Zeitschrift: IABSE reports of the working commissions = Rapports des commissions de travail AIPC = IVBH Berichte der Arbeitskommissionen

Band: 31 (1978)

Rubrik: Session I: Technical implications

I.

MAIN SESSIONS

I Session — Technical Implications

Session I — Implications techniques

I. Sitzung — Technische Zusammenhänge

Leere Seite
Blank page
Page vide

IABSE
AIPC
IVBH

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Effective Use of Structural Computer Programs

Utilisation efficace des programmes de structures

Wirkungsvoller Einsatz baustatischer Computer-Programme

D.D. PFAFFINGER

Dr. sc. techn, Senior Consultant
FIDES Trust Company
Zürich, Switzerland

Summary

The use of structural engineering programs can raise difficulties. Some of the typical problems and their solutions are outlined on an illustrative example. On this basis general aspects of effective use of structural programs are discussed, leading to a list of requirements on the software as well as on the user. In the conclusions some suggestions are made to further improve the effectiveness of use of structural programs in the future.

Résumé

L'utilisation de programmes pour le calcul de structures peut conduire à des difficultés. Quelques problèmes typiques ainsi que leur solution sont illustrés par un exemple. On discute sur cette base les aspects généraux de l'utilisation efficace des programmes de structures. Ceci conduit à une série de requêtes concernant autant le logiciel que les utilisateurs. On conclut par quelques suggestions permettant dans le futur d'utiliser les programmes de structures avec encore plus d'efficacité.

Zusammenfassung

Der Einsatz baustatischer Computer-Programme kann mit Schwierigkeiten verbuden sein. Anhand eines Beispiels werden einige der typischen Probleme wie auch ihre Lösung aufgezeigt. Vor diesem Hintergrund werden allgemeine Gesichtspunkte des wirkungsvollen Einsatzes solcher Programme erörtert, welche zu einer Reihe von Anforderungen an Programme wie auch Benutzer führen. Die Schlussfolgerungen enthalten einige Vorschläge, um baustatische Programme in der Zukunft noch wirkungsvoller einsetzen zu können.

1. INTRODUCTION

Norbert Wiener once estimated that of all problems worked on a computer only 10% were adequately formulated, because in 90% of the cases the solutions had not been conceptually worked out in the mind before coding them for the machine. For structural engineering problems this percentage may not be as pessimistic. The general observation Wiener's, however, remains valid. In structural engineering the solutions of many problems require comprehensive understanding of the problem, knowledge of advanced solution methods and frequently also extensive calculations. Examples for this situation are highly redundant beam structures, plate and shell problems or dynamic analyses. Before the advent of the electronic computer the engineer was hence usually forced to simplify his problem considerably to be able to solve it. The quality of his analysis depended largely on his ability to set up an analysis model which preserved the characteristic properties of the real structure and also to interpret the results obtained from the model with respect to their meaning for the real structure. These tasks had conceptually to be solved by the human brain and not by any calculating device. Today, inspite of the abundance of computer power and the existence of numerous structural engineering programs, this situation basically has not changed.

One frequent cause of problems with electronic calculations stems from this basic misunderstanding: the most comprehensive computer programs and the most powerful machines do not conceptually solve a problem. It is still the engineer who works out the solution conceptually in his mind but uses the computer to do the numerical operations. The engineer's ability to set up the analysis model and to interpret the results is hence still a prerequisite to a meaningful solution as it was before. He may now, however, set up a very complex and comprehensive model and analyse it by means of already coded advanced mathematical procedures without knowing all their details. Other causes of difficulties with computer solutions are: problems in applying the methods of a program properly to the chosen model while observing their limitations and restrictions; bad results due to wrong input data or program errors; time delays due to time consuming input preparation, hardware failures etc.; problems with the evaluation and interpretation of voluminous results.

In order to use the modern computational means effectively, these difficulties have to be reduced. To do so, a number of requirements on modern structural engineering computer programs can be set up ([3], [5], [10], [11], [19], [21]). The user of such programs, on the other hand, has to acquire new skills and attitudes which enable him to use these tools effectively ([6], [12], [13], [16], [17], [20]). While having in mind larger and complex analyses rather than simple routine calculations it is the purpose of this paper, starting with an illustrative example, to investigate these two sets of requirements and to show how to use structural engineering computer programs effectively.

