

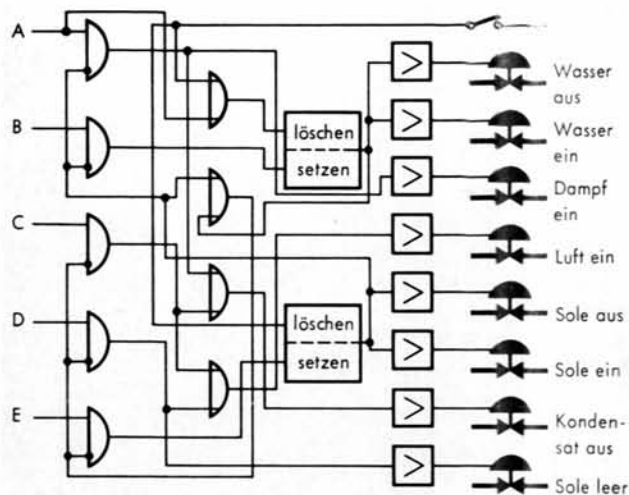


Bild 12. Beispiel: Verriegelung.

Ein Rechteckgenerator steuert einen aus pneumatischen Logik-Elementen bestehenden Ringzähler an, der nacheinander den Weg zu den einzelnen Ventilen freigibt. Durch einen weiteren Impuls des Rechteckgenerators wird das angewählte Ventil für eine definierte Zeit geöffnet. Der eine Impulsformer spricht auf die ansteigende Flanke, der andere Impulsformer auf die abfallende Flanke an.

Im Bild 12 ist als Beispiel die gegenseitige Verriegelung der Energieversorgung eines Reaktionskessels der Verfahrenstechnik dargestellt. Der Kessel ist an verschiedene Energiearten (Dampf, Wasser, Sole usw.) angeschlossen, die über manuell betätigte Drucktasten vom Bedienungsmann angewählt werden können. Es ist nun Aufgabe der Verriegelung, dafür zu sorgen, daß die Betätigung der Druckknöpfe durch den Bedienungsmann ausschließlich in sinnvoller Reihenfolge wirksam werden kann. Die Lösung der Aufgabe erfolgt praktisch durch eine sinnvolle Kombination von „Und“- und „Oder“-Funktionen.

Programmsystem zur Simulation allgemeiner Regelsysteme auf einem Digitalrechner

Von H. TRAUBOTH, München

Mit Hilfe des in der vorliegenden Arbeit beschriebenen Programmsystems kann die Dynamik von beliebig vermaschten allgemeinen Regelsystemen mit beliebig vielen Eingangssignalen auf einem elektronischen Digitalrechner simuliert werden. Die Eingangssignale und Gewichtsfunktionen des Systems können beliebige, auch numerisch gegebene Zeitfunktionen sein. Das Programmsystem sorgt für eine sehr einfache Eingabe der Struktur des Regelsystems in den Digitalrechner und berechnet die Signale als Funktionen der Zeit an allen Punkten des Systems.

The programming method described in this contribution enables the dynamics of any multiple-loop control system of a general nature, having any number of signals, to be simulated on an electronic digital computer. The input signals and weighting functions of the system may be of any kind, even numerical time functions. This method renders it very easy to feed the structural characteristics of the control system into the digital computer and enables the signals to be calculated as time functions in all points of the system.

1. Einleitung

Die Dynamik von physikalischen Systemen wird allgemein durch Differentialgleichungen beschrieben. Da die analytische Berechnung dieser Gleichungen selbst für wenig vermaschte Systeme oft auf große Schwierigkeiten stößt, verwendet besonders der Ingenieur zur Ermittlung des zeitlichen Verhaltens eines Systems gern einen elektronischen Analogrechner. Nun gibt es aber viele Probleme, für deren Lösung ein elektronischer Digitalrechner durch seine größere Flexibilität oft besser geeignet ist. Es seien hier nur einige Punkte angegeben, in denen der Digitalrechner durch seine andersartige Struktur und seine größere Flexibilität dem Analogrechner überlegen sein kann:

a) Die Genauigkeit der Rechenergebnisse kann praktisch beliebig verändert werden und einen höheren Grad als bei einem Analogrechner erreichen.

b) Physikalisch schwer realisierbare, unstetige oder nur als Meßergebnisse vorliegende Funktionen und Eingangssignale können ohne erhöhten Apparatenaufwand nachgebildet werden. Teure Funktionsgeneratoren für nichtlineare Glieder entfallen. Totzeiten können exakt nachgebildet werden.

c) Bei Zunahme des Umfangs des zu simulierenden Systems wächst nicht der Apparatenaufwand, sondern es erhöhen sich nur die Arbeitszeit und der Speicherbedarf des Digitalrechners.

d) Das System kann durch Ändern der Parameter, der Funktionen und der Verknüpfung seiner Glieder unvergleichlich schnell umgestaltet werden. Diese Veränderungen können zur Erfüllung bestimmter vorgegebener Kriterien automatisch vorgenommen werden. Die Simulation kann damit auch der Synthese und Optimierung eines Systems dienen.

Um die Vorteile des Digitalrechners zu nutzen, ohne dabei die relativ einfache Programmierung des Analogrechners einzubüßen, hat man kürzlich damit begonnen, Analogrechner auf Digitalrechnern zu simulieren [1], [2]. Da die hierbei simulierten Analogrechner-Bausteine — wie Integratoren, Addierer etc. — einfache Zeitfunktionen nachbilden, muß im allgemeinen das zu untersuchende System aus sehr vielen Bausteinen aufgebaut werden. Je größer aber die Zahl der Bausteine ist, desto mehr Rechenzeit ist erforderlich. Bausteine, die komplexere Funktionen ausführen, können daher die Rechenzeit verkürzen.

In vielen Fällen, z. B. bei der Regelung von Prozessen, kennt man die Differentialgleichungen nicht, die den Prozeß steuern. Man mißt deshalb nach Einspeisen von Testsignalen die Gewichtsfunktion des Prozesses, die dann nicht als analytischer Ausdruck, sondern als eine Reihe numerischer Werte vorliegt [3]. Solche physikalischen Systeme können nicht unmittelbar mit Analogrechner-Bausteinen dargestellt werden.

