Zeitschrift: Schweizer Ingenieur und Architekt
Herausgeber: Verlags-AG der akademischen technischen Vereine
Band: 101 (1983)
Heft: 4

Artikel: Nichtlineare Finite-Element-Berechnungen und Informatik
Autor: Anderheggen, Edoardo
DOI: https://doi.org/10.5169/seals-75045

*Angewandte Mechanik in Industrie und Hochschule III**

Nichtlineare Finite-Element-Berechnungen und Informatik

Von Edoardo Anderheggen, Zürich

Es werden einige Eigenschaften des neuen, am Institut für Informatik der ETH Zürich entwickelten Computerprogramms FLOWERS erörtert, das u.a. zur Lösung eines breiten Spektrums statischer und dynamischer nichtlinearer Probleme anwendbar ist. Im Vordergrund steht dabei die Möglichkeit, den Ablauf des iterativen Rechenprozesses vom Bildschirmterminal aus während der Programmausführung interaktiv zu steuern.

On discute certains critères sur lesquels on a basé, à l'institut d'informatique de l'EPF Zurich, le développement du nouveau programme FLOWERS capable, entre autre, de traiter toute une gamme de problèmes statiques et dynamiques nonlinéair. Ce programme permet de suivre et de diriger le procès iteratif en temps réel pendant l'exécution du programme.

Some of the criteria which led to the development of the computer program FLOWERS at the Institute of Informatics of the Swiss Federal Institute of Technology in Zurich are discussed. This program can be used for several different kinds of static and dynamic nonlinear analysis based on the finite element method. It allows to control and steer the iteration process in real time from a terminal during program execution.

Einleitung

Im vorliegenden Aufsatz geht es um nichtlineare Finite-Element-Berechnungen aus der Sicht eines Informatikers, der sich die Aufgabe gestellt hat, dafür möglichst geeignete Hilfsmittel in Form von *Computerprogrammen* anderen Forschern, Studenten und Ingenieuren zur Verfügung zu stellen. Es geht also weder um spezifische Anwendungen noch um die mathematischen Modelle zur Lösung bestimmter Probleme, sondern um die zur Entwicklung dieser besonderen Art von Software wegleitenden Kriterien.

Diese stellen jedoch nicht nur Wunschvorstellungen dar, sondern bilden die Grundlagen, die zur Entwicklung des Computerprogramms FLOWERS am *Institut für Informatik* der *ETH Zürich* geführt haben. Allgemeine Informationen über dieses Programmsystem sind im «FLOWERS User's Manual» [1] (eine zweite Auflage wird im Frühling 1983 erscheinen) sowie in einem weiteren, in dieser selben Zeitschrift demnächst veröffentlichten Artikel [2] zu finden.

Hier wird man sich auf die *besondere Problematik der statischen und dynamischen nichtlinearen Analyse von Tragkonstruktionen* beschränken, wobei die ins Auge gefassten nichtlinearen Effekte wie folgt klassifiziert werden können:

1. *Geometrische* Nichtlinearitäten (Stabilität, grosse Verschiebungen und Rotationen, grosse Verzerrungen);
2. *Materialbedingte* Nichtlinearitäten (nichtlinear-elastische, elasto-plastische und sonstige inelastische Materialeigenschaften);
3. *Langzeitphänomene* (Kriechen);
4. *Kontaktprobleme;*
5. *Verschiebungsabhängigkeit* der äusseren Lasten;
6. Nichtlineare viskose Dämpfungseffekte.

Es ist heute bekannt, dass alle diese Effekte, einzeln oder kombiniert, bei praktisch allen Arten von Tragwerken mit der Methode der Finiten Elemente (FEM), wenigstens prinzipiell, numerisch erfasst werden können. Zudem findet die FEM je länger je mehr auch noch nichtkontinuumsmechanische Anwendungen. Die Entwicklung der dazu nötigen Software, vor allem wenn man sich als Ziele Benützerfreundlichkeit, Allgemeinheit, Anpassungsfähigkeit und Portabilität setzt, ist jedoch weder trivial noch als gelöstes Problem zu betrachten. Wie aus den folgenden Ausführungen klar werden dürfte, verlangen zudem nichtlineare FE-Berechnungen von den Programmbenützern viel Know-how und Erfahrung, so dass Ausbildungsfragen auch sehr im Vordergrund stehen. Daraus sowie aus der wachsenden Bedeutung von nichtlinearen Berechnungen in der *Praxis des Bau- und des Maschineningenieurwesens* ergeben sich die Gründe, die zum FLOWERS-Programm geführt haben. Einige Aspekte davon, beschränkt auf nichtlineare festigkeitstheoretische statische und dynamische Problemstellungen, werden im folgenden erörtert.