2. CASE STUDY

Fig. 1 shows the building site of the new office building of the Bayerische Hypotheken- und Wechsel-Bank in Munich, comprising a conventional office building and a high-rise structure supported by four towers. This latter structure has a total hight of 115 m and consists of 33 stories. In Fig. 2 a simplified groundplan of the arrangement of the floors between the towers is shown. It was planned to transmit all story loads to the towers along one story only by means of a

joint of unreinforced high-strength concrete. Fig. 3 is a section through tower D (Fig. 2) showing the inner core (tower), the outer core (story) and the joint. Circular post-tensioning provided the required compression forces. The extensive numerical analyses posed numerous problems typical for electronic computations. Their solutions hence serve well as an illustration to the subject of this paper.

Fig. 1 View of building site

First linear elastic analyses of the towers under story loads and wind loads had to be performed. One basic modelling problem was the simple representation of the concrete joint. To solve this problem preliminary studies were made comparing an axisymmetric three-dimensional model, a model with excentrically connected membrane elements, a general three-dimensional model and a model with flat shell elements for the joint (programs ANSYS, NASTRAN, ROST, [2], [7], [8], [9], [18]). Evaluation of the results (Fig. 4) showed that the joint could well be represented by the simple membrane model.

For one of the towers it was decided to perform a limit load analysis. This created serious methodical and modelling problems because none of the available nonlinear programs possessed all of the required facilities. A solution was obtained by deciding on an approximate solution based on the lower bound

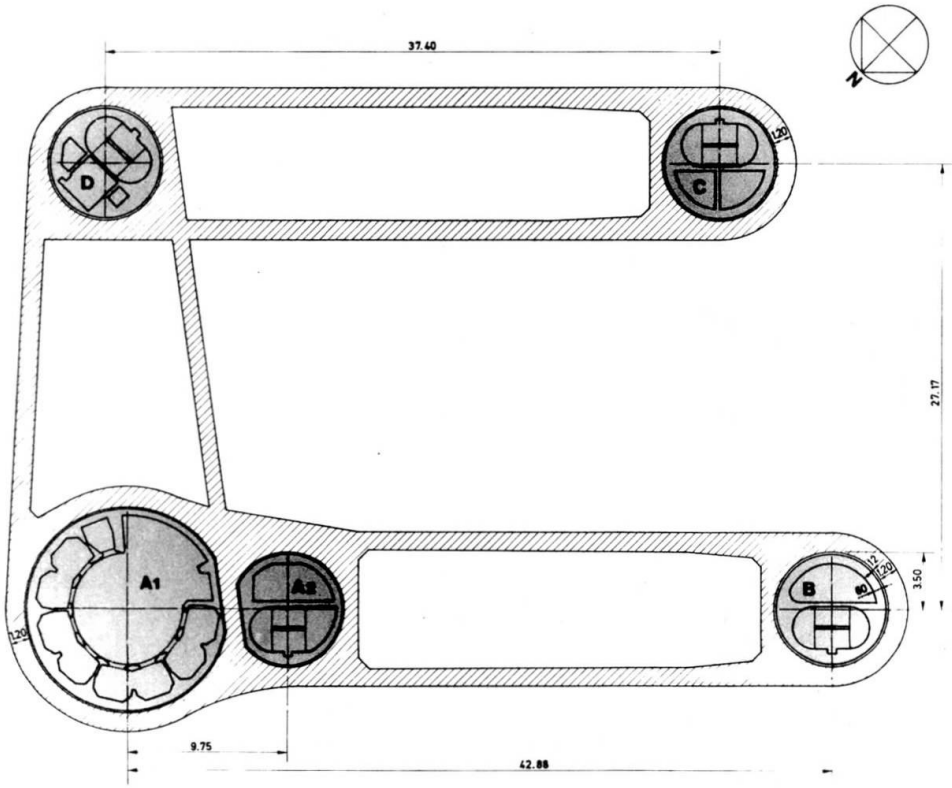


Fig. 2 Simplified ground-plan of structure

theorem of limit analysis and using an elastic anisotropic membrane material model ([1], [14], [15]). Extensive preliminary numerical tests were performed to define the material constants of the limit load model and to verify the solution algorithm.

A number of familiar problems arose for the different analyses, which were performed with MSC/NASTRAN. The problem of checking the input data could be solved by using mesh generators and plotting (Fig. 5) as well as by printing specific tables and matrices. The voluminous numerical results were represented graphically as for instance in Fig. 6. Numerous hand calculations and checks were done to verify the results. A specific problem arose for parts of the structure which were modelled by three-dimensional elements. The interpretation of the calculated stress field with respect to the dimensioning could not be solved satisfactorily. This problem stems from the lack of an adequate dimensioning theory for three-dimensional reinforced concrete structures.