Aus diesen Gründen soll hier eine andere Art der Darstellung gewählt werden. In der Regelungstechnik ist es üblich, ein automatisches Regelsystem als Blockschaltbild darzustellen [4]. Das Blockschaltbild besteht aus verschiedenartigen Gliedern, die zu einem stark vermaschten Gesamtsystem beliebig miteinander verbunden werden können. Der Begriff „Regelsystem“ sei hier im weiteren Sinn für ein physikalisches System verstanden, das als Blockschaltbild angegeben werden kann [5]. So kann man ein elektrisches Netzwerk auch als Regelsystem auffassen, wenn es sich in ein Blockschaltbild umformen läßt. Gewöhnliche Differentialgleichungssysteme lassen sich ebenfalls als Blockschaltbilder darstellen. Die Analogrechner-Bausteine können ebenfalls mit diesen Gliedern gebildet werden.

Wenn man in der Lage ist, diese Blockschaltbilder auf einem Digitalrechner zu berechnen, so kann man Systeme gewöhnlicher Differentialgleichungen, technisch realisierte, aber mathematisch nicht voll erfassbare Regelsysteme und elektrische Netzwerke gleichermaßen simulieren.

Im folgenden wird ein Programmsystem beschrieben, das die Simulation solcher universeller Blockschaltbilder auf einem Digitalrechner erlaubt. Die Bedienung des Digitalrechners ist dabei so einfach, daß auch ein Laie alle notwendigen Daten über das zu simulierende System in den Rechner eingeben kann.

2. Aufgabe des Programmsystems

Das zu simulierende Blockschaltbild ist aus mehreren rückwirkungsfreien Gliedern zusammengesetzt, die verschiedene Funktionen darstellen können (Bild 1), und zwar bei *linearen* Systemen:

1. eine Gewichtsfunktion $g(t)$, die entweder als analytischer Ausdruck oder als numerische Wertetabelle bekannt ist;

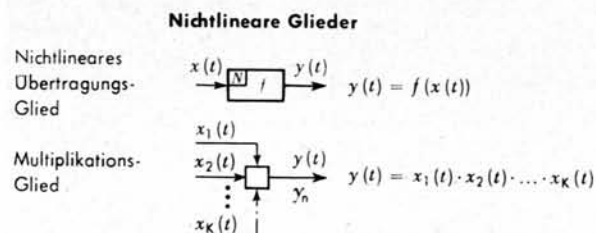
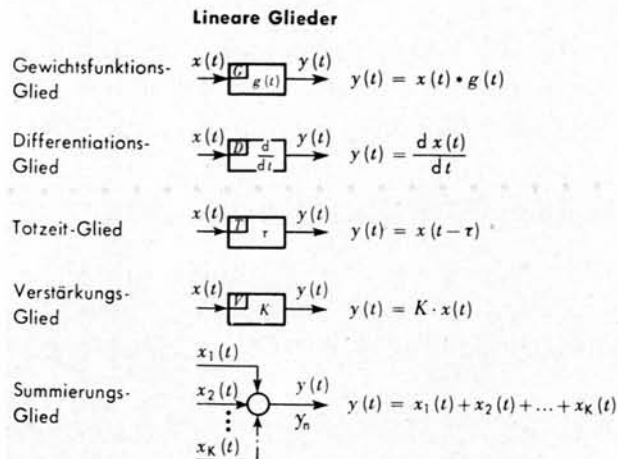


Bild 1. Die verschiedenartigen Glieder, aus denen ein Blockschaltbild zusammengesetzt sein kann, und die Beziehungen zwischen Eingangs- und Ausgangssignal als Funktionen der Zeit t .

2. einen Integrator als Sonderfall der Gewichtsfunktion ($g(t) = 1$);
 3. einen Differentiator 1. Ordnung;
 4. eine Totzeit bekannter Länge;
 5. einen Verstärker bekannter positiver oder negativer Verstärkung;
 6. einen Summierer mit praktisch beliebig vielen Eingängen.
- Bei *nichtlinearen* Systemen kommen noch zwei Arten von Gliedern hinzu:

7. das nichtlineare Übertragungsglied;
 8. der Multiplikator mit praktisch beliebig vielen Eingängen.
- Diese Glieder lassen ein Signal nur in einer Richtung durch. Sie sollen beliebig miteinander zu einem umfangreichen Gesamtsystem verknüpft werden können. In das System sollen sich praktisch unbegrenzt viele Eingangssignale einspeisen lassen, deren Zeitfunktionen entweder als analytischer Ausdruck oder als Wertetabelle bekannt sind und die auch Impulsfunktionen sein können. Anfangswerte des Systems werden in einfacher Weise wie Eingangssignale behandelt. An den Anwendungen des Programmsystems werden keine besonderen Anforderungen gestellt. Er schreibt lediglich die Struktur des Blockschaltbildes mit den Daten, die die Glieder und Eingangssignale kennzeichnen, und einige Zusatzangaben über die gewünschten Ausgangssignale in einfach normierter Weise auf ein Blatt Papier. Durch das Programmsystem werden dann vom Rechner die Signale an den gewünschten Punkten des Systems als Funktionen der Zeit berechnet und protokolliert. Das Programmsystem soll aus mehreren relativ unabhängigen Teilprogrammen aufgebaut sein, wodurch es leicht ausbaufähig ist und als Kernstück eines größeren Programmkomplexes verwendet werden kann.

3. Aufbau des Programmsystems

Die dem Programmsystem zugrundeliegende mathematische Methode ist in [6] beschrieben.

Dem Rechner muß das zu simulierende Blockschaltbild in vereinbarter Form mitgeteilt werden. Dazu sind einige Vorbereitungen nötig. Die Verbindungen zwischen den einzelnen Gliedern werden mit der Zahl 1 beginnend fortlaufend durchnummeriert, wobei keine Nummer mehrmals vorkommen darf. Es ist dabei belanglos, welche Nummer einer Verbindung zugeteilt wird. Dadurch erhält jedes Glied eine Eingangs- und eine Ausgangsnummer, so daß sein Ort im System eindeutig festgelegt ist. Die externen Eingangssignale, das sind die über Summierer von außen in das Gesamtsystem eingespeisten Signale, werden ebenfalls unter sich mit „1“ beginnend fortlaufend durchnummeriert. Außerdem ist vom Anwender nur ein Protokoll über die Eigenschaften der Glieder und externen Eingangssignale niederzuschreiben.