Über Finite Elemente

Auf die theoretischen Grundlagen der FEM kann hier nicht eingegangen werden (siehe z.B. [3] sowie unzählige andere Veröffentlichungen). Eine ihrer fundamentalen Eigenschaften, die für die Strukturierung und die Arbeitsweise von Computerprogrammen eine zentrale Rolle spielt, soll aber im folgenden erörtert werden.

Bei der Durchführung von FE-Berechnungen kann man zwischen *Verarbeitungsschritten,* die das *globale* System, und solchen, die jedes *einzelne* Element betreffen, klar unterscheiden. In der hiernach verwendeten Terminologie werden die ersten von den *«Systemroutinen»,* die zweiten von den *«Elementroutinen»* durchgeführt. Ein Programm wie FLOWERS besteht aus vielen System- und Elementroutinen, die eng zusammenarbeiten.

Aufgaben der *Systemroutinen* sind z.B. das Lesen der Systemdaten, die Aufstellung und Lösung der globalen Gleichungssysteme oder die Datenverwaltung auf Primär- und Sekundärspeicher. Aufgabe der *Elementroutinen* ist die Bereitstellung der numerischen Koeffizienten, die das Verhalten jedes einzelnen Elementes numerisch beschreiben und die von den Systemroutinen dann weiterverarbeitet werden.

Die Systemroutinen hängen vom Lösungsverfahren, nicht vom Problemtyp ab. So verlangen beispielsweise statische und dynamische Berechnungen auf globaler Systemebene eine völlig verschiedene Behandlung und damit andere Systemroutinen. Diese brauchen aber z.B. zwischen einer Rahmen- und einer Schalenberechnung nicht zu unterscheiden, da sie die lokalen Elementkoeffizienten, die sie einmal von Rahmenelementroutinen, einmal von Schalenelementroutinen übermittelt erhalten, nach identischen Prozeduren weiterverarbeiten. Dies macht es möglich, FE-Programme zu schreiben, die eine aus vielen Elementroutinen bestehende und beliebig erweiterbare Elementbibliothek besitzen und die somit völlig verschiedenartige Probleme behandeln können.

Die meisten grossen FE-Programme arbeiten nach diesem Prinzip, so auch das Programm FLOWERS, bei dem die scharfe Trennung zwischen Systemroutinen und Elementroutinen auch deswegen wichtig ist, weil FLOWERS ein *Forschungsinstrument* werden soll. Dies bedingt nämlich, dass neue Elementrou-

* Vgl. Schweizer Ingenieur und Architekt, Heft 51/52: 1117–1121, 1982; Heft 1/2: 2–7, Heft 4: 47–50, 1983

tinen für besondere Problemarten, Materialeigenschaften, Lasteneinwirkungen usw. von fremden Programmierern ohne Detailkenntnisse über die Arbeitsweise der Systemroutinen leicht hinzugefügt werden können.

Wichtig in diesem Zusammenhang sowie auch zum prinzipiellen Verständnis der Arbeitsweise des Programms FLOWERS ist die Frage nach den *Daten, die zwischen System- und Elementroutinen übertragen werden müssen.* Man betrachte einen vom Knotenverschiebungsparameter definierten Verschiebungszustand, bei dem das Gleichgewicht zwischen internen Spannungen, externen Lasten, Auflagerkräften sowie, bei den dynamischen Problemen, Massen- und Dämpfungskräften erfüllt ist. Für ein Element e (e = Elementindex) ist dieser Zustand von den lokalen Knotenverschiebungen a_j^e bestimmt, wobei der Index j (wie auch der unten eingeführte Index i) über alle elementeigenen Knotenverschiebungen geht. Dieser Zustand wird um bestimmte für das e-te Element mit Δa_j^e bezeichnete Knotenverschiebungsinkremente geändert. In dem neuen, aus $a_j^e + \Delta a_j^e$ definierten Zustand, gelten unter Vernachlässigung viskoser Dämpfungskräfte die Beziehungen:

$$q_i^e = -f_i^e(...a_j^e + \Delta a_j^e ...) - \sum_j m_{ij}^e \cdot (\ddot{a}_j^e + \Delta \ddot{a}_j^e) + p_j^e(...a_j^e + \Delta a_j^e ...)$$

$$dq_i^e = -\sum_j k_{Tij}^e(... a_j^e + \Delta a_j^e ...) \cdot da_j^e$$

wobei mit q_i^e die vom e-ten Element an die angeschlossenen Knoten übertragenen Knotenkräfte bezeichnet werden. Die f_i^e stellen den Anteil der Elementknotenkräfte dar, die mit den internen Elementspannungen das Gleichgewicht (näherungsweise) erfüllen und die in der Regel mit Hilfe virtueller Arbeitsprinzipien bestimmt werden. Im linearelastischen Fall gilt