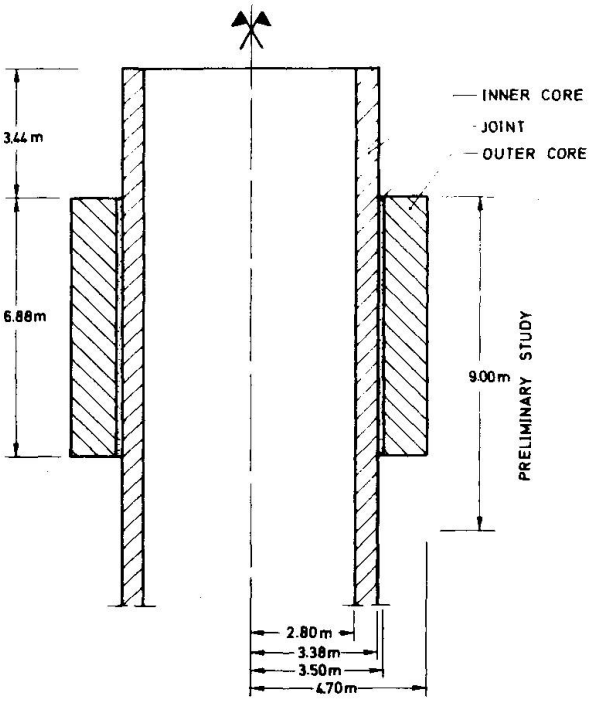


Fig. 3 Cross-section of tower D

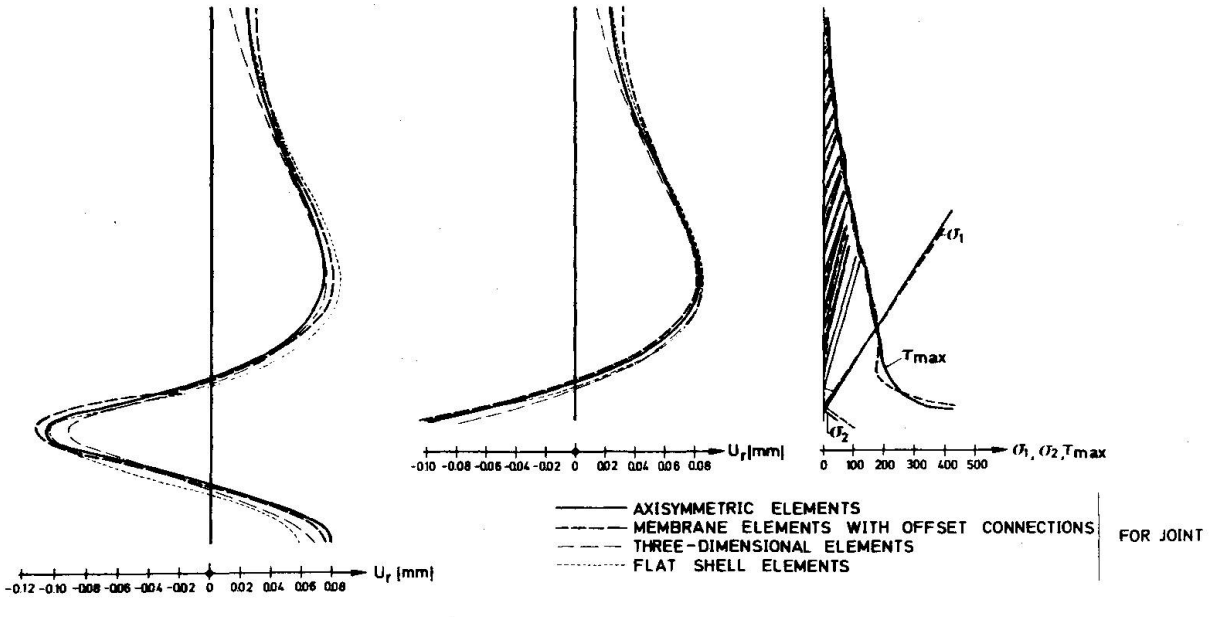
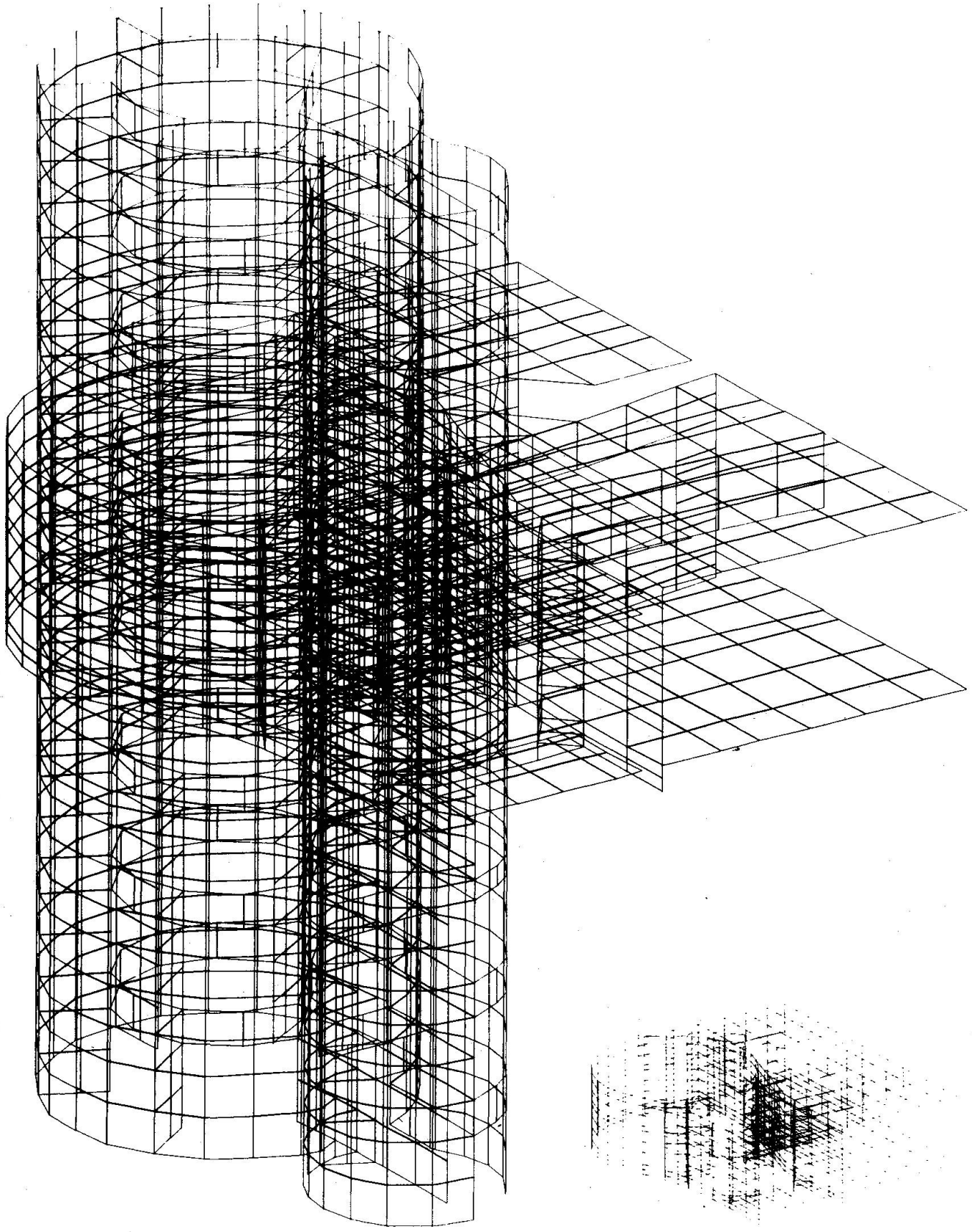


Fig. 4 Results for different models for the joint under axisymmetric loading

The performed analyses showed that the structure and especially the joint were statically sound under the service loads as well as under limit load conditions. In spite of this fact and due to other reasons it was decided, however, to use reinforced concrete also for the joint.