Das Programmsystem gliedert sich in mehrere Teile. Das sind

1. die Listen,
2. das Arbeitsprogramm,
3. das Druckprogramm,
4. der Protokoll-Übersetzer,
5. das Dateneinleseprogramm.

Diese Programme setzen sich wiederum aus abgeschlossenen Unterprogrammen zusammen.

Der Protokoll-Übersetzer, das Druck- und Dateneinleseprogramm bedürfen keiner numerischen Analyse und bereiten keine prinzipiellen Schwierigkeiten.

Für die Simulation eines Regelsystems sind eine Vielzahl von Daten erforderlich, die nach bestimmten Gesichtspunkten geordnet in vorgeschriebenen Speicherbereichen ab-

gelegt werden. Diese geordneten Daten stellen Listen dar, die verschiedenen Aufgaben dienen. Die wichtigsten Listen sind

1. die Struktur-Liste,
2. die Diskretwert-Liste,
3. die Matrix-Liste.

Die *Struktur-Liste* ist ein Abbild der Struktur des zu simulierenden Blockschaltbilds im Arbeitsspeicher des Rechners. Sie enthält alle Daten, die zur Beschreibung des zu simulierenden Systems benötigt werden. Jedes Glied und jedes externe Eingangssignal ist

1. durch eine Adresse und anschließend
2. durch eine Funktionsbeschreibung

gekennzeichnet. Die Adresse gibt den Typ, die Eingangs- und Ausgangs-Nummer des Glieds an und besagt, ob die zugehörige Zeitfunktion analytisch oder numerisch gegeben ist. Die Funktionsbeschreibung umfaßt für jeden Glied-Typ eigene Parameter. So wird z. B. die einem Gewichtsfunktions-Glied bzw. externen Eingangssignal zugeordnete analytisch gegebene Zeitfunktion TF_1 durch die Parameter C , K_v , A_v , B_v , F_v und D_v beschrieben. Diese Zeitfunktion lautet

$$TF_1 = \sum_{v=0}^C K_v \cdot e^{-A_v t} \cdot t^{D_v} \cdot \cos(B_v t + F_v). \quad (1)$$

In der *Diskretwert-Liste* sind die diskreten Werte aller Zeitfunktionen des Systems nach der Zeit geordnet abgelegt. Für jeden diskreten Zeitpunkt werden dort die numerischen

Werte der Signale, die an allen Ausgängen der Glieder des Systems erscheinen, in der natürlichen Folge der Nummern dieser Ausgänge eingetragen. Daran schließen sich die diskreten Werte der Gewichtsfunktionen und externen Eingangssignale in der Reihenfolge an, wie deren Adressen in der *Struktur-Liste* stehen.

Wie in [6] erläutert wurde, ist für jeden diskreten Zeitpunkt ein algebraisches Gleichungssystem zu lösen, dessen Größe von der Struktur des rückgekoppelten Systems abhängt. Zur Lösung dieses Gleichungssystems sind die Koeffizienten in der *Matrix-Liste* abzuspeichern.

Das *Arbeitsprogramm*, das aus mehreren Teilprogrammen besteht, führt alle für die Simulation notwendigen Berechnungen durch. Aus Bild 2 kann man die Aufgaben und die Reihenfolge der Abarbeitung der Teilprogramme ersehen. Das Teilprogramm DISWET wird nur einmal zu Beginn der Simulation durchlaufen und berechnet aus den Daten der *Struktur-Liste* die diskreten Werte aller Zeitfunktionen, die in analytischer Form durch Parameter gegeben sind.

Es speichert sie dann in der *Diskretwert-Liste* ab. Berechnet man die Zeitfunktion TF_1 nur für äquidistante Zeitpunkte, so läßt sich Gl. (1) durch das Additionstheorem so zerlegen, daß nur zu Beginn der Berechnung ($t = \Delta t$) die Unterprogramme für die e- und sin-Funktion herangezogen zu werden brauchen. Für jeden folgenden Zeitpunkt sind nur einfache Multiplikationen auszuführen. Durch Hinzufügen entsprechender Unterprogramme können auch beliebige andere Zeitfunktionen behandelt werden. Die Teilprogramme AEDZAE, EINTRG, LINGL und DRUCK werden für jeden diskreten Zeitpunkt durchlaufen.

Das Ausdrucken der Ergebnisse der Simulation und der Fehlermeldungen des Protokoll-Übersetzers besorgt in geeignetem Format und mit Text versehen das *Druckprogramm*. Dem Anwender des Programmsystems soll möglichst wenig Arbeit zur Vorbereitung der Simulation zugemutet werden. Er soll deshalb dem Rechner nur die unbedingt notwendigen Angaben über das zu simulierende System in einer normierten Form mitteilen, eine Form, die einprägsam und leicht zu erlernen ist und zu keinen Schreibfehlern Anlaß gibt. Das Eingabe-Protokoll wird auf keinem starr eingeteilten Formular, sondern auf einem normalen Schreibpapier in bestimmter Reihenfolge niedergeschrieben. Fest zugeordnete Worte und Zeichen dienen der Abgrenzung bestimmter Teile des Protokolls.

Der *Protokoll-Übersetzer* hat die Aufgabe, das Eingabe-Protokoll in die *Struktur-Liste* zu übersetzen, wobei er die Bemerkungen überliest. Während des Übersetzens errechnet er außerdem Zählzeiten und Adressen von Arbeitsspeicherelementen, die für das Arbeitsprogramm benötigt werden. Er stellt formale Fehler innerhalb des Protokolls fest. Falls nur ein kleiner Arbeitsspeicher zur Verfügung steht, kann der Übersetzer vom Arbeitsprogramm überschrieben werden, so daß er bei Wiederverwendung in den Rechner neu einlaufen muß. Im Bild 3 ist ein Beispiel eines Protokolls angegeben.

Liegen externe Eingangssignale und Gewichtsfunktionen numerisch vor, so werden diese diskreten Werte vom Lochstreifen oder von der Lochkarte mit Hilfe des *Dateneinlese-Programms* direkt in die *Diskretwert-Liste* eingelesen.

