsverhalten

$$X = \frac{K_P K_u(1 - a_u)\left[\bar{b}z^{-(d+1)} - bz^{-(d+2)}\right]}{N_2(z)} W \tag{115}$$

Der Nenner $N_2(z)$ ist hier etwas einfacher:

$$\begin{aligned} N_2(z) = {} & 1 - (1 + a_u)z^{-1} + a_u z^{-2} + K_P K_u(1 - a_u)\bar{b}z^{-(d+1)} \\ & - K_P K_u(1 - a_u)bz^{-(d+2)} . \end{aligned}$$

Die Gln. (114) und (115) sind die gesuchten Klemmenmodelle, die die Zusammenhänge zwischen der Ausgangsgröße X (Regelgröße) und den jeweiligen Eingangs-

größen (Z bzw. W) herbeiführen. Diese Gleichungen können nach dem im Abschn. 4.2.1. beschriebenen Verfahren direkt in Rechenprogramme umgesetzt werden. Da die beiden Nenner $N_1(z)$ und $N_2(z)$ unterschiedliche Ordnungen aufweisen (N_1: $d+3$, N_2: $d+2$), sind auch zwei unterschiedliche Programme angebracht.

4.4.2. Eingangssignale

Als Eingangssignale für Störgröße z und Führungsgröße w sollen, wie schon bei den Programmen ZUST (s. Abschn. 3.2.) und KLEMM (s. Abschn. 4.2.) verwendet, vorgesehen werden: Sprungfunktionen, Sinusfunktionen und Zufallsfunktionen.
Für Sprungfunktionen ergeben sich keine Besonderheiten.
Für Sinusfunktionen gibt es, wie schon bei den beiden anderen Programmen, Zusammenhänge zwischen der Kreisfrequenz ω und der höchstzulässigen Abtastdauer T, die im vorliegenden Programm TLKRS mit $\omega T \leqq 1/3$ vorgegeben wurden. Die Forderungen sind also strenger als bei den beiden anderen Programmen, bei denen $\omega T \leqq \pi/4$ verlangt wird. Folge dieser strengeren Forderung ist u. a., daß eine Periode der Sinusschwingung durch mehr Punkte auf dem Bildschirm dargestellt wird, wodurch sich gut auswertbare Bilder ergeben.
Für Zufallssignale als Eingangsgrößen muß beachtet werden, daß auch das Führungsverhalten der Regelkreise untersucht werden soll. Weiße Rauschsignale, wie sie durch Aufruf der Funktion RND nach Gl. (87) erhalten werden, sind aber als Führungsgrößen für Regelkreise sehr wenig sinnvoll; denn von keinem trägheitsbehafteten System kann erwartet werden, daß es in seiner Bewegung einem solchen Signal folgt. (Für das Störsignal existiert eine solche Einschränkung nicht; außerdem übt das Trägheitsverhalten der Störstrecke G_z eine gewisse Filterung auf das Störrauschen aus.)
Das Führungssignal muß aber durch ein gesondertes Filter aus dem weißen Rauschen erzeugt werden. Ein solches Signalfilter mit veränderbaren Parametern zu programmieren wäre im Prinzip leicht möglich. Im Programm TLKRS wurde aber darauf verzichtet, die Filterparameter veränderbar zu gestalten. Anstatt dessen wurde ein festes Rauschfilter vorgesehen, das durch die z-Übertragungsfunktion

$$G_f(z) = \frac{(1-k)\,z^{-2}}{(1-kz^{-1})^2}$$

beschrieben wird. Das entspricht einem Verzögerungsglied 2. Ordnung. Für k wurde der feste Wert $e^{-0.3}$ gewählt. Dadurch erhalten die beiden gleichen Zeitkonstanten des Verzögerungsgliedes die Werte $T_1 = T_2 = 3.3\ T$.

4.4.3. Rechenprogramm TLKRS

Das Rechenprogramm TLKRS verlangt die Eingabe der Parameter von Störstrecke (KZ, TZ), Stellstrecke (KU, TL, TU) und Regler (KP, TN) in den Zeilen 3Ø bis 5Ø. (Die Namen der Programmvariablen sind denen der Veränderlichen in den Abschnitten 4.4.1. und 4.4.2. angeglichen.)
Die Abtastdauer T wird zunächst zu TL/4 festgelegt (Zeile 6Ø); dieser Wert kann in Zeile 15Ø, 16Ø ggf. noch verändert werden. Dabei ist zu beachten, daß sich die Ord-

nung des Gesamtsystems für Störung zu n = d + 3 und für Führung zu n = d + 2 ergibt (Zeilen 18Ø, 19Ø). Im Programm ist als höchste Ordnung n = 15 vorgesehen (Zeile 2Ø); das entspricht T/TL ≧ 1/12. Normalerweise ist der vorgegebene Wert T/TL = 1/4 ausreichend. Lediglich, wenn die Reaktion des Regelkreises auf Sinussignale höherer Frequenz untersucht werden soll, ist wegen der im Abschn. 4.4.2. beschriebenen Bedingung ωT ≦ 1/3 ggf. eine Verminderung von T und damit Erhöhung der Ordnung erforderlich (Zeile 14Ø).