Full mesh

Detail

Fig. 5 Mesh of towers A1 and A2

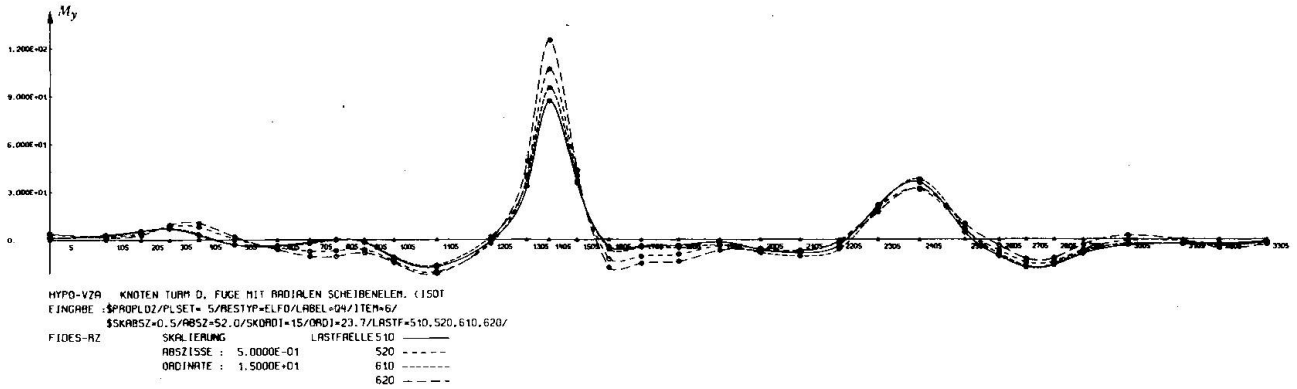


Fig. 6 Plot of bending moments M_y along vertical section of tower

3. REQUIREMENTS ON STRUCTURAL ENGINEERING PROGRAMS

The above case study illustrated the fact that many computer solutions are by no means off-the-shelf solutions and usually require consideration of several aspects of the problem. Fig. 7 shows the different phases of a typical structural analysis and their interaction. The most demanding phases are the setting up of the conceptual analysis model and the interpretation of the results. In the first one, the solution concept is worked out (statement of the problem, simplifications, required algorithms etc.) while considering all additional conditions such as available programs, available computer capacity, required accuracy, time frame and budget. From the interpretation of the results consequences for the real structure as well as for all phases of the solution are derived. The phase of preparing the numerical model is in close interaction with the conceptual model. Solving the problem for the numerical model and displaying the results allows the derivation of conclusions. Structural engineering computer programs can provide the means for the user, to accomplish the tasks of some of these phases very effectively. In the following hence the most important requirements on structural programs to permit effective use are discussed.

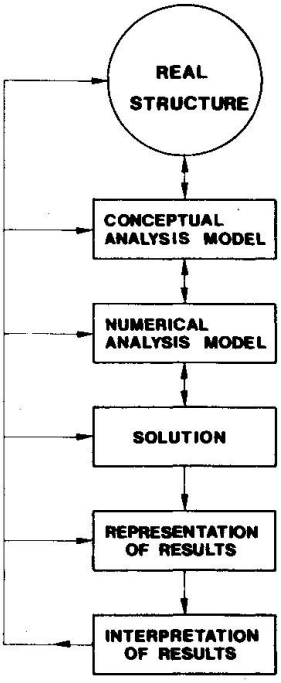


Fig. 7 Phases of solution

3.1 Easy data preparation and verification

This requirement concerns the phase of setting up the numerical model. It means, that a program should offer data generators, extensive facilities for graphical displays and print options for important lists such as for instance element connection tables in finite element analyses. The program also has to perform comprehensive formal tests on the input data and has to furnish meaningful error messages. Due to the importance of easy preparation of the numerical data several preprocessor systems exclusively for this task for finite element models are under development or are already available. These systems allow the definition of the model in a unified input language or by means of interactive graphics,

they perform extensive checks and produce the formated input cards for several FE-programs at the user's option.

3.2 Efficient and verified numerical procedures

The fast development of the computing facilities has lead to very active development of new numerical methods. Big progress with respect to efficiency and reliability has been made over the past years in such fundamental tasks as decomposition, forward-backward substitution, eigenvalue extraction or integration of the coupled equations of motion. It is a requirement that a modern structural engineering program should have incorporated advanced numerical techniques. The numerical procedures have to be tested and verified. In the case of finite element programs new elements have to pass a series of tests before they are made available to the user. The requirement of verified procedures also means, that the program can print out fundamental characteristic magnitudes for the procedure such as residual forces, number of negative terms on the diagonal of the triangular factor, error bounds for eigenvalues and so on.

3.3 Generation and representation of results

It is essential, that a program can produce all the results necessary for the interpretation phase. Depending on the problem this can mean displacements, velocities, accelerations, reactions, forces and stresses, strains, strain energies, corner forces etc. It is also a basic requirement that the usually voluminous results of a structural engineering program can be displayed in a condensed way. Here the graphical means play an important role. It is also required, that results can be demanded in a selective way and that by means of a cheap restart more results can be obtained. There is a trend to separate the task of representing the results from the main program by means of postprocessors.