$$f_i^e = -\sum_j k_{ij}^e \cdot (a_j^e + \Delta a_j^e)$$

wobei die k_{ij}^e-Koeffizienten die linearelastische Steifigkeitsmatrix bilden. Die Koeffizienten m_{ij}^e bilden die hier und im FLOWERS-Programm als zeit- und verschiebungsunabhängig angenommene Elementmassenmatrix. Die p_i^e sind Lastkoeffizienten, die den Einfluss von möglicherweise verschiebungsabhängigen, im Element wirkenden externen Lasten erfassen. Schliesslich bilden die k_{Tij}^e-Koeffizienten die sogenannte tangente Steifigkeitsmatrix, welche die aktuellen inkrementellen Beziehungen zwischen infinitesimalen Elementknotenkräften und Elementknotenverschiebungen angibt. Die konstanten m_{ij}^e-Koeffizienten sowie die vom momentanen Verschiebungszustand und oft von der Verzerrungsgeschichte abhängige f_{ij}^e, k_{Tij}^e- und p_i^e-Koeffizienten müssen von den Elementroutinen bestimmt und den Systemroutinen zur Weiterverarbeitung übermittelt werden. Alle Arten von nichtlinearen Effekten (z.B. auch bei Kontaktproblemen oder verschiebungsabhängigen Lasten) sind mit Hilfe entsprechender Elemente zu behandeln. Die Systemroutinen müssen ihrerseits die aktuellen Werte der a_j^e- und Δa_j^e-Koeffizienten den Elementroutinen übermitteln. Da in vielen Fällen die Verformungsgeschichte eine Rolle spielt, müssen zudem weitere elementspezifische Daten, deren Umfang bei grösseren Problemen sehr relevant sein kann, auch noch übermittelt werden. Diese Elementdaten werden von den Elementroutinen erzeugt und verarbeitet, müssen jedoch von den Systemroutinen verwaltet werden.

Eine weitere wichtige Frage betrifft die *möglichen Lösungsprozeduren auf Systemebene.* Mit anderen Worten: Was können die Systemroutinen mit den von den Elementroutinen erhaltenen f_i^e, p_i^e, k_{Tij}^e und m_{ij}^e-Koeffizienten anfangen? FLOWERS kann heute statische nichtlineare Analysen nach dem modifizierten *Newton-Raphson-Verfahren* (siehe z.B. [3]) sowie dynamische nichtlineare Analysen wahlweise nach zwei impliziten Algorithmen (*Newmark-und* ϱ-Methode siehe [4, 5]) sowie nach dem expliziten Verfahren der zentralen Differenzen (siehe z.B. [4]) durchführen. Andere, ähnlich arbeitende Algorithmen sind jedoch ohne weiteres implementierbar. Der dafür nötige Know-how dürfte allerdings grösser sein als bei der Implementation neuer Elementroutinen, da Eingriffe in die Systemroutinen nötig sind.

Über statische nichtlineare Analysen

Beim modifizierten *Newton-Raphson-Verfahren* werden die äusseren Lasten, der gegebenen Belastungsgeschichte folgend, in Lastinkremente angebracht. Ausgehend von einem bekannten Gleichgewichtszustand werden iterativ die Verschiebungsinkremente jeweils gesucht, die dem nach dem Lasteninkrement veränderten Gleichgewichtszustand entsprechen. Dazu sind folgende Schritte durchzuführen:

1. Vor dem ersten Schritt bzw. falls erwünscht: Neubildung der globalen tangenten Steifigkeitsmatrix für den Schrittanfangszustand aus den von den Elementroutinen übermittelten k_{Tij}^e-Koeffizienten und Dreieckszerlegung derselben.
2. Inkrement der äusseren Lasten
3. Bestimmung der dadurch verursachten Verschiebungsinkremente Δa_j^e durch Lösung eines entsprechenden globalen linearen Gleichungssystems. Dabei wird als Koeffizientenmatrix die zuletzt gebildete und dreieckszerlegte tangente Steifigkeitsmatrix verwendet.
4. Bestimmung der Elementknotenkräfte $q_i^e = -f_i^e + p_i^e$ durch die Elementroutinen für den Verschiebungszustand $a_j^e + \Delta a_j^e$
5. Bestimmung der Residualknotenkräfte. Diese entsprechen den äusseren Knotenlasten, die zur Erfüllung des Knotengleichgewichts notwendig wären.
6. Kontrolle, ob Gleichgewicht und damit Konvergenz erreicht ist. Dies ist der Fall, wenn die Residualknotenkräfte klein genug sind und die Verschiebungen sich vom vorhergehenden Iterationsschritt nicht wesentlich geändert haben.
7. Ist Konvergenz erreicht, so werden die Verschiebungen inkrementiert, womit ein weiterer Lastschritt, beginnend wahlweise vom Punkt 1. oder 2., behandelt werden kann. Ist Konvergenz nicht erreicht, dann:
8. Falls erwünscht: Neubildung der tangenten Steifigkeitsmatrix für den Verschiebungszustand $a_j^e + \Delta a_j^e$ und Dreieckszerlegung derselben.
9. Belastung des Systems mit den Residualknotenkräften als fiktive äussere Knotenlasten und Bestimmung entsprechender Verschiebungsinkremente, wofür wieder die Lösung eines globalen linearen Gleichungssystems notwendig ist. Als Koeffizientenmatrix wird die zuletzt gebildete und dreieckzerlegte tangente Steifigkeitsmatrix verwendet.
10. Inkrement der Verschiebungen und Fortsetzung des Iterationsprozesses ab Punkt 4.