4. Die Rechenergebnisse

Für die Siemens-Datenverarbeitungsanlage-2002 wurde ein Programmsystem (ohne Protokoll-Übersetzer und Dateneinleseprogramm) mit insgesamt etwa 1200 Befehlen zur Simulation linearer Regelsysteme aufgestellt. Die Faltung und Differentiation werden dabei linear angenähert. Es wur-

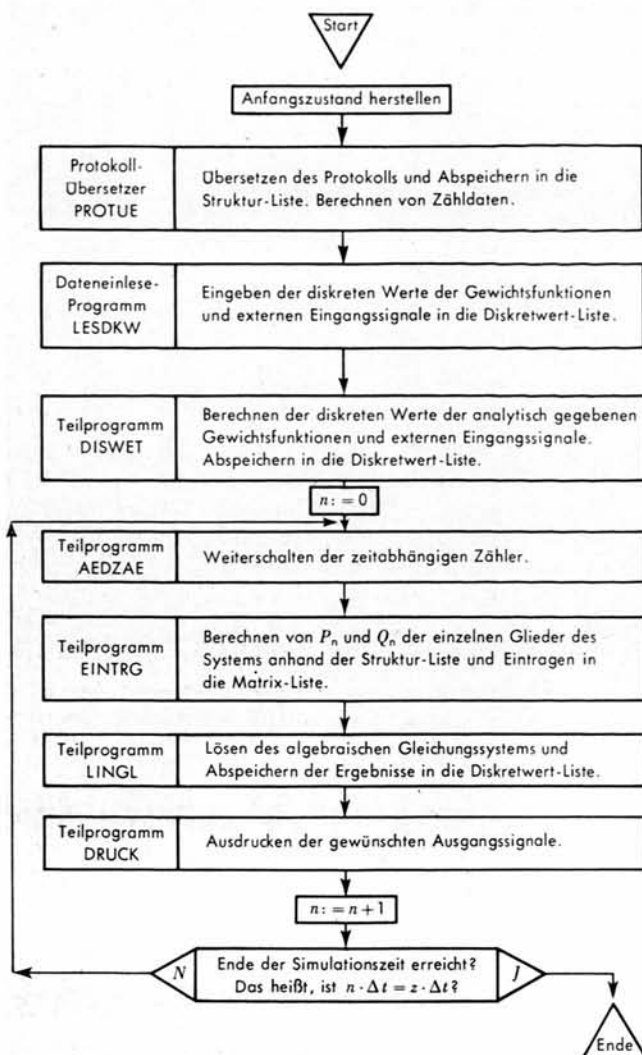


Bild 2. Übersichtsflußdiagramm des Programmsystems zur Simulation allgemeiner Regelsysteme.

ABTASTINTERVALL DT = 0.1
 ANZAHL VON DT Z = 100
 AUSDRUCKEN NR = 05, 01;
 ANZAHL SIGNALE S = 05

STRUKTUR DES SYSTEMS

V0 03 05 K = -1
 D0 04 02
 S0 01 00 NR = 03, B 01;
 L0 05 04 T = 0.1
 GA 02 01 TYPE = 01, C = 02
 K1 = 1, A1 = 0, B1 = 0, F1 = 0, D1 = 0
 K2 = -1, A2 = 1, B2 = 0, F2 = 0, D2 = 0
 FA 01 00 TYPE = 02
 K = 1
 ENDE

Bild 3. Das Eingabe-Protokoll für das Regelsystem des Bildes 6 c.

den mehrere Blockschaltbilder als Beispiele simuliert und mit den Ergebnissen, die auf anderem Wege berechnet wurden, verglichen. Aus diesen Beispielen wurden einige typische Fälle ausgewählt.

Beispiel 1:

Bild 4 zeigt die Druckregelung des Dampferzeugers eines Dampfkraftwerks [7]. Bei diesem Regelsystem wird die Wirkung einer Störung (Sprungfunktion) der Ventilstellung H_v und des Dampfverbrauchs Q_B getrennt untersucht. In die von einem Analogrechner aufgezeichneten Ausgangssignale wurden die vom Programmsystem berechneten diskreten Werte zum Vergleich eingetragen (Bild 5). Es wurde mit einem Abtastintervall $\Delta t = \tau = 16$ s gerechnet. Die Rechenzeit (ohne Druckzeit) beträgt 3,2 min gegenüber der echten Zeit von 26 min.

Beispiel 2:

Die Blockschaltbilder im Bild 6 a bis c stellen ein einfaches Regelsystem mit einem Totzeit-Glied dar. Das Ausgangssignal dieses Systems läßt sich noch mit verhältnismäßig erträglichem Aufwand analytisch berechnen, wenn am Eingang eine Sprungfunktion anliegt. Es wurden drei verschie-

dene Blockschaltbilder für ein und dasselbe Regelsystem simuliert, um den Einfluß eines Differentiations-Gliedes und den Einfluß einer Totzeit vor einem Gewichtsfunktions-Glied auf die Genauigkeit der Rechenergebnisse feststellen zu können. Im Bild 7 sind die relativen Fehler dieser Werte graphisch aufgetragen. Die Rechenzeit des Programmsystems

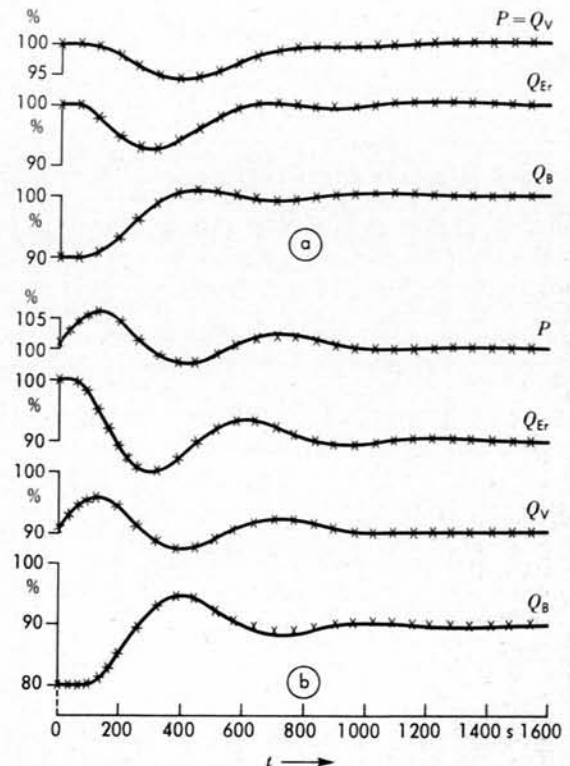


Bild 5. Verlauf der Ausgangssignale des im Bild 4 dargestellten Regelkreises.

a bei Störung Beheizung $\Delta Q_B = -10\%$ Sprung,
 b bei Störung Ventilstellung $\Delta H_v = -10\%$ Sprung.