3.4 Intermediate results and user interaction

For programs for more advanced applications, such as nonlinear applications or dynamics, it is required that intermediate results can be produced at the user's option. The user has to be in the position to print intermediate matrices such as stiffness or damping matrices and also intermediate results as for instance after every load step in nonlinear analyses. These intermediate results can be essential also during the phase of finding the conceptual model if computer runs are made to verify the assumptions. User interaction is usually required to an increasing degree in large and/or complex problems. This includes capabilities of the program to restart after a machine failure by using saved intermediate results and also to restart for more load cases, eigenvalues, load or time steps.

3.5 Documentation

A basic requirement for all computer programs is thorough documentation. Here first of all the user's documentation has to comprise all information necessary to use the program. The limits of applicability have to be clearly stated. For programs for advanced applications also documentation of the methods and mathematical procedures, their applicability, their implementation and verification is required. It also means documentation of a collection of examples.

3.6 Maintenance, development, support

Computer programs have to be maintained. Error corrections have to be made and improvements have to be incorporated. Programs should be kept up to date with new technologies and new mathematical procedures. One final but basic requirement is the support of a program and its users. This means especially professional support of the user to apply the capabilities of the program adequately and assistance in the case of difficulties. This support can be provided by the developer or by specially trained professional people. For large structural engineering programs with a wide range of capabilities the quality of the support can become the decisive factor for effective or ineffective use.

4. REQUIREMENTS ON THE USER

Many technological achievements like the car or the telephone are used to-day without understanding the details of their functioning. It is the user, however, who bears the final responsibility of using these means adequately. To do so, certain new skills are required. In a comparable sense this situation exists also for usage of computer programs. In Fig. 8 the degree of automation for the different phases of a typical computer solution of a structural problem is qualitatively sketched. It is seen that the phases of setting up the conceptual model and of interpreting the results can very little be automated and are thus left to the user. Here as well as in the other phases only a knowledgable user can master his tasks effectively. In the following hence the most important skills and requirements on a user of structural programs are discussed.

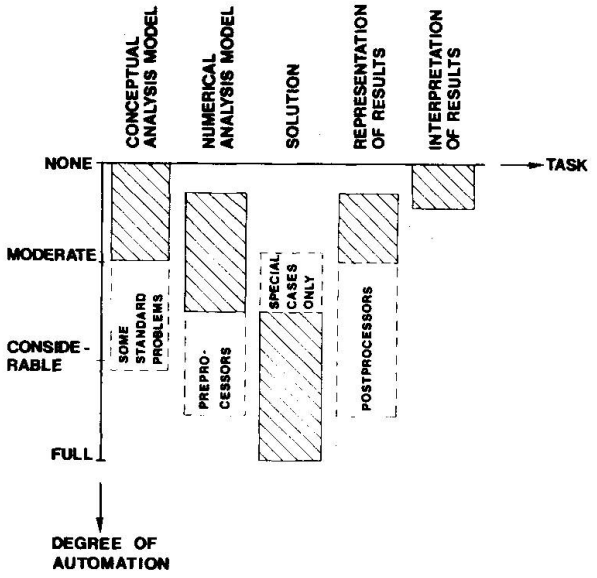


Fig. 8 Degree of automation

4.1 Understanding of the problem

In order to set up the conceptual model and to interpret the results effectively, the user has to understand his problem from an engineering point of view. He has to comprehend the basic behavior of the structure and the type of analysis that is required. The understanding of his problem enables the user to find the appropriate model, to perform plausibility checks and to distinguish unexpected results from erroneous results. This engineering insight into a problem can be gained by experience, by successive studies of the problem with increasing complexity and to a certain extent by formal training.

4.2 Understanding of the solution methods

It is seen from Fig. 8, that the solution for the numerical model is usually

highly automated. Inspite of this, all solution algorithms have peculiarities and limitations which have to be known to the user in order to use them effectively. It is thus required that the user has a working knowledge of these methods from the applications point of view. This means understanding of the basic assumptions, properties and limitations of the methods. As many engineering problems lead to the same mathematical expressions the knowledge of their numerical solution methods also sometimes allows the solution of problems by analogy.

4.3 Knowledge of the computational facilities

Before setting up a major analysis model the user is required to check on the computational facilities. The availability and capacity of the hardware has to be investigated. The available programs for the calculations in mind have to be evaluated and a selection has to be made. The different restrictions from hardware, software, budget, time frame etc. have to be considered at the very beginning of a major analysis and the solution concept has to be defined accordingly. Here close cooperation with specialists in different fields may be required.

4.4 Critical attitude

The engineer, who is finally responsible for the calculations, is required to be critical towards the calculated results. He has to check the adequacy of the assumptions for the model, the solution path and the results. Only verified results can be used as a basis for decisions.