Diese kurze Zusammenfassung soll vor allem zeigen, welche die für die Anwendung des modifizierten Newton-Raphson-Verfahrens *wichtigsten Steuerparameter* sind, nämlich:

1. Wahl geeigneter Lastinkremente
2. Wahl geeigneter Toleranzgrenzen
3. Wahl geeigneter Strategien für die Neubildung der tangenten Steifigkeitsmatrix.

Wenn es darum geht, eine Traglast zu bestimmen, was bei statischen nichtlinearen Problemen meistens der Fall ist, müssen die Lastinkremente in der Nähe der vorerst unbekannten Traglast, wo Gleichgewicht und damit Kon-

vergenz nicht mehr möglich sind, reduziert werden. Bei inelastischem Materialverhalten ist zudem die erhaltene Lösung eine Funktion der Belastungsgeschichte und damit auch der Grösse der Lastinkremente. Es ist daraus ersichtlich, dass die Wahl und die in vielen Fällen nötige laufende Anpassung der Lastinkremente ein heikles Problem darstellen können.

Eine weitere wichtige Frage ist die nach der *Spezifikation der Toleranzgrenzen,* aus denen das Programm Konvergenz feststellen kann. Zulässige Residualknotenkräfte bzw. zulässige Maximalwerte entsprechender, geeignet gewählter Vektornormen stellen natürliche Toleranzgrenzen dar. Die Residualknotenkräfte entsprechen nämlich einer zufälligen Streuung der ohnehin oft relativ willkürlich festgelegten äusseren Lasten dar. Weitere mögliche Toleranzgrenzen betreffen die relativen Verschiebungsinkremente zwischen zwei nachfolgenden Iterationsschritten.

Eine letzte wichtige Frage betrifft die Strategien zur *Neubildung der tangenten Steifigkeitsmatrix,* deren Koeffizienten das jeweils aktuelle Tragverhalten des Systems beschreiben. Allgemein kann man dazu nur sagen, dasss eine Neubildung vorteilhaft oder sogar nötig ist, wenn das Tragverhalten sich wesentlich geändert hat. Ist dies nicht der Fall, so soll man mit der alten Steifigkeitsmatrix weiterfahren, womit der Iterationsprozess zwar langsamer konvergiert, der für die Bildung und die Dreieckszerlegung der tangenten Steifigkeitsmatrix notwendige Rechenaufwand jedoch erspart bleibt.

Bei gewissen Computerprogrammen wurde versucht, die oben erwähnten Steuerungsparameter, oder wenigstens einige davon, durch automatische Prozeduren vom Programm selbst mehr oder weniger «optimal» festlegen zu lassen mit dem Ziel, den Programmbenützer von schwierigen Entscheidungen möglichst zu befreien. Wie aus den folgenden Ausführungen klar werden dürfte, wäre dies jedoch nicht FLOWERS-Philosophie.

Über dynamische nichtlineare Analysen

Eine dynamisch nichtlineare Analyse verlangt die direkte Lösung der nichtlinearen Bewegungsdifferentialgleichungen in *kleinen Zeitschritten,* ausgehend von einem *bekannten Anfangszustand.* Dafür kann man entweder implizite Algorithmen verwenden, die in jedem Zeitschritt die Lösung eines quasistatischen nichtlinearen Problems und folglich iterative Lösungen eines linearen Gleichungssystems sowie Konvergenzkontrollen verlangen, oder explizite Algorithmen, die ohne Lösung von Gleichungssystemen und ohne Konvergenzkontrollen zum Ziel führen (siehe z.B. [4]).