— Ergebnisse des Analogrechners,
 × × Ergebnisse des Digitalrechners.

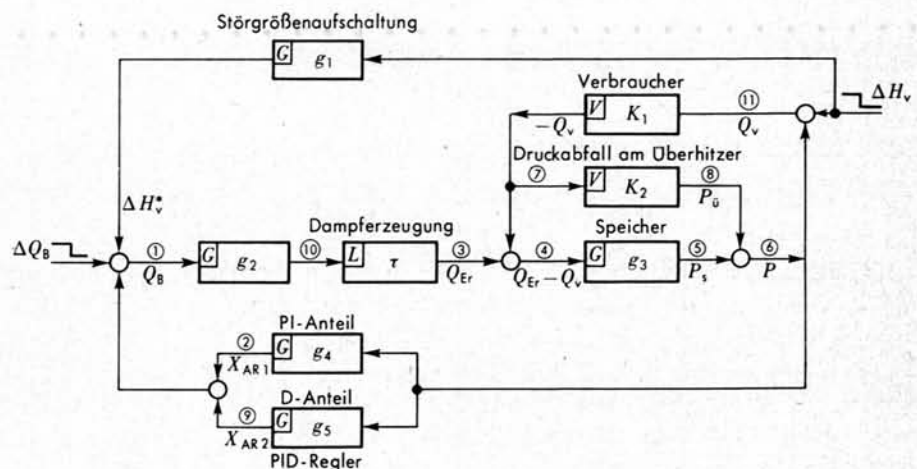


Bild 4. Blockschaltbild der Druckregelung eines Dampfdruckerzeugers (siehe auch [7], S. 259).

$$g_1 = L^{-1} \left\{ 0,5 + \frac{1,5 \cdot s}{s + 0,01} \right\},$$

$$g_2 = 0,42 \cdot 10^{-7} \cdot t^3 \cdot e^{-0,0224215 \cdot t},$$

$$g_3 = 0,00869565,$$

$$g_4 = -0,8 \cdot u(0) - 0,00482,$$

$$g_5 = -0,8 \cdot u(0) + 0,00695 \cdot e^{-0,00869565 \cdot t},$$

$$K_1 = -1,$$

$$K_2 = 0,119,$$

$$\tau = 16 \text{ s},$$

$$Q_B = H_v = 100\%.$$

Fall 1:

Störung Beheizung $\Delta Q_B = -10\%$,
 Störung Ventilstellung $\Delta H_v = 0$.

Fall 2:

Störung Beheizung $\Delta Q_B = 0$,
 Störung Ventilstellung $\Delta H_v = -10\%$,
 $\Delta H_v^* = -5 - 15 \cdot e^{-0,01 t}.$

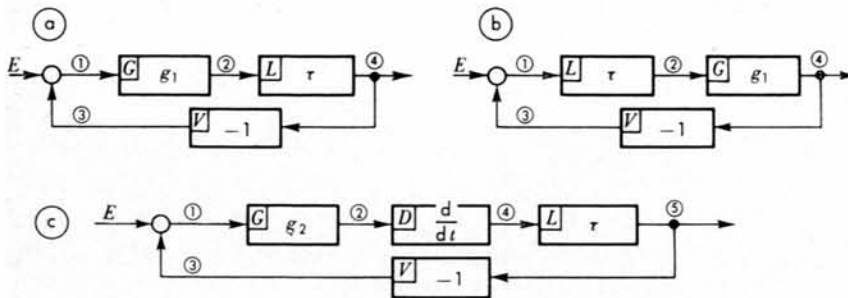


Bild 6. Verschiedene Blockschaltbilder eines einfach rückgekoppelten Regelsystems mit einem Totzeit-Glied.

$E(t) = 1$, $\tau = 0,1$ s, $g_1 = e^{-t}$, $g_2 = 1 - e^{-t}$.

beträgt für die Berechnung aller Ausgangssignale (Abtastintervall $\Delta t = 0,1$ s und Anzahl der Zeitpunkte $z = 100$) 50 s. Dagegen benötigt die Berechnung nur eines Ausgangssignals x_4 für 100 Zeitpunkte nach der klassischen analytischen Methode auf dem gleichen Rechner 25 min.

Beispiel 3:

Das im Bild 8a abgebildete dreistufige RC-Filter wird in die beiden Blockschaltbilder Bild 8b und Bild 8c umgeformt. Das Ausgangssignal e_2 des Filters wurde zum Vergleich über ein spezielles Programm nach der analytischen Methode berechnet. Das Blockschaltbild Bild 8c wurde simuliert, um den Einfluß eines Differentiations-Gliedes auf die Genauigkeit der Rechenwerte zu beobachten. Im Bild 9 sind die relativen Fehler dieser Werte graphisch aufgetragen. Die Rechenzeit des Programmsystems beträgt für die Berechnung aller Ausgangssignale (Abtastintervall $\Delta t = 10$ ms und Anzahl der Zeitpunkte $z = 100$) 3,1 min.

5. Fehler der Rechenergebnisse

Anhand der gerechneten Beispiele können gewisse Aussagen über die Fehler der Rechenergebnisse gemacht werden. Die Ergebnisse des Beispiels 1 decken sich mit den Kurven des Analogrechners fast überall. Da die vom Analogrechner aufgezeichneten Kurven bereits bei Vollausschlag nicht mit größerer Genauigkeit als etwa 5% abgelesen werden können, eignen sie sich nicht für Vergleiche von größerer Genauigkeit und zur Feststellung von Fehlern bei kleinen Ausschlägen.