Programm TLKRS

```
1Ø  REM REGELKREIS MIT LAUFZEIT
2Ø  DIM Q(15)
3Ø  INPUT "KZ ";KZ: INPUT "T1Z ";TZ: INPUT "KU ";KU: INPUT "TL ";TL
4Ø  INPUT "T1U ";TU: LET SF = Ø: LET RW = 25
5Ø  INPUT "KP ";KP: INPUT "TN ";TN
6Ø  LET T = TL/4
7Ø  INPUT "MODUS Z/W ";M$
8Ø  PRINT "SIGNALFORM SPRUNG:   1"
9Ø  PRINT "SINUS:               2"
1ØØ PRINT "ZUFALL:              3"
11Ø INPUT ST: CLS
12Ø IF ST < >2 GOTO 15Ø
13Ø INPUT "OMEGA ";OM
14Ø IF T > 1/(3 * OM) THEN LET T = 1/(3 * OM)
15Ø PRINT "BESTAETIGEN (B) ODER NEUER WERT FUER T = ";T: INPUT T$
16Ø IF T$ < > "B" THEN LET T = VAL(T$)
17Ø CLS: LET D = INT (TL/T) IF SF THEN GOSUB 97Ø
18Ø IF M$ = "Z" THEN LET N = D + 3: GOTO 2ØØ
19Ø IF M$ = "W" THEN LET N = D + 2
2ØØ LET AU = EXP(−T/TU): LET AZ = EXP(−T/TZ)
21Ø LET B1 = KZ * (1−AZ): LET A1 = −(AU + AZ): LET A2 = AU * AZ
22Ø LET KK = EXP(−.3): LET B4 = KP * KU * (1 − AU)
23Ø LET B = 1 − T/(2 * TN): LET BQ = 1 + T/(2 * TN)
24Ø LET ZW = Ø: LET XE = 1
25Ø IF M$ = "W" GOTO 51Ø
26Ø REM STOERUNG
27Ø FOR I = 1 TO N: LET Q(I) = Ø: NEXT I
28Ø LET Q1 = Ø :LET H = Ø: LET XM = Ø: LET UM = Ø
29Ø IF ST = 3 THEN LET XE = Ø: LET ZW = Ø
3ØØ FOR K = Ø TO RW
31Ø IF ST = 3 THEN LET XE = KK * XE + ZW: LET ZW = KK * ZW + (1−KK)
    * (RND(1)−.5)
32Ø IF ST = 2 THEN LET XE = SIN(H * OM * T)
33Ø IF SF THEN PSET 14Ø + 2 * K, 66 + 25 * XE
34Ø LET X = Q(1): IF ABS(X) > XM THEN LET XM = ABS(X)
35Ø LET Q(1) = Q(2) + B1 * XE − (A1 − 1) * X
36Ø LET Q(2) = Q(3) − (1 + AU) * B1 * XE − (A2 − A1) * X
37Ø LET Q(3) = Q(4) + AU * B1 * XE + A2 * X
38Ø FOR G = 4 TO N−3: LET Q(G) = Q(G + 1): NEXT G
39Ø LET Q(N − 2) = Q(N − 1) − B4 * X * BQ
4ØØ LET Q(N − 1) = Q(N) + (B + AZ * BQ) * X * B4: LET Q(N) = −AZ * B * B4 * X
41Ø LET U = Q1 − KP * X * BQ: IF ABS(U) > UM THEN LET UM = ABS(U)
```

```
42Ø  LET Q1 = KP * B * X + U
43Ø  IF SF = Ø THEN PRINT H * T;TAB(13);U;TAB(26);X: GOTO 47Ø
44Ø  LET UA = 46 + FU * U: LET XA = 1Ø6 + FX * X
45Ø  IF UA > 9Ø OR UA < = 4 OR XA > 2ØØ OR XA < = 8 THEN GOTO 81Ø
46Ø  PSET 8 + 2 * K,XA: PSET 14Ø + 2 * K,UA
47Ø  LET H = H + 1: NEXT K
48Ø  IF SF = Ø THEN LET FX = 6Ø/XM: LET FU = 3Ø/UM
49Ø  GOTO 71Ø
5ØØ  REM FUEHRUNG
51Ø  FOR I = 1 TO N: LET Q(I) = Ø: NEXT I
52Ø  LET XM = Ø: LET UM = Ø: LET Q1 = Ø: LET H = Ø
54Ø  FOR K = Ø TO RW
55Ø  IF ST = 3 THEN LET XE = KK * XE + ZW: LET ZW = KK * ZW + (1 - KK)
     * (RND(1) - .5)
56Ø  IF ST = 2 THEN LET XE = SIN(H * OM * T)
57Ø  IF SF THEN PSET 14Ø + 2 * K, 166 + 25 * XE
58Ø  LET X = Q(1): IF ABS(X) > XM THEN LET XM = ABS(X)
59Ø  LET Q(1) = Q(2) + (1 + AU) * X: LET Q(2) = Q(3) - AU * X
6ØØ  FOR G = 3 TO N - 2: LET Q(G) = Q(G + 1): NEXT G
61Ø  LET Q(N - 1) = Q(N) + B4 * BQ * (XE - X)
62Ø  LET Q(N) = B * B4 * (X - XE)
63Ø  LET U = Q1 - KP * BQ * (X - XE): IF ABS(U) > UM THEN LET UM = ABS(U)
64Ø  LET Q1 = KP * B * (X - XE) + U
65Ø  IF SF = Ø THEN PRINT H * T;TAB(13);U;TAB(26);X: GOTO 69Ø
66Ø  LET UA = 46 + FU * U: LET XA = 1Ø6 + FX * X
67Ø  IF UA > 9Ø OR UA < = 4 OR XA > 2ØØ OR XA < = 8 THEN GOTO 81Ø
68Ø  PSET 14Ø + 2 * K,UA: PSET 8 + 2 * K,XA
69Ø  LET H = H + 1: NEXT K
7ØØ  IF SF = Ø THEN LET FX = 6Ø/XM: LET FU = 3Ø/UM
71Ø  INPUT "WEITERE WERTE J/N ? ";W$: CLS
72Ø  IF W$ = "N" GOTO 76Ø
73Ø  IF SF THEN GOSUB 97Ø
74Ø  IF M$ = "Z" GOTO 3ØØ
75Ø  GOTO 54Ø
76Ø  IF ST = 2 THEN PRINT "NEUES OMEGA:                O"
77Ø  IF SF = Ø THEN PRINT "DIAGRAMM:                   D"
78Ø  IF SF THEN PRINT "ZAHLEN:                      Z"
79Ø  PRINT "NEUE REGLERPARAMETER:    P"
8ØØ  PRINT "ANDERER MODUS:           M"
81Ø  IF SF THEN PRINT "ANDERE SKALIERUNG:       F"
82Ø  PRINT "NEUEINGABE:              N"
83Ø  PRINT "ENDE:                    E"
84Ø  INPUT E$: CLS
85Ø  IF E$ = "M" THEN LET SF = Ø: LET RW = 25: GOTO 7Ø
86Ø  IF E$ = "N" THEN RUN
87Ø  IF E$ = "D" THEN LET SF = 1: LET RW = 6Ø: GOSUB 97Ø: GOTO 94Ø
88Ø  IF E$ = "Z" THEN LET SF = Ø: LET RW = 25: GOTO 94Ø
89Ø  IF E$ = "F" THEN PRINT "FX = ";FX;"FU = ";FU: INPUT FX,FU: CLS: GOSUB
     97Ø: GOTO 94Ø
9ØØ  IF E$ = "O" THEN PRINT "OMEGA = ";OM: GOTO 13Ø
91Ø  IF E$ = "P" THEN PRINT "KP = ";KP: PRINT "TN = ";TN: GOTO 93Ø
92Ø  END
93Ø  INPUT "KP ";KP: INPUT "TN ";TN: LET SF = Ø: LET RW = 25: CLS: GOTO 22Ø
```

```
94Ø  IF M$ = "Z" GOTO 27Ø
95Ø  GOTO 51Ø
96Ø  REM KOORDINATENSYSTEME
97Ø  LINE 8,1Ø6,128,1Ø6: LINE 14Ø,46,26Ø,46: LINE 14Ø,166,26Ø,166
98Ø  LINE 8,4Ø,8,172: LINE 14Ø,1Ø,14Ø,82: LINE 14Ø,13Ø,14Ø,2Ø2
99Ø  PRINT AT(8,2);"X": PRINT AT(5,18);M$: PRINT AT(2Ø,18);"U"
1ØØØ PRINT AT(18,23);"T = ";T: RETURN
```