Auf Details kann hier nicht eingegangen werden. Es sei nur bemerkt, dass bei den *impliziten Algorithmen* sich sehr ähnliche Fragen wie bei der modifizierten Newton-Raphson-Methode stellen. Statt Lastinkremente hat man hier Zeitschritte, deren Länge aber nach wie vor mit grosser Sorgfalt zu wählen ist. Würde man eine lineare dynamische Analyse mit einem impliziten Algorithmus lösen, dann sollte man als Zeitschrittlänge einen Bruchteil der kürzesten Eigenperiode aller Eigenschwingungen wählen, die von den äusseren Lasten erregt werden. Dies hilft allerdings wenig bei nichtlinearen Problemen, wo von Eigenschwingungen nicht die Rede sein kann. Es wäre nur denkbar und in gewissen Fällen schon aus Vergleichsgründen angebracht, zuerst für ein ähnliches lineares System eine modale Analyse und erst dann eine nichtlineare Analyse durchzuführen. Bei den impliziten Algorithmen stellen sich zudem auch noch die Fragen nach der Festlegung von Toleranzgrenzen sowie nach der Strategie für die Neubildung der tangenten Steifigkeitsmatrix. Schliesslich ist zu erwähnen, dass meistens auch noch weitere Steuerparameter festzulegen sind, die u.U. ebenfalls einen wichtigen Einfluss auf die Endresultate haben können (z.B. β und γ bei der Newmarks-Methode [7], ϱ bei der ϱ-Methode [5], α bei der α-Methode [8] usw.).

Bei den *expliziten Algorithmen* entfallen die Fragen betreffend Toleranzen und Neubildung der tangenten Steifigkeitsmatrix. Um so heikler ist dafür die Frage nach der Zeitschrittlänge, weil die expliziten Algorithmen nur bedingt stabil sind. Bei linearen Anwendungen kann man zeigen, dass aus Stabilitätsgründen (nicht aus Genauigkeitsgründen) die Zeitschritte grössenordnungsmässig so lang wie die kürzeste Eigenperiode des diskreten Systems, d.h. in vielen Fällen ausserordentlich kurz sein müssen. Wieder hilft dies jedoch wenig bei nichtlinearen Anwendungen.

Ob implizite oder explizite Algorithmen vorteilhafter sind, hängt vom Problem ab (siehe z.B. [4]). Darauf kann hier nicht eingegangen werden, es ist jedoch zu bemerken, dass FLOWERS beide Möglichkeiten bietet. Es ist sogar möglich, die nichtlinearen Bewegungsdifferentialgleichungen zeitlich gestaffelt abwechslungsweise nach der einen oder nach der anderen Methode zu lösen. Nach dem gleichen Prinzip ist es möglich, zuerst eine statische Analyse durchzuführen (z.B. um den Einfluss ständiger Lasten zu berücksichtigen) und erst dann dynamische Lasten (z.B. infolge Erdbeben) anzubringen. Schliesslich ist noch die Möglichkeit zu erwähnen, sogenannte «Restart-Punkte» im Laufe des Iterationsprozesses zu setzen, die Zeitpunkten der Belastungsgeschichte entsprechen, in denen die Berechnung nachträglich wieder angefangen werden kann. Dies erlaubt auf einfache Art und Weise die Wiederholung der Berechnungen für die gleiche Zeitspanne der Belastungsgeschichte, jedoch mit anderen Schrittlängen, anderen Toleranzen, anderen Strategien oder sogar anderen Lösungsalgorithmen, was zur Beurteilung der Zuverlässigkeit der erhaltenen Resultate wesentlich beitragen kann.

Interaktive Steuerung des Rechenablaufes

Aus den bisherigen Ausführungen dürften schon einige wichtige Eigenschaften des Programms FLOWERS ersichtlich sein. Seine in bezug auf nichtlineare Berechnungen wohl markanteste Eigenschaft ist jedoch die Möglichkeit, den Rechenablauf vom *Bildschirmterminal* aus in *Echtzeit* zu verfolgen und zu steuern.

Dazu sind einige *Vorbereitungen* notwendig, die zuerst kurz beschrieben werden sollen. Eine nichtlineare Berechnung mit dem Programm FLOWERS erfolgt in *mehreren* Schritten. Diese entsprechen unabhängigen Programmen, sogenannten Moduln, die nacheinander ausgeführt werden. In einem ersten, für lineare und nichtlineare Probleme weitgehend identischen Schritt werden die globalen System- und Lastdaten (Knotenkoordinaten, Elementinzidenzen, Lastkoeffizienten usw.) sowie, je nach verwendetem Elementtyp, die dafür notwendigen elementspezifischen Daten (z.B. Materialbeiwerte) eingegeben. Auf diesen bestimmen die Elementroutinen der linearen sowie auch der meisten nichtlinearen Elementtypen die Matrixkoeffizienten, die zur Durchführung linearer statischer und dynamischer Analysen notwendig sind (lokale linear-elastische und geometrische Steifigkeits-, Massen-, Last- und Spannungsmatrizen). Bei den nichtlinearen Elementen werden zudem noch weitere, später benötigte Daten ermittelt und gespeichert. Es ist hier zu bemerken, dass es ohne weiteres möglich ist, lineare und nichtlineare Elemente zu mischen. Dies entspricht der Annahme, dass die nichtlinearen

Effekte auf gewissen Tragwerksteilen konzentriert sind (z.B. bei Kontaktproblemen), während sich die übrigen Tragswerksteile linear verhalten. Weil die linearen Elemente mit einem viel kleineren Aufwand als die nichtlinearen verbunden sind, kann damit die Rechenzeit u.U. merklich reduziert werden.