Der Vergleich der vom Programmsystem errechneten Werte mit den über analytischen Formeln ermittelten Werten gibt einen besseren Aufschluß über den Verlauf der Fehler besonders im Bereich kleiner Fehler. Als relativer Fehler F wird definiert:

$$F = \frac{x_R(n \cdot \Delta t) - x_E(n \cdot \Delta t)}{x_M} 100\%; \quad n = 0, 1, 2, \dots; \quad (2)$$

$x_R(n \cdot \Delta t)$ vom Programmsystem errechneter Wert für $t = n \cdot \Delta t$,

$x_E(n \cdot \Delta t)$ analytisch berechneter Wert für $t = n \cdot \Delta t$,

x_M analytisch berechneter maximaler Wert.

Diese Fehler wurden auf doppelseitig logarithmischem Papier aufgetragen, um auch die Fehler in der Nähe des Nullpunkts mit genügender Genauigkeit verfolgen zu können (Bild 7 und Bild 9). An diesen Fehlerkurven kann man beobachten, daß

1. eine Verringerung des Abtastintervalls Δt die Kurven fast parallel in Richtung kleinerer relativer Fehler F verschiebt,
2. bei Vorhandensein eines Differentiations-Gliedes der Fehler sein Vorzeichen ändert,
3. der relative Fehler keinen konstanten Wert hat. Aber bereits bei einem Abtastintervall von $1/100$ der Einschwingzeit ist der Fehler über den gesamten Zeitraum kleiner als 5%. Beim Blockschaltbild Bild 6a ist der Fehler sogar kleiner als 0,02%.

Die erste Beobachtung deckt sich mit der Tatsache, daß der Näherungsfehler der Integration bei der Faltung mit Δt^2 abnimmt. Dies kann durch Extrapolation zur Verbesserung der Ergebnisse ausgenutzt werden.

Die zweite Beobachtung veranschaulicht die ungenügende Näherung der Differentiation. Die Differentiation einer Zeitfunktion führt zu einer „aufgerauten“ Funktion, während die Integration eine „geglättete“ Funktion ergibt. Ein starker Wechsel der Steigung einer Kurve zwischen den Abtastpunkten führt bei der Differentiation zu stärkeren Fehlern als bei der Integration. Die Verwendung von Polynomen höherer Ordnung für die Näherung der Differentiation dürfte daher eine Verbesserung der Genauigkeit erbringen. Daß sich diese Fehler aber auch kompensieren können, zeigt Bild 9c.

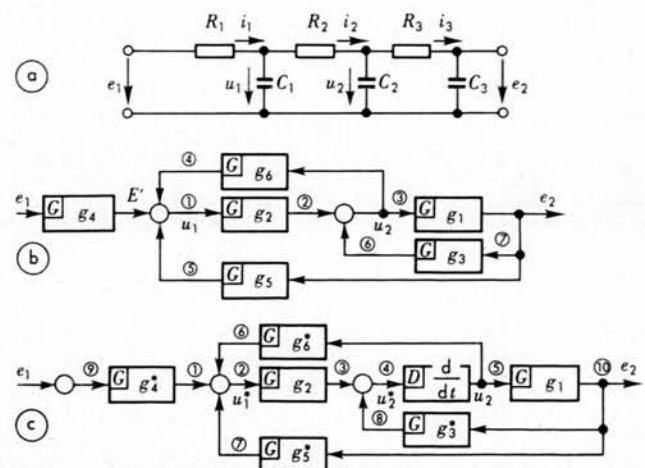


Bild 8. Darstellung eines dreistufigen RC-Filters.

a als elektrisches Netzwerk,

b als Blockschaltbild ohne Differentiations-Glied,

c als Blockschaltbild mit Differentiations-Glied.

$$e_1(s) = \frac{1}{s}, \quad E'(s) = e_1(s) \cdot G_4(s), \quad E'(t) = \frac{1}{T_1} (1 - e^{-t/T_1}),$$

$$C_3/C_1 = C_3/C_2 = C_2/C_1 = 1,$$

$$T_1 = R_1 C_1 = 0,0125 \text{ s},$$

$$T_2 = R_2 C_2 = 0,0375 \text{ s},$$

$$T_3 = R_3 C_3 = 0,0250 \text{ s},$$

$$g_1 = \frac{1}{T_3} \cdot e^{-t/T_3}, \quad u(0) = \text{Einheitsimpulsfunktion für } t = 0,$$

$$g_2 = \frac{1}{T_2} \cdot e^{-t/T_2},$$

$$g_3 = -\frac{C_3}{C_2} \left(u(0) - \frac{1}{T_2} \cdot e^{-t/T_2} \right), \quad g_3^* = -\frac{C_3}{C_2} \cdot e^{-t/T_2},$$

$$g_4 = \frac{1}{T_1} \cdot e^{-t/T_1}, \quad g_4^* = \frac{1}{T_1} (1 - e^{-t/T_1}),$$

$$g_5 = -\frac{C_3}{C_1} \left(u(0) - \frac{1}{T_1} \cdot e^{-t/T_1} \right), \quad g_5^* = -\frac{C_3}{C_1} \cdot e^{-t/T_1},$$

$$g_6 = -\frac{C_2}{C_1} \left(u(0) - \frac{1}{T_1} \cdot e^{-t/T_1} \right), \quad g_6^* = -\frac{C_2}{C_1} \cdot e^{-t/T_1}.$$

Bild 7. Der Verlauf und die relativen Fehler der vom Programmsystem errechneten Ausgangssignalwerte x_2 bzw. x_3 des im Bild 6 gezeigten Regelsystems für verschiedene Abtastintervalle Δt .

a Exakter Verlauf des Ausgangssignals, c Relativer Fehler beim Blockschaltbild Bild 6 b *),

b Relativer Fehler beim Blockschaltbild Bild 6 a, d Relativer Fehler beim Blockschaltbild Bild 6 c.

*) Ohne Berücksichtigung der Unstetigkeit am Eingang von g_1 (siehe [6]).