Bei der Umsetzung der Übertragungsfunktionen (114) für Störverhalten und (115) für Führungsverhalten in das Rechenprogramm (Grundlage ist der PAP Bild 25; s. Abschn. 4.2.1.) müssen die Koeffizienten des Zählerpolynoms g_i und Nennerpolynoms h_i dieser Übertragungsfunktionen gebildet werden. Die Bildungsvorschriften sind aus (114) und (115) abzulesen. Im Programm wurde lediglich versucht, einige in Zählern und Nennern beider Übertragungsfunktionen öfter auftretenden Koeffizientenkombinationen zu neuen Konstanten zusammenzufassen. So wurde gesetzt

b_1 für $K_z(1 - a_z)$
b_4 für $K_P K_u(1 - a_u)$
a_1 für $-(a_u + a_z)$
a_2 für $a_u a_z$

(Zeilen 22Ø, 23Ø). KK ist der Programmname für die Rauschfilterkonstante k; BQ steht für $\bar{b}$.
Mit diesen zusammengefaßten Koeffizienten können die Gleichungen nach PAP Bild 25 relativ kurz und übersichtlich formuliert werden. In den Zeilen 34Ø bis 4ØØ wird der Verlauf von x für den Störungsfall, in 58Ø bis 62Ø für den Führungsfall berechnet. Im Gegensatz zum PAP Bild 25 und den darauf beruhenden Zeilen 36Ø bis 43Ø im Programm KLEMM wurden die Speichergrößen q_i im Programm TLKRS in umgekehrter Reihenfolge indiziert; also entspricht q_n im PAP der Programmgröße Q(1) und umgekehrt.
In den Zeilen 31Ø bzw. 55Ø ist jeweils der Algorithmus zur Rauschfilterung, also die Nachbildung der Filterfunktion G_f, programmiert. Dazu wird die Zwischengröße ZW benötigt.
Da das Programm auch die Verläufe der Steuergrößen u ausrechnen soll, werden diese aus den berechneten Verläufen von x mit der Übertragungsfunktion $G_R(z)$ des Reglers berechnet. Bild 29 zeigt den einfachen PAP zur Nachbildung von (113), der sich in den Programmzeilen 41Ø, 42Ø bzw. 63Ø, 64Ø wiederfindet.
Die Gestaltung des Gesamtprogrammablaufs und die Dialogführung folgt den bereits in den Programmen ZUST und KLEMM benutzten Prinzipien. Zunächst werden 25 Zahlenwerte berechnet und als Tabelle mit den Spalten Zeit, Wert der Steuergröße, Wert der Regelgröße ausgegeben (Zeilen 43Ø bzw. 65Ø).

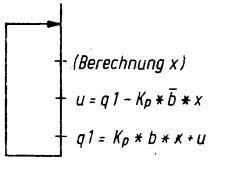


Bild 29. Programmablaufplan zur Berechnung der Ausgangsgröße Steuergröße u aus der Eingangsgröße Regelgröße x des PI-Reglers

Dabei werden gleich die Skalierungsfaktoren FX und FU für eine spätere Diagrammausgabe errechnet (Zeile 48Ø bzw. 7ØØ). Nach eventueller Fortführung der Tabellenausgabe mit weiteren 25 Werten (Zeilen 71Ø bis 75Ø) können Diagramme ausgegeben oder eine der anderen, auch aus Bild 23 erkennbaren Dialogvarianten gewählt werden (Zeilen 76Ø bis 92Ø). Eine weitere Möglichkeit besteht darin, bei konstanten Regelstreckenparametern die Einstellwerte des Reglers zu verändern. Hierzu dient die Dialogeingabe "P" in Zeile 79Ø. Die alten Reglereinstellwerte werden angezeigt, und die neuen können eingegeben werden (Zeilen 91Ø, 93Ø).
Wird Diagrammausgabe gewählt (SF = 1), dann wird in der linken Bildschirmhälfte der Werteverlauf der Regelgröße x dargestellt. In der rechten Hälfte des Bildschirms werden in kleinerem Maßstab die Verläufe von Störgröße z bzw. Führungsgröße w und darunter der Steuergröße u (Stellgröße) gezeigt. Dazu dienen die Anweisungen in den Zeilen 33Ø, 44Ø bis 46Ø für Störung und 57Ø, 66Ø bis 68Ø für Führung. Droht eine der Kurven x oder u die Bildschirmgrenzen zu überschreiten, dann wird automatisch zu Zeile 81Ø gesprungen, und der Nutzer kann die Werte der Abbildungsmaßstäbe FX oder FU ändern, deren Werte ihm dazu mit Zeile 89Ø angezeigt werden.
Bild 30 zeigt zwei Regelvorgänge, die mit dem Programm TLKRS berechnet worden sind. Die Parameter der Regelstrecke waren

$$K_z = 50,\ T_{1z} = 12,\ K_u = 1,\ T_L = 4,\ T_{1u} = 15.$$

Für die Tastperiode T wurde der vom Rechner vorgeschlagene Wert T = 1 verwendet.
Die Reglereinstellung waren

Kurve 1 (Kreise): $K_P = 2.62$, $T_n = 12.1$
Kurve 2 (Kreuze): $K_P = 3.9$, $T_n = 15.52$.