Nach diesem ersten Schritt ist es möglich und in vielen Fällen empfehlenswert, zuerst eine lineare Analyse durchzuführen. Aus den dabei in numerischer oder auch in graphischer Form erhaltenen Resultaten (statische Verschiebungen, Eigenfrequenzen und Eigenvektoren, Spannungsverteilungen usw.) kann man nämlich aufschlussreiche Informationen für die nachfolgende nichtlineare Analyse vor allem in bezug auf die Wahl der Zeitschritte und der Toleranzgrenzen erhalten.

Bevor die eigentliche nichtlineare Analyse anfangen kann, ist noch ein weiterer Vorbereitungsschritt notwendig, bei dem es um folgendes geht:

a. Spezifikation der erwünschten Analysistypen (statisch, dynamisch implizit nach verschiedenen Algorithmen, dynamisch explizit). Dies ist notwendig, weil die programminterne Datenorganisation von den möglichen Analysistypen abhängt.
b. Spezifikation der Belastungsgeschichte, bestehend aus einem oder mehreren mit unabhängigen Zeitfunktionen multiplizierten Lastfällen (bei statischen Problemen spielt die Zeit die Rolle eines Lastparameters).
c. Nur bei dynamischen Analysen: Spezifikation initialer Verschiebungen, Geschwindigkeiten und Beschleunigungen (diese können auch vom Programm berechnet werden).
d. Nur bei dynamischen Analysen: Spezifikation linearer Dämpfungskoeffizienten (nichtlineare Dämpfungseffekte sollen mittels entsprechender Elemente erfasst werden).
e. Spezifikation der Vektornormen (arithmetische Mittel, quadratische Mittel oder Maximalwert) für bestimmte Residualknotenkräfte oder relative Verschiebungskomponenten, welche für Konvergenzkontrollen zu verwenden sind. Die dazugehörigen numerischen Toleranzwerte werden allerdings später spezifiziert.
f. Spezifikation der system- oder elementbezogenen Variablen, deren aktuelle numerische Werte im Laufe des nachfolgenden Iterationsprozesses auf dem Bildschirm erscheinen können (Knotenverschiebungen, Elementknotenkräfte, Elementspannungen, Residualknotenkräfte, Knotenlasten, Auflagereaktionen usw.). Der Verlauf dieser Variablen kann nachträglich von einem speziellen Programmodul des FLOWERS-Systems auch graphisch dargestellt werden.
g. Spezifikation der system- oder elementbezogenen Variablen, deren aktuelle numerische Werte auf dem Zeilendrucker ausgegeben werden sollen.

Bei allen bisher erwähnten Vorbereitungsschritten ist eine direkte Benützerprogramm-Interaktion weder möglich noch, unserer Ansicht nach, notwendig (selbstverständlich können und sollen aber auch beim FLOWERS-Programm interaktive Vorlaufprogramme zur Generierung der Elementmasche verwendet werden). Ganz anders liegen die Verhältnisse bei der eigentlichen nichtlinearen Analyse, deren Erfolg, wie man gesehen hat, von der Wahl mehrerer, im voraus oft schwer bestimmbarer Steuerungsparameter abhängt.

Nach den erwähnten Vorbereitungsschritten kommt man zur eigentlichen nichtlinearen Analyse, die vom Programmbenützer mit Hilfe einer gewöhnlichen alphanumerischen Bildschirmkonsole Schritt für Schritt verfolgt und gesteuert werden kann. Während der Berechnung schreibt das Programm auf der Bildschirmkonsole (mit 24 Zeilen zu je 80 Zeichen) laufend Informationen über den Fortgang des Iterationsprozesses, bis die Programmausführung, sei es zu vorgegebenen Zeitpunkten, sei es, indem der Programmbenützer einfach eine Taste drückt, am Ende eines Iterationsschrittes unterbrochen wird. Diese Unterbrechungen können sowohl zur genauen Prüfung der auf dem Bildschirm erscheinenden Daten (diese ändern sich nämlich im Laufe der Berechnung, vor allem bei kleinen Problemen, manchmal blitzartig) als auch zur Spezifikation folgender Steuerungsparameter, wofür übersichtliche «Menues» verwendet werden:

1. Änderung des Analysistyps (statisch-dynamisch, implizit-explizit).
2. Festlegung der Anzahl und Länge der zu berechnenden Zeitschritte bis zur nächsten automatischen Unterbrechung.
3. Spezifizierung bzw. Änderung der Toleranzwerte für die vorher spezifizierten Konvergenzkriterien (siehe oben Punkt e) oder Aktivierung/Desaktivierung derselben. Auf dem Bildschirm werden dann mit Hilfe von Balkendiagrammen entsprechende prozentuelle Werte laufend gezeigt (› 100% = Konvergenz nicht erfüllt, ‹ 100% = Konvergenz erfüllt), womit der Programmbenutzer das Konvergenzverhalten seines Systems mit einem Blick beurteilen kann.
4. Spezifizierung der Strategie für die Neubildung der tangenten Steifigkeitsmatrix (z.B. alle «n» Zeitschritte oder in jedem Zeitschritt nach «n» erfolglosen Iterationen).
5. Spezifizierung der system- oder elementbezogenen Variablen (siehe oben Punkt f), deren am Ende der drei letzten Iterationen berechneten Werte auf dem Bildschirm erscheinen sollen (höchstens 12 Variablen; es wäre allerdings schwierig, den zeitlichen Verlauf von mehr als 12 Variablen in Echtzeit zu verfolgen!).
6. Einmalige, momentane Ausgabe aller nach Punkt f definierten system- und elementbezogenen Variablen.
7. Einmalige, momentane Ausgabe auf dem Bildschirm oder auf dem Zeilendrucker von elementspezifischen Detailinformationen für bestimmte Elemente. Diese Art von Ausgabe ist nicht Sache der Systemroutinen, sondern der Elementroutinen, die dafür speziell angerufen werden. Der Programmierer von Elementroutinen, der sonst an strikte Konventionen gebunden ist, hat damit die Möglichkeit, nach eigener Darstellungsart aktuelle elementspezifische Informationen dem Programmbenützer mitzuteilen.
8. Spezifizierung sonstiger Daten, die auf dem Zeilendrucker ausgegeben werden sollen.
9. Rettung aller Knotenverschiebungen der momentanen Systemkonfiguration. Diese können nachträglich von einem anderen Programmodul des FLOWERS-Systems zur graphischen Darstellung der verformten Systemkonfiguration verwendet werden. Es wäre denkbar, aus mehreren solcher Bilder eine Art Trickfilm über die Verformungsgeschichte des Systems herzustellen.
10. Spezifizierung bzw. Änderung bestimmter, je nach Elementtyp speziell definierter Parameter, die von den Elementroutinen dann verarbeitet werden. Damit besteht nicht nur die unter Punkt 7. erwähnte Möglichkeit, dass die Elementroutinen mit dem Programmbenützer direkt kommunizieren, sondern der umgekehrte Weg ist auch noch offen.
11. Definition des aktuellen Zeitpunktes der Belastungsgeschichte als möglicher «Restart-Punkt» bzw. Wiederholung der Berechnungen beginnend von einem früher definierten Restart-Punkt.

Schliesslich hat der Programmbenützer nach jeder Unterbrechung die Möglichkeit, entweder die Berechnung fortzusetzen oder die Programmausführung zu beenden. In diesem Fall hat er Gelegenheit, gewisse Zwischenresultate, wie oben erwähnt, in graphischer Form auszugeben oder die gedruckten Ausgaberesultate in Ruhe zu studieren und später in einem weiteren Programmlauf die Berechnungen von einem der definierten Restart-Punkte fortzusetzen.

Bei einer derartigen interaktiven Arbeitsweise stellt sich sofort die Frage nach den *unterschiedlichen Arbeitsgeschwindigkeiten des Computers und des vor dem Bildschirm sitzenden Programmbenützers.* Unproblematisch ist der Fall, bei dem der Computer zu schnell rechnet, da die Programmausführung jederzeit mit einem Tastendruck unterbrochen werden kann, um den Bildschirminhalt in Ruhe anzuschauen. Bei grösseren Problemen und vor allem bei dynamischen Analysen, die sehr viele Zeitschritte verlangen, ist es aber durchaus denkbar, dass die Programmausführung unerträglich langsam wird. Dies hängt selbstverständlich auch von der Leistungsfähigkeit und von der sonstigen Auslastung des verwendeten Computers ab (das Programm wird auf der DEC-10-Anlage des *Zentrums für Interaktives Rechnen* der ETHZ entwickelt). Weil das Programm sich noch in der Testphase befindet, sind unsere diesbezüglichen Erfahrungen bis jetzt zwar ermutigend, jedoch noch nicht ganz aussagekräftig. Dazu aber zwei Bemerkungen. Die Betriebssysteme aller moderner Computer erlauben die interaktive Arbeitsweise zu simulieren, in dem die vom Programm erwarteten Eingabedaten nicht von der Benützerkonsole, sondern von einer früher vorbereiteten Datei gelesen werden. Eine Programmausführung, z.B. während der Nacht, ohne physische Präsenz des Programmbenützers und nur mit Zeilendruckerausgabe ist deswegen möglich.