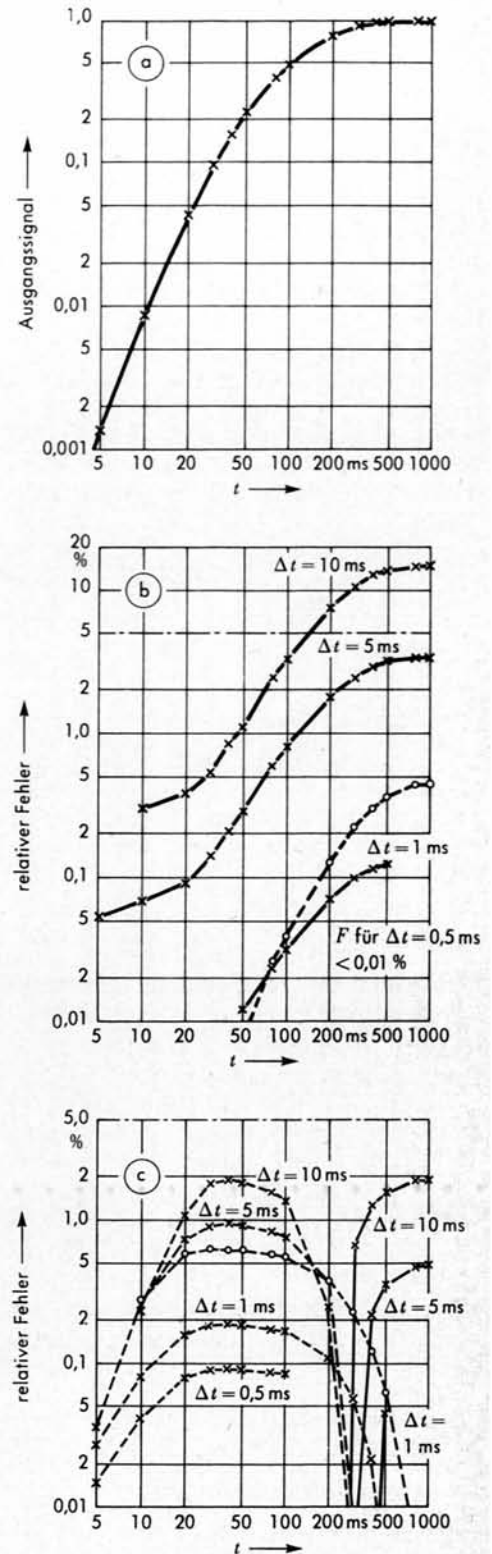
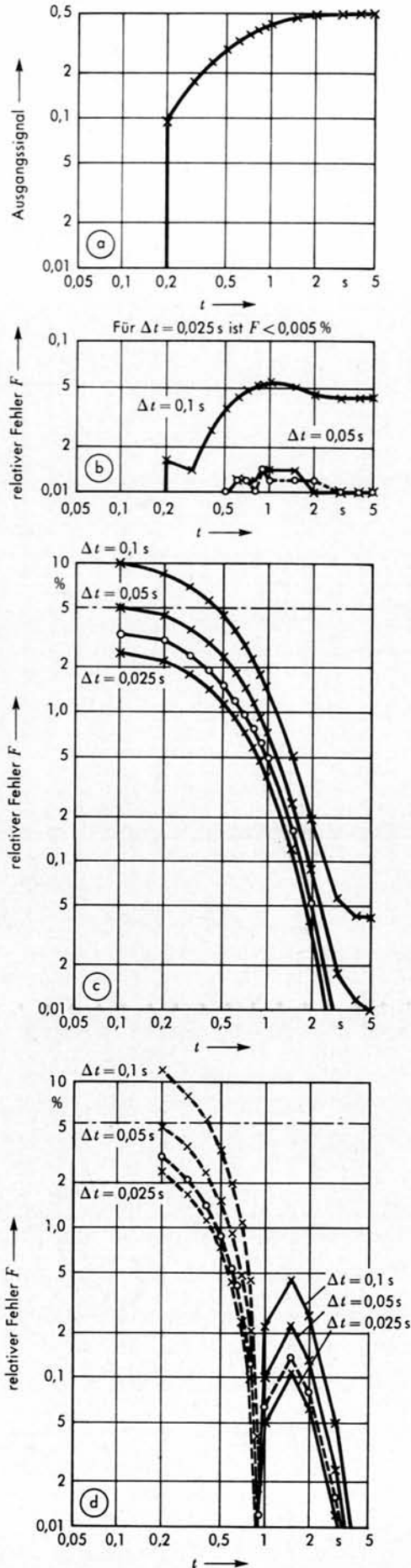


Bild 9. Der Verlauf und die relativen Fehler der vom Programmsystem errechneten Ausgangssignalwerte e_2 eines dreistufigen RC-Filters für verschiedene Abtastintervalle Δt (siehe Bild 8 a).

a Exakter Verlauf des Ausgangssignals,
b Relativer Fehler beim Blockschaltbild ohne Differentiationsglied (Bild 8 b),
c Relativer Fehler beim Blockschaltbild mit Differentiationsglied (Bild 8 c).

— negative Werte,
— positive Werte,
× normaler Wert,
○ extrapolierter Wert ($\Delta t_1 = 10 \text{ ms}$, $\Delta t_2 = 5 \text{ ms}$).

Die dritte Beobachtung deutet darauf hin, daß sich mehrere Fehler abhängig von der Struktur des Systems überlagern und mit der Zeit fortpflanzen.

Die Fehlerkurven des Bildes 7c zeigen eine deutliche Verschlechterung der Ergebnisse, wenn die Unstetigkeit des Signals am Eingang des Gewichtsfunktions-Gliedes zur Zeit $t = T \cdot \Delta t$ bei der Faltung nicht gesondert berücksichtigt wird (siehe [6], Abschnitt 3c).

Alle im Abschnitt 4 angeführten Beispiele wurden auch mit einem anderen Unterprogramm zur Lösung der linearen Gleichungen gerechnet, bei dem für jeden Zeitpunkt das lineare Gleichungssystem nach dem Gaußschen Eliminationsverfahren direkt gelöst wird. Der Rechnungsgang ist ein anderer als in [6], angegeben, und die Berechnungen dauern wesentlich länger. Die über dieses Unterprogramm errechneten Werte der Ausgangssignale decken sich mit den Zahlenwerten, die nach dem in [6] angeführten Verfahren ermittelt wurden und zwar bis auf einige wenige Werte, die sich aber erst ab der achten Stelle unterscheiden. Aus diesem Vergleich kann geschlossen werden, daß die in den Beispielen auftretenden Matrizen von guter Kondition sind und daß daher bei der Lösung der linearen Gleichungen gute Ergebnisse erwartet werden können.