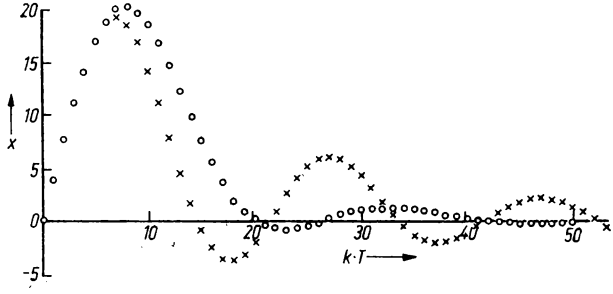


Bild 30. Regelvorgänge für T_L-T_1-Strecke
ooo Reglereinstellung nach ITAE-Kriterium; xxx Reglereinstellung nach quadratischer Regelfläche

Diese Einstellwerte entsprechen optimalen Regelvorgängen, die für verschiedene Gütekriterien ermittelt wurden (Kurve 1: ITAE-Kriterium, Kurve 2: quadratische Regelfläche).

5. Parameteroptimierung

5.1. Optimierungsaufgaben

Optimierungsaufgaben kommen in der Technik sehr oft vor. Als automatisierungstechnisches Beispiel soll die Ermittlung optimaler, d. h. günstigster Einstellwerte der Reglerparameter (K_P, T_n, T_v) in einem Regelkreis behandelt werden. Dazu muß gesagt werden, was unter „günstig" verstanden werden soll. Es muß also ein Gütekriterium formuliert werden, dessen Wert berechnet oder gemessen werden kann. Günstige Reglerparameter sind dann solche, die den Wert dieses Gütekriteriums möglichst groß oder möglichst klein werden lassen.
Gütekriterien zur Beurteilung von Regelvorgängen sind seit langem bekannt. Wenn man von der Antwort x(t) des geschlossenen Regelkreises auf eine sprungförmige Veränderung der Störgröße oder auch Führungsgröße ausgeht, dann wird mit dem Gütekriterium der Zeitverlauf dieser Sprungantwort bewertet.
Gebräuchlich sind u. a. folgende Gütemaße:

Quadratische Regelfläche

$$F = \int_0^{t_e} x_w^2(t)\,dt \qquad \text{bzw.} \quad F = \sum_{k=0}^{m} x_{wk}^2 \tag{116}$$

ITAE-Kriterium

$$F = \int_0^{t_e} |x_w|\,t\,dt \qquad \text{bzw.} \quad F = \sum_{k=0}^{m} k\,|x_{wk}| \tag{117}$$

Die Endzeit t_e bzw. m, die Anzahl der zu erfassenden zeitdiskreten Werte, ist so zu wählen, daß die Regelvorgänge praktisch abgeklungen sind.
Da der Zeitverlauf $x_w(t)$ bzw. die Werte der Folge x_{wk} von der Reglereinstellung abhängen, sind auch die Werte der Gütemaße (116) bzw. (117) von den Einstellparametern abhängig.
Die Suche nach optimalen Reglereinstellwerten ist nur ein Beispiel für Optimierungsaufgaben. Deshalb soll anstatt von Reglerparametern allgemein von „Entscheidungsvariablen" gesprochen werden [13]. Die Entscheidungsvariablen sind mithin so zu wählen, daß die Werte der Gütemaße möglichst klein werden:

$$K_P,\ T_n,\ T_v \text{ so, daß } F \rightarrow \min !$$

Zur Lösung solcher Probleme gibt es viele Verfahren [13].
Ist der Zusammenhang zwischen den Entscheidungsvariablen und dem Gütemaß kompliziert, vielleicht nur numerisch berechenbar, werden Suchverfahren verwendet, bei denen der Rechner die Werte der Entscheidungsvariablen so lange verändert, bis das gewünschte Minimum (hoffentlich) gefunden worden ist.
Auch bei den Suchverfahren gibt es sehr viele Varianten [14]. Das hier vorgestellte Verfahren nach Rosenbrock ist schon sehr lange bekannt. Es ist leistungsfähig und verlangt nur einen geringen Programmieraufwand.

5.2. Suchverfahren nach Rosenbrock

5.2.1. Grundgedanke des Verfahrens

Die Entscheidungsvariablen werden mit x_i; $i = 1 \ldots n$ bezeichnet; der kürzeren Darstellung halber auch zu einem Spaltenvektor $\mathbf{x}$ zusammengefaßt.
Das Verfahren beginnt mit einem vom Nutzer vorzugebenden Startwert $\mathbf{x}_0$, mit dem der Wert F_0 des Gütekriteriums berechnet wird. Dann wird das Verfahren in großen Zyklen durchgeführt. Der Startzyklus läuft wie folgt ab:
Die erste Entscheidungsvariable x_1 wird um die Schrittweite h_1 vergrößert und damit der Wert F_1 des Gütekriteriums berechnet. Ist $F_1 \leqq F_0$, dann war der Schritt erfolgreich. Der Wert x_1 wird beibehalten, die Schrittweite h_1 für spätere Verwendung verdreifacht. War der Schritt nicht erfolgreich, wird er zurückgenommen, der Wert h_1 für spätere Verwendung mit -0.5 multipliziert. Dann erfolgt der zweite Schritt mit der Entscheidungsvariablen x_2, der Schrittweite h_2, danach der dritte usw. bis zur Veränderung von x_n um h_n. Sodann wird wieder mit x_1 und der veränderten Schrittweite h_1 begonnen.
Der Startzyklus wird beendet, sobald jede der Entscheidungsvariablen einmal erfolgreich und danach erfolglos verändert worden ist. Dann wird zum zweiten Zyklus übergegangen.
Während beim Startzyklus im Schritt i nur die einzelne Entscheidungsvariable x_i um die Schrittweite h_i verändert wird, die anderen dagegen unverändert bleiben, werden in den Schritten i der folgenden Zyklen mit der Schrittweite h_i gleichzeitig alle Entscheidungsvariablen verändert, und zwar