5. CONCLUSIONS

The highly competitive market of structural engineering computer programs has lead to substantial improvements in the reliability, capabilities and easiness of use of such programs. Yet there still remains much to be done. It is proposed, that the evaluation and the quality control of programs is done to a still greater extent by the engineering comunity. This could be done by the professional societies which then would also organize the exchange of such information. It is also proposed that high emphasis is put on the development of general purpose data preparation and result evaluation programs. Here the capabilities of minicomputers for interactive work might prove to be of great value.

Many difficulties with electronic computations stem from the inadequate preparation of the user to his tasks. Here much remains to be done concerning the training of the engineers. More emphasis should be given to courses which develop the basic understanding of structural behaviour and also to courses on the principles of modern numerical methods. Inspite of all sophistication in computers and programs it is and will still be the human mind which conceptually has to solve the problems before going on a computer. The critical and knowledgable engineer, however, will be able to use these modern computational means to set up analysis models closer and closer to reality which will also permit him to solve his main task - to design structures - more effectively.

Acknowledgements

Most of the numerical analysis cited here was done by FIDES Rechenzentrum GmbH,

München under a contract with Dyckerhoff + Widmann AG and Held + Franke AG, both Munich, with the author as consultant for the electronic computations. The kind permission of Dyckerhoff + Widmann AG, and Held + Franke AG to cite this project and to show some results of the analysis is gratefully acknowledged.

References

1. Argyris, J.H., Faust, G., Willam, K.J., Limit Load Analysis of Thick-Walled Concrete Structures - a Finite Element Approach to Fracture. Computer Methods in Applied Mechanics and Engineering, 8. 1976.

2. De Salvo, G., Swanson, J., ANSYS Engineering Analysis System User's Manual. Swanson Analysis Systems, Inc., Elizabeth, PA. 1975.

3. Fenves, S.J., Methods Appropriate to an Engineering Software Center. ASCE Annual National and Environmental Engineering Meeting, New York. 1973.

4. Fredriksson, B., Mackerle, J., Structural Mechanics Finite Element Computer Programs, Parts I and II. Linköping Institute of Technology, Dep. of Mechanical Engineering. 1975.

5. Gallagher, R.H., Trends and Directions in the Applications of Numerical Analysis. Numerical and Computer Methods in Structural Mechanics. Fenves, S., Perrone, N., Robinson, A., Schnobrich, W., Editors. Academic Press, New York and London. 1973.

6. Huré, D., Morysse, M., Comparative Methods for Analysis of Piping Systems Subjected to Seismic Motion. Nuclear Engineering and Design, 38. 1976.

7. Kohnke, P., ANSYS Engineering Analysis System Theoretical Manual. Swanson Analysis, Inc., Elizabeth PA. 1977.

8. MacNeal, R.H., Editor, The NASTRAN Theoretical Manual. NASA SP-221 (01). 1972.

9. McCormick, C.W., Editor, The MSC/NASTRAN User's Manual. The MacNeal-Schwendler Corp., Los Angeles. 1977.

10. Medearis, K., An Investigation of the Feasibility of Establishing a National Civil Engineering Software Center. ASCE, New York. 1973.

11. Pauli, K., Programmwünsche-Weisse Flecken, Hefte 1,2.3. Hauptverband der Deutschen Bauindustrie, Wiesbaden. 1975.

12. Pfaffinger D., Computer-Graphik als Hilfsmittel des Bauingenieurs. Schweizerische Bauzeitung, 92. Jahrgang, 3. 1974.

13. Pfaffinger D., Comparative Seismic Analysis. Proc. 3rd NASTRAN European User's Conference, Munich. 1976.

14. Pfaffinger D., Thielen G., Limit Load Analysis of a Concrete Structure. Proc. 5th NASTRAN European User's Conference, Munich. 1978.

15. Phillips, D., Zienkiewicz, O., Finite Element Non-Linear Analysis of Concrete Structures. Proc. Instn. Civ. Engineers, Part 2, Vol. 61. 1976.

16. Polónyi, S., Reyer, E., Zuverlässigkeitsbetrachtungen und Kontrollmöglichkeiten (Prüfung) zu praktischen Berechnungen mit der Finiten-Element-Methode. Die Bautechnik, 11. 1975.

17. Robinson, J., A Single Element Test. Computer Methods in Applied Mechanics and Engineering, 7. 1976.

18. ROST-Handbuch. FIDES Rechenzentrum, Publ.-Nr. 30, Zürich. 1970.

19. Stillman, R.B., Trends in Computer Software. Structural Mechanics Computer Programs, Pilkey, W., Saczalski, K., Schaeffer, H., Editors. University Press of Virginia, Charlottesville. 1974.