6. Abschätzung der Rechenzeit

Die gesamte Rechenzeit für das ausgeführte Programmsystem (einschließlich der Zeit für die Berechnung der analytisch gegebenen Zeitfunktionen, aber ohne Druckzeit) kann nach folgender Formel abgeschätzt werden:

$$t_R \approx (3 \cdot s^3 + 3,2 \cdot s^2 \cdot z + 80 \cdot z \cdot (g + f) + 1,4 \cdot z^2 \cdot g) \cdot t_m; \quad (3)$$

s Anzahl der Ausgangssignale,

z Anzahl der Abtastintervalle Δt ,

f Anzahl der externen Eingangssignale,

g Anzahl der Gewichtsfunktions-Glieder,

t_m Dauer der Gleitkomma-Multiplikation.

Die über diese Formel errechneten Rechenzeiten stimmen mit den gemessenen Rechenzeiten bis auf eine Abweichung von maximal 10% der gemessenen Zeit überein. Der Hauptanteil der Rechenzeit kommt durch die Lösung der linearen Gleichungen und durch die Berechnung der Faltung zustande.

Um einen Begriff von der Größenordnung der Rechenzeit zu erhalten, sei angenommen, daß ein mittelgroßes und ein sehr großes Regelsystem jeweils auf einem mittleren und einem sehr schnellen Digitalrechner berechnet werden. Die Anzahl z der Abtastintervalle Δt sei 100. Als mittlerer Rechner sei der Siemens-2002 und als sehr schneller Rechner die neue IBM 360/70 zugrundegelegt. Die Gleitkommamultiplikation bei vergleichbarer Wortlänge dauert

bei der Siemens-2002 1,5 ms,

bei der IBM 360/70 4 μ s.

Die Kennzahlen seien für

das mittelgroße Regelsystem: $s = 8, g = 4, f = 1$;

das sehr große Regelsystem: $s = 40, g = 20, f = 1$.

	Siemens-2002	IBM 360/70
Mittelgroßes Regelsystem	3 min	0,48 s
Sehr großes Regelsystem	29 min	4,6 s

Aus diesen Werten läßt sich ersehen, daß man durchaus auch einen mittelgroßen Rechner im Echtzeitbetrieb innerhalb eines mittelgroßen Regelsystems einsetzen kann, wenn die Zeitkonstanten des Systems nicht zu klein sind.

7. Schlußwort

Das beschriebene Programmsystem kann man sich als Baustein eines Komplexes von Programmen vorstellen. So könnte ein zusätzliches Programm elektrische Netzwerke in Blockschaltbilder umformen. Dadurch wäre es möglich, die Struktur elektronischer Schaltungen nicht als Blockschaltbild, sondern direkt als Netzwerk mit den entsprechenden elektrischen Werten in den Rechner einzugeben.

Zusätzliche Programme für weitere Typen von Gliedern, die z. B. Hysteresis-Schleifen, Mehrpunkt-Schalter und logische Funktionen nachbilden, können leicht eingebaut werden.

Wird ein digitaler Rechner zur übergeordneten Regelung von Herstellungsprozessen als sogenannter Prozeßrechner eingesetzt, so wird ihm oft die Aufgabe obliegen, das zu regelnde System als Modell nachzubilden und mögliche Betriebsfälle an diesem durchzuspielen, bevor er Eingriffe am wirklichen System vornimmt. Auch hierfür kann das Programmsystem verwendet werden.

Durch weitere Zusätze kann das Programmsystem der Synthese und Optimierung von Regelsystemen und elektrischen Netzwerken dienen. Die Parameter einzelner Glieder und die Struktur des Systems werden systematisch und schnell nach einer Strategie solange verändert, bis bestimmte Kriterien der Ausgangssignale — wie z. B. maximale Anstiegszeit, minimales Überschwingen und minimale Regelfläche — erfüllt sind.

Es wäre auch denkbar, das Programmsystem zur Simulation von Abtastregelsystemen, von Regelsystemen mit zeitabhängigen Parametern oder von Regelsystemen, die bestimmten partiellen Differentialgleichungen gehorchen, heranzuziehen.

Ganz besonderer Dank gebührt den Herren Prof. Dr. L. Merz und Prof. Dr. F. L. Bauer, Technische Hochschule München, die durch ihren Rat und ihre Unterstützung diese Arbeit sehr gefördert haben. Der Verfasser dankt Herrn Dipl.-Math. D. Krönig, Technische Hochschule München, für die Hilfe bei der Aufstellung des Programms zur Lösung linearer Gleichungen. Er dankt ferner Herrn Dipl.-Ing. D. Winkler, Karlsruhe, für die am Analogrechner Beckman EASE 1100 durchgeführten Rechnungen. Dem Zentrallaboratorium der Siemens & Halske AG, München sei gedankt, daß für die Berechnungen eine Datenverarbeitungsanlage S 2002 zur Verfügung stand.

Literatur:

- [1] Gaskill, R. A.: A Versatile Problem-Oriented Language for Engineers. Trans. IEEE, EC-13, Nr. 4, August 1964, S. 415 bis 421.
- [2] Brennan, R. D., und Sano, H.: PACTOLUS — A Digital Analog Simulator Program for the IBM 1620. Fall Joint Computer Conference, San Francisco, Oktober 1964.
- [3] Mishkin und Braun: Adaptive Control Systems. McGraw Hill, New York 1961.
- [4] Oppelt, W.: Kleines Handbuch technischer Regelvorgänge. Verlag Chemie, Weinheim (Bergstr.) 1956.
- [5] Trauboth, H.: Die Umformung von gewöhnlichen Differentialgleichungen in Blockschaltbilder. Regelungstechnik 13 (1965), Heft 10, S. 487—493.
- [6] Trauboth, H.: Ein numerisches Verfahren zur Berechnung allgemeiner Regelsysteme. Regelungstechnik 13 (1965), Heft 11, S. 535—543.
- [7] Winkler, D.: Untersuchung verschiedener Verfahren für die Regelung des Druckes von Dampferzeugern. Regelungstechnik 11 (1963), S. 253—261.
- [8] Alt, F. L.: Electronic Digital Computers. Academic Press, New York 1958.