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad \text{um } h_i * \begin{pmatrix} v_{1,i} \\ v_{2,i} \\ \vdots \\ v_{n,i} \end{pmatrix}. \tag{118}$$

In Vektorform läßt sich dafür kurz schreiben

$$\mathbf{x} := \mathbf{x} + h_i \mathbf{v}_i .$$

Dabei ist $\mathbf{v}_i$ der aus (118) ablesbare Spaltenvektor, der zukünftig „Richtungsvektor" genannt wird.
Auch für den Startzyklus läßt sich die Schreibweise (118) verwenden, wobei die Komponenten des Richtungsvektors $v_{j,i}$ für $j = i$ den Wert 1, für alle anderen j den Wert 0 haben.
Schritt 1 im zweiten Zyklus beginnt demnach mit dem Richtungsvektor $\mathbf{v}_1$ und der Schrittweite h_1, Berechnung von F, Vergleich mit dem vorherigen Wert, Feststellung, ob der Schritt erfolgreich war oder nicht, und entsprechender Veränderung von h_1 ($* 3$ oder $* -0.5$). Dann folgt Schritt 2 mit $\mathbf{v}_2$, h_2 usw. Wurde auch in diesem Zyklus jeder Schritt i einmal erfolgreich und danach erfolglos ausgeführt, dann wird ein neuer Satz von Richtungsvektoren $\mathbf{v}_i$ berechnet, und der Zyklus beginnt von neuem. Die neuen Richtungsvektoren werden aus den Ergebnissen der im jeweiligen Zyklus erfolgreich absolvierten Schritte berechnet. Die theoretischen Hintergründe sollten der Literatur [13; 14] entnommen werden.

5.2.2. Programmablaufplan

Bild 31 zeigt den Programmablaufplan für das Rosenbrock-Verfahren. Die einzelnen Größen, soweit sie noch nicht beschrieben wurden, haben folgende Bedeutung:
Zeile '1': Die Schrittweiten $h_1 \ldots h_n$ werden zu einem Spaltenvektor **h** zusammengefügt; die Spaltenvektoren $v_1 \ldots v_n$ zu einer Richtungsmatrix V. Zu Beginn des Startzyklus wird V gleich der Einheitsmatrix E gesetzt.
Zeile '3': Der Summiervektor **s** mit den Komponenten s_i; $i = 1 \ldots n$ summiert die Werte h_i der erfolgreichen Schritte (Zeile '13'). Er wird zur Neuberechnung der Richtungsvektoren gebraucht. Der Zählvektor **a**, ebenfalls mit n Komponenten, dient dazu, festzustellen, ob für jedes i nach einem erfolgreichen ein erfolgloser Schritt vorgenommen wurde. Dazu werden die Komponenten a_i entsprechend dem Ausgang jeden Schrittes verändert (Zeilen '12' und '16'). Daß und wie damit die beabsichtigte Entscheidung erreicht wird, erkennt man am besten, wenn man einige Durchläufe durch den PAP in Gedanken durchspielt.

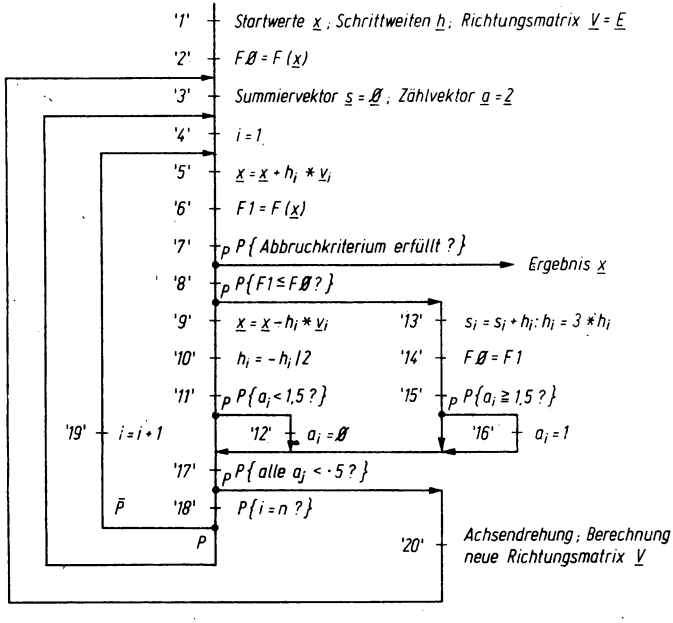


Bild 31. Programmablaufplan für Minimumsuche nach Rosenbrock

Die übrigen Zeilen im PAP verwirklichen den Algorithmus, wie er im Abschn. 5.2.1. beschrieben wurde. Der Startzyklus mit V = E wird mit Rücksprüngen auf Zeile '4' so lange durchlaufen, bis in Zeile '17' festgestellt wird, daß für alle i nach einem erfolgreichen ein erfolgloser Schritt durchgeführt wurde. Dann wird in Zeile '20' ein neuer Satz von Richtungsvektoren berechnet, mit dem von vorne begonnen wird.
Im Verlauf der Rechnung muß immer wieder überprüft werden, ob das gewünschte

Minimum des Gütekriteriums F gefunden wurde und die Suche abgebrochen werden kann (Zeile '7'). Dazu gibt es verschiedene Möglichkeiten:

- Abbruch, wenn der Unterschied zwischen zwei aufeinanderfolgenden Werten des Gütekriteriums genügend klein ist:
  $|F_1 - F_0| < \text{eps}$ (119)
- Abbruch, wenn sich die Werte der Entscheidungsvariablen nur noch unwesentlich ändern. Wird mit d_i der Unterschied zwischen zwei aufeinanderfolgenden Werten von x_i bezeichnet, dann wird abgebrochen, wenn für alle i gilt
  $|d_i| < \text{eps}$. (120)