Die zweite Bemerkung betrifft die *Art der Modellbildung,* wenn es darum geht, das nichtlineare Verhalten bestimmter Tragkonstruktionen unter extremen Lasteinwirkungen abzuklären. Zu oft werden in der Praxis nichtlineare Berechnungen mit einem Rechenaufwand durchgeführt, der in keinem Verhältnis mit den Unsicherheiten der zu Grunde gelegten last- und materialtechnischen Annahmen steht. Weil es dabei meistens um das grundsätzliche, globale Tragverhalten und folglich eigentlich nur um grobe quantitative Aussagen geht, sollte eher die *Überblickbarkeit* des verwendeten mathematischen Modells als dessen (oft vorgetäuschte) Genauigkeit im Vordergrund stehen. In vielen Fällen können nämlich schon relativ einfache Modelle zum Ziel führen, deren Wahl allerdings ein tiefgreifendes Verständnis der prinzipiellen Tragwirkung der untersuchten Konstruktion verlangt. Das hier beschriebene Programmsystem, das seinen Benützer zwingt, jeden einzelnen Rechenschritt genau zu verfolgen, dürfte diesbezüglich einen wesentlichen Beitrag leisten können.

Schlussbemerkungen

Aus diesen Ausführungen sollten viele der vom Programm FLOWERS gebotenen Möglichkeiten ersichtlich sein. *Einige Fragen bleiben allerdings noch offen.* Die erste betrifft die *nichtlinearen Elemente,* die dem Programmbenützer zur Verfügung stehen, und damit die konkreten Probleme, die sich lösen lassen. Eine Reihe von nichtlinearen Elementroutinen für Rahmen-, Fachwerk-, Kontakt-, Platten- und Schalenelemente wird z.Z. entwickelt. Andere werden folgen, worauf an einer anderen Stelle eingegangen werden soll. Hier ist vor allem zu betonen, dass FLOWERS ein beliebig erweiterbares Forschungsinstrument werden soll, in dem viele, vom Forscher mit sehr verschiedenen Zielsetzungen entwickelte Elemente aufgenommen werden können. Dazu ist noch zu bemerken, dass dies auch für manche nichtkontinuumsmechanischen, stationären und instationären Feldprobleme, die sich nach der FE-Methode behandeln lassen, zutrifft.

Literatur

[1] *Anderheggen, E., Bazzi, G., Elmer, H., Friedrich, T., Maag, H., Theiler, J.:* «FLOWERS User's Manual». Institut für Informatik, ETHZ, 1. Auflage, Juni 1981

[2] *Anderheggen, E.:* «FLOWERS: ein neues Computerprogramm für Finite-Element-Berechnungen», Schweizer Ingenieur und Architekt, Heft 6, 1983

[3] *Zienkiewicz, O.C.:* «The Finite Element Method». McGraw-Hill, 1977

[4] *Bathe, K.:* «Finite Element Procedures in Engineering Analysis». Prentice Hall, 1982

[5] *Bazzi, G., Anderheggen, E.:* «The ϱ-Family of Algorithms for Time-Step Integration with Improved Numerical Dissipation». J. of Earthquake Engineering and Struktural Dynamics», Vol. 10, 537-550 (1982)

[6] *Bazzi, G.:* «Ein Beitrag zur Dynamischen Berechnung nichlinearer Tragwerke». Diss. No. 7080, ETH Zürich, 1982

[7] *Bathe, K., Wilson, E.L.:* «Numerical Methods in Finite Element Analysis». Prentice Hall, 1976

[8] *Hilber, H.M., Hughes, T.J.R., Taylor, R.L.:* «Improved Numerical Dissipation for Time Integration Algorithms in Struktural Dynamics». J. of Earthquake Engineering and Struktural Dynamics», Vol. 5, 283-292 (1977).

Eine weitere Frage betrifft die *Verfügbarkeit des Programms* innerhalb und ausserhalb der ETHZ. Für Lehrzwecke wird FLOWERS bzw. dessen lineare Programmteile seit zwei Jahren zu Übungszwecken von den Bauingenieurstudenten des 7. Semesters verwendet. Es ist zudem geplant, das Programm innerhalb einer an der Abteilung für Informatik neu einzuführenden Lehrveranstaltung einzusetzen. Für Ingenieure aus der Praxis und für Assistenten und Forscher der ETHZ wird im Laufe des Sommersemesters 1983 der Fortbildungskurs «Lineare und nichtlineare Finite-Element-Methoden» durchgeführt. Dabei werden die Kursteilnehmer neben der Behandlung theoretischer Fragen die Möglichkeit haben, das Programm FLOWERS von Bildschirmterminals aus im Time-Sharing-Betrieb bei der Lösung von Übungsaufgaben zu verwenden.

Adresse des Verfassers: Prof. Dr. *E. Anderheggen,* Institut für Informatik, ETH-Hönggerberg, 8093 Zürich.