5.2.3. Berechnung der Richtungsvektoren

Die neuen Richtungsvektoren $\mathbf{v}_i$; $i = 1 \ldots n$ werden nach Beendigung jeden Zyklus aus den bisherigen $\mathbf{v}_i$ und den Summiergrößen s_i gebildet. Zunächst berechnet man die Vektoren $\mathbf{w}_j$ nach der Vorschrift

$$\begin{aligned} \mathbf{w}_1 &= s_1\mathbf{v}_1 + s_2\mathbf{v}_2 + \ldots + s_n\mathbf{v}_n \\ \mathbf{w}_2 &= s_2\mathbf{v}_2 + \ldots + s_n\mathbf{v}_n \\ &\vdots \\ \mathbf{w}_n &= s_n\mathbf{v}_n . \end{aligned} \tag{121}$$

Der neue Richtungsvektor $\mathbf{v}_1$ ergibt sich daraus sofort zu

$$\mathbf{v}_1 = \frac{\mathbf{w}_1}{|\mathbf{w}_1|} .$$

Der Betrag $|\mathbf{w}_1|$ wird folgendermaßen berechnet:

$$|\mathbf{w}_1| = \sqrt{w_{11}^2 + w_{21}^2 + \ldots + w_{n1}^2} . \tag{122}$$

Zur Berechnung der übrigen Vektoren $\mathbf{v}_2 \ldots \mathbf{v}_n$ wird nach dem Programmablaufplan Bild 32 verfahren. Zunächst werden in Zeile '4' die Skalarprodukte $p_{k,l}$ nach folgendem Schema gebildet:

$$p_{k,l} = w_{1,k}v_{1,l} + w_{2,k}v_{2,l} + \ldots + w_{n,k}v_{n,l} .$$

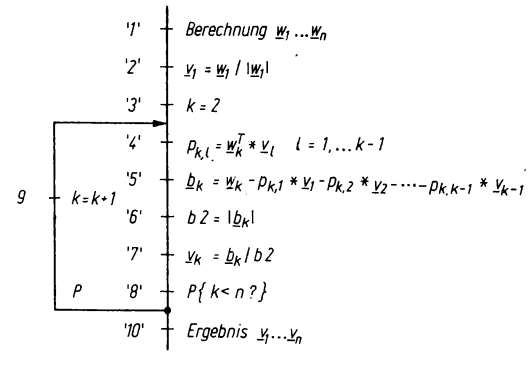


Bild 32. Programmablaufplan zur Berechnung der Richtungsvektoren des Rosenbrock-Verfahrens

Beim ersten Durchlauf (k = 2) ist nur ein einziger Koeffizient $p_{2,1}$ zu berechnen, beim zweiten (k = 3) die Koeffizienten $p_{3,1}$ und $p_{3,2}$ usw.
Entsprechend erhöht sich auch die Anzahl der Summanden in Zeile '5' bei jedem Durchlauf, und in Zeile '4' und '5' werden immer die neuen Vektoren $\mathbf{v}_i$ verwendet, die in den vergangenen Durchläufen errechnet worden sind.
Der Betrag | b2 | in Zeile '6' errechnet sich in Analogie zu (122) aus den Komponenten des Vektors $\mathbf{b}_k$.

5.2.4. Rechenprogramm ROSE

Das Optimierungsprogramm ROSE ist anhand der Programmablaufpläne der Bilder 31 und 32 leicht zu durchschauen. Die vom Nutzer nach Eingabe der Ordnung (Anzahl der Entscheidungsvariablen) verlangten Startwerte (Zeilen 5Ø, 6Ø) sollten möglichst in der Nähe der vermuteten Optimalwerte liegen. Die Schrittweiten H(I) können etwa mit 0.01...0.1 * X(I) gewählt werden (Zeilen 8Ø bis 1ØØØ).
Als Abbruchkriterium wurde von der Bedingung (120) ausgegangen; allerdings wird der Wert der d_i noch durch $(1 + | x_i |)$ dividiert (Zeile 16Ø). Der Selektor SR erhält dann den Wert 1 und leitet zum Abbruch der Rechnung und Ergebnisausdruck in Zeile 54Ø über, wenn die Bedingung Zeile 16Ø für jede Entscheidungsvariable erfüllt ist.

Programm ROSE

```
1Ø  REM MINIMUMSUCHE ROSENBROCK
2Ø  REM FUNKTION STEHT•ALS UP AB ZEILE 1ØØØ
3Ø  INPUT "DIMENSION N ";N: DEFFN Q(X) = X * X
4Ø  DIM A(N),B(N),E(N),H(N),P(N,N),S(N),W(N,N),V(N,N),X(N)
5Ø  PRINT "STARTWERTE X(1)...X(";N;")": PRINT
6Ø  FOR P = 1 TO N: INPUT X(P): NEXT P
7Ø  GOSUB 1ØØØ: LET FØ = F: PRINT
8Ø  PRINT "SCHRITTWEITEN H(1)...H(";N;")": PRINT
9Ø  FOR I = 1 TO N: FOR J = 1 TO N: LET V(I,J) = Ø: NEXT J: LET V(I,I) = 1: NEXT I
1ØØ FOR P = 1 TO N: INPUT H(P): NEXT P
11Ø FOR P = 1 TO N: LET S(P) = Ø: NEXT P
12Ø FOR P = 1 TO N: LET A(P) = 2:NEXT P
13Ø LET I = 1: LET SR = 1
14Ø FOR J = 1 TO N
15Ø LET D = H(I) * V(J,I)
16Ø IF ABS(D/(ABS(X(J)) + 1)) > 1E-4 THEN LET SR = Ø
17Ø LET X(J) = X(J) + D: NEXT J
18Ø IF SR GOTO 54Ø
19Ø GOSUB 1ØØØ
2ØØ IF F > FØ GOTO 22Ø
21Ø LET S(I) = S(I) + H(I): LET FØ = F: LET H(I) = 3 * H(I): GOTO 27Ø
22Ø FOR J = 1 TO N
23Ø LET X(J) = X(J) - H(I) * V(J,I): NEXT J
24Ø LET H(I) = -H(I)/2
25Ø IF A(I) < 1.5 THEN LET A(I) = Ø
26Ø GOTO 28Ø
27Ø IF A(I) > = 1.5 THEN LET A(I) = 1
```

```
28Ø  LET SL = 1: FOR K = 1 TO N
29Ø  IF A(K) > = .5 THEN LET SL = Ø
3ØØ  NEXT K
31Ø  IF SL = 1 GOTO 34Ø
32Ø  IF I = N GOTO 13Ø
33Ø  LET I = I + 1: GOTO 14Ø
34Ø  FOR P = 1 TO N: FOR R = 1 TO N: LET W(P,R) = Ø: NEXT R: NEXT P
35Ø  FOR J = 1 TO N: FOR K = J TO N: FOR I = 1 TO N
36Ø  LET W(I,J) = W(I,J) + S(K) * V(I,K)
37Ø  NEXT I: NEXT K: NEXT J
38Ø  LET WB = Ø: FOR P = 1 TO N
39Ø  LET WB = WB + FN Q(W(P,1)): NEXT P
4ØØ  LET WB = SQR(WB)
41Ø  FOR P = 1 TO N: LET V(P,1) = W(P,1)/WB: NEXT P
42Ø  FOR P = 1 TO N: FOR R = 1 TO N: LET P(P,R) = Ø
43Ø  NEXT R: NEXT P
44Ø  FOR K = 2 TO N: FOR L = 1 TO K − 1: FOR I = 1 TO N
45Ø  LET P(K,L) = P(K,L) + W(I,K) * V(I,L): NEXT I
46Ø  FOR I = 1 TO N
47Ø  LET B(I) = B(I) − P(K,L) * V(I,L): NEXT I: NEXT L
48Ø  FOR I = 1 TO N: LET B(I) = B(I) + W(I,K): NEXT I
49Ø  LET B2 = Ø: FOR I = 1 TO N
5ØØ  LET B2 = B2 + FN Q(B(I)): NEXT I
51Ø  FOR I = 1 TO N
52Ø  LET V(I,K) = B(I)/SQR(B2): LET B(I) = Ø: NEXT I: NEXT K
53Ø  GOTO 11Ø
54Ø  CLS: PRINT "MINIMUMSKOORDINATEN X(1)...X(";N;")"
55Ø  PRINT: FOR P = 1 TO N: PRINT X(P): NEXT P: PRINT
56Ø  PRINT "FUNKTIONSMINIMUM ";FØ
57Ø  END
8ØØ  REM ANWENDUNGSBEISPIEL : AUSSCHNITT AUS PROGRAMM TLKRS
81Ø  DIM Q(7): INPUT "KZ";KZ: INPUT "T1Z";TZ: INPUT "KU";KU
82Ø  INPUT "TL";TL: INPUT "T1U";TU
83Ø  LET T = TL/4
84Ø  LET AU = EXP(−T/TU): LET AZ = EXP(−T/TZ): LET B1 = KZ * (1 − AZ)
85Ø  LET A1 = −(AU + AZ): LET A2 = AU * AZ
86Ø  GOTO 3Ø
1ØØØ REM UNTERPROGRAMM
1Ø1Ø REM PARAMETERUEBERGABE
1Ø2Ø LET KP = X(1): LET TN = X(2)
1Ø3Ø LET B4 = KP * KU * (1 − AU): LET B = 1 − T/(2 * TN): LET BQ = 1 + T/(2 * TN)
1Ø4Ø LET F = Ø: FOR E = 1 TO 7: LET Q(E) = Ø: NEXT E
1Ø5Ø FOR K = Ø TO 8Ø
1Ø6Ø LET X = Q(1)
1Ø7Ø LET Q(1) = Q(2) + B1 − (A1 − 1) * X
1Ø8Ø LET Q(2) = Q(3) − (1 + AU) * B1 − (A2 − A1) * X
1Ø9Ø LET Q(3) = Q(4) + AU * B1 + A2 * X
11ØØ LET Q(4) = Q(5)
111Ø LET Q(5) = Q(6) − B4 * X * BQ
112Ø LET Q(6) = Q(7) + (B + AZ * BQ) * B4 * X
113Ø LET Q(7) = −AZ * B * B4 * X
114Ø LET F = F + K * ABS(X): NEXT K
115Ø RETURN
```

Anstelle der Abbruchbedingung (120) kann auch Bedingung (119) programmiert werden. Dazu sind im Programm folgende Änderungen vorzunehmen:

13Ø LET I = 1; der Rest entfällt
16Ø, 18Ø: Zeilen entfallen
195 ((einfügen)) IF ABS(F − FØ) < 1E − 5 GOTO 54Ø.

Es kann sein, daß das Optimum mit dieser Form der Abbruchbedingung sicherer gefunden wird als mit der anderen. Das hängt sowohl von der Art der Zielfunktion als auch von den gewählten Startwerten ab.
Wenn die Berechnung eines Funktionswerts längere Zeit in Anspruch nimmt (bei dem im Programm vorgestellten Anwendungsbeispiel dauert die Berechnung eines Funktionswerts auf dem KC 85/3 etwa 12 s, auf dem ZX Spectrum etwa 18 s), dann erhält der Nutzer über längere Zeit keine Information, ob der Rechenprozeß überhaupt noch voranschreitet. Deshalb kann man nach dem Unterprogrammaufruf in Zeile 19Ø ein akustisches Signal (BEEP) oder ein Zeichen auf dem Bildschirm einfügen (PRINT "!";), wobei die letztgenannte Ausgabe es auch möglich macht, nachzuzählen, wieviel Funktionsaufrufe erfolgt sind. Natürlich kann man zu diesem Zweck auch eine Zählvariable mitführen.
Der Selektor SL stellt fest, ob alle $a_j < 0.5$ sind (Zeile '17' bzw. 28Ø bis 31Ø). Zeile 32Ø entspricht der PAP-Zeile '18', Zeile 33Ø der Zeile '19'. In den Zeilen 34Ø bis 53Ø werden die neuen Richtungsvektoren zusammengefaßt zum Feld V(N,N) berechnet, wie das aus dem PAP Bild 32 erkennbar ist.
Die Zeilen 8ØØ bis 115Ø im Programm ROSE sind problemspezifisch. Als Anwendungsbeispiel wurde die Aufgabe programmiert, optimale Reglerparameter K_P und T_n für einen Regelkreis mit Laufzeit und Verzögerung zu bestimmen.
Deshalb wurde auf das Programm TLKRS zurückgegriffen, aus dem diejenigen Zeilen übernommen werden konnten, die dazu dienen, das Störverhalten des Regelkreises bei Sprungstörungen zu ermitteln. Unnötige Eingaben, etwa zur Dialogführung oder für andere Eingangssignalformen, wurden weggelassen. Damit waren vor allem die Zeilen 3Ø, 4Ø, 2ØØ bis 23Ø, 27Ø, 34Ø bis 4ØØ, 47Ø des Programms TLKRS tw. in modifizierter Form zu übernehmen. Für $d = T_L/T$ wurde der Wert 4 fest vereinbart; damit wird DIM Q(7) (Zeile 81Ø).
Einige Konstanten im Programm TLKRS sind unabhängig von den Werten der Reglerparameter (AU, AZ, B1, A1, A2); sie werden daher nur einmal nach Eingabe der Streckenparameter berechnet (Programm ROSE, Zeilen 81Ø bis 85Ø). Die anderen Konstanten (B4, B, BQ) hängen von den Reglerparametern ab; sie müssen bei jedem Funktionsaufruf neu berechnet werden und stehen deshalb im Unterprogramm, das bei Zeile 1ØØØ beginnt und in den Zeilen 7Ø und 19Ø des Hauptprogramms ROSE aufgerufen wird. Zeile 1Ø2Ø dient der Anpassung der Namen für die Entscheidungsvariablen im Hauptprogramm (X(1), X(2)) und Unterprogramm (KP, TN).
In Zeile 114Ø wird das Gütemaß F berechnet. Programmiert ist das ITAE-Kriterium nach (117). Für m, die Anzahl der berechneten x-Werte, wurde 8Ø gewählt, nachdem durch Voruntersuchungen festgestellt worden war, daß damit die Regelvorgänge bei günstigen Reglerparametern gut abgeklungen sind.
Soll als Gütemaß die quadratische Regelfläche verwendet werden, dann muß die Zeile lauten

114Ø LET F = F + X * X: NEXT K.

Vergleicht man die sich entsprechenden Zeilen 51Ø (TLKRS) und 1Ø4Ø (ROSE), dann fällt auf, daß in ROSE ein anderer Laufindex (E) gewählt wurde als in TLKRS (I). Im Hauptprogramm ROSE wird der Laufindex I nämlich in der Schleife verwendet, in der das UP aufgerufen wird (Zeilen 13Ø bis 33Ø). I darf deshalb keinesfalls durch das Unterprogramm verändert werden; deshalb ist dort ein anderer Name zu wählen.
Gestartet wird das Gesamtprogramm mit RUN 8ØØ. Nach Eingabe der Streckenparameter durch den Nutzer springt das Programm dann von Zeile 86Ø in das eigentliche Optimierungsprogramm, wobei die Dimension (2) und die Startwerte für X(1) = KP und X(2) = TN einzugeben sind.
Das Anwendungsbeispiel, das allerdings mit der im Text erwähnten Form der Abbruchbedingung (119) für EPS = 1E − 4 durchgerechnet wurde, erbrachte folgende Ergebnisse:

$K_z = 50$, $T_{1z} = 12$, $K_u = 1$, $T_L = 4$, $T_{1u} = 15$.

Startwerte für die Optimierungsrechnung

$K_P = 2$, $T_n = 20$.

Schrittweiten: $h_1 = 0.1$, $h_2 = 0.5$.

ITAE-Kriterium

$K_P = 2.652$, $T_n = 12.33$.

Diese Ergebnisse wurden nach 104 Funktionswertberechnungen und einer Rechenzeit (KC 85/3) von 22.5 min erreicht.

Quadratische Regelfläche

$K_P = 3.9$, $T_n = 15.54$

134 Funktionswertberechnungen, Rechenzeit 28.5 min.
Die Regelvorgänge für diese Reglereinstellungen sind im Bild 30 gezeigt worden.

Literaturverzeichnis

[1] *Brack, G.:* Dynamische Modelle verfahrenstechnischer Prozesse. Berlin: VEB Verlag Technik 1972

[2] *Brack, G.:* Einfache Modelle kontinuierlicher Prozesse. Berlin: VEB Verlag Technik 1982

[3] *Wede, J.; Werner, D.:* Echtzeitprozeßmodelle auf der Basis von Parameterschätzverfahren. Berlin: VEB Verlag Technik 1985

[4] *Brack, G.; Helms, A.:* Automatisierungstechnik. 2. Aufl. Leipzig: VEB Deutscher Verlag für Grundstoffindustrie 1987

[5] *Zurmühl, R.:* Praktische Mathematik. Berlin, Heidelberg, New York: Springer-Verlag 1965

[6] *Kopchenova, N.V.; Maron, I.A.:* Computational mathematics. Moskau: Mir 1975

[7] *Göldner, K.:* Lineare Systeme der Regelungstechnik. Berlin: VEB Verlag Technik 1973

[8] *Göldner, K.:* Mathematische Grundlagen der Systemanalyse. Bd. 2. Leipzig: Fachbuchverlag 1982

[9] *Faddejew, D. K.; Faddejewa, W. N.:* Numerische Verfahren der linearen Algebra. Berlin: VEB Deutscher Verlag der Wissenschaften 1973

[10] *Kerner, I. O.:* Numerische Mathematik und Rechentechnik. T. 2/1. Leipzig: Teubner Verlagsgesellschaft 1973.

[11] *Rint, C.* (Hrsg.): Handbuch für Hochfrequenz- und Elektrotechniker, Bd. 3. Berlin: Verlag für Radio-Foto-Kinotechnik 1954

[12] *Günther, M.:* Zeitdiskrete Steuerungssysteme. Berlin: VEB Verlag Technik 1986

[13] *Erfurth, H.; Bieß, G.:* Optimierungsmethoden. Leipzig: VEB Deutscher Verlag für Grundstoffindustrie 1975

[14] *Hartmann, K.* (Hrsg.): Analyse und Steuerung von Prozessen der Stoffwirtschaft. Berlin: Akademie-Verlag, Leipzig: VEB Deutscher Verlag für Grundstoffindustrie 1971.

Sachwörterverzeichnis

ISBN 3-341-00531-5