20. Svalbonas, V., Transient Dynamic and Inelastic Analysis of Shells of Revolution - a Survey of Programs. Nuclear Engineering and Design, 37. 1976.

21. The State-of-the-Art of Computer Graphics. Applications in Structural Engineering. Report of the Task Committee on Computer Graphics of the Committee on Electronic Computation, ASCE Structural Division. ASCE National Structural Engineering Meeting. 1973.

Leere Seite
Blank page
Page vide

IABSE
A I P C
I V B H

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

STATIK: A Computer Program for Everyday's Structural Engineering Applications

STATIK: Un programme d'ordinateurs pour les problèmes courants de génie des structures

STATIK: Ein Computerprogramm zur täglichen Anwendung im Bauingenieurwesen

E. ANDERHEGGEN

Professor for Applied Computer Science
Swiss Federal Institute of Technology
Zürich, Switzerland

Summary

Within the context of some of the developments in the field of computer applications to structural engineering which took place in Switzerland in the last 15 years, the criteria followed for the design of a new general purpose computer program called STATIK are presented. A number of specific questions concerning man-machine interface problems are raised which may serve as a basis for further more general discussions.

Résumé

On discute les critères retenus pour le développement d'un nuveau programme d'analyse structurale appelé STATIK, en tenant compte des conditions particulières qu'on retrouve en Suisse dans le domaine de l'application des ordinateurs au génie civil. Un certain nombre de questions concernant la relation homme-machine peuvent servir de base à des discussions ultérieures.

Zusammenfassung

Es werden die Kriterien geschildert, die zur Entwicklung eines neuen Computerprogrammes namens STATIK geführt haben, unter Berücksichtigung der speziellen Bedingungen, die in der Schweiz auf dem Gebiet der Computeranwendungen im Bauingenieurwesen zu finden sind. Eine Reihe Fragen tauchen auf, die als Basis für allegemeinere Diskussionen dienen können.

1. INTRODUCTION

If one considers the way computer programs for structural engineering applications are written and used the following classification seems reasonable:

a. Programs, generally written by non-professional programmers which are used only by their authors or by few very closed associates of their authors. Such programs are often found in the larger firms owning a computer. In fact, although this looks like a great duplication of efforts, it might well make sense to write programs for more or less personal use as the interface problems we are concerned with in this colloquium can then be easily solved in a local, personal way. As an example it is believed that most optimum design programs used today in structural engineering (and there are not many of them) are of this kind, design being too much influenced by personal taste and habits to be left to some not clearly understood black box program written for general use. But of course not everybody can write his own programs.

b. Programs written by professionsl programmers requiring from the user a high level of specific competence generally not found among practising engineers. Most well-known general purpose finite element programs are of this kind, an extreme example being NASTRAN which was written for use in the aerospace industry by highly professional numerical analysts. The use of such programs for civil engineering applications generally requires the professional services of a specialized software firm acting as an interface between the program and its user.

c. Programs written by professional programmers to be used directly by practising structural engineers for whom numerical analysis is only one, and seldom the most important, aspect of their professional activities.

The present paper is only concerned with this last kind of programs where interface problems between automatic computation and everyday's design work are of greatest concern. The criteria to be followed when developing such programs as well as the problems arising by their use are to be discussed. It is felt, however, that a discussion in very general terms would not make much sense. Too many factors greatly varying from place to place would have to be taken into account. Therefore, only some aspects of the developments which took place in Switzerland in the last 15 years and which are closely related to the activities of the writer shall be discussed. It is hoped that such a "case study Switzerland" will give rise to useful discussions and lead to generally valid conclusions.

The following factors certainly had a great influence on the historical developments of computer applications to structural engineering in Switzerland and indeed it would be interesting to compare and discuss the influence that similar factors had in different countries:

a. Swiss structural engineers are not too often confronted with problems where a sophisticated structural analysis would make much sense: Switzerland has practically no aerospace industry; unusual or unusually large structures are rare; rivers flowing near their sources are narrow and therefore not too difficult to span; since over 400 years no major earthquake has occurred; most large dams were designed and built before the advent of computers. However, there are some large reinforced concrete structures for nuclear power plants being designed today.

b. For many years Switzerland had by far the highest per capita cement comsumption of the world which is a good mesure of overall construction activities. Worth mentioning are the national and cantonal road construction programs with hundreds of individually designed, very slender and elegant cast-in-place reinforced concrete posttensioned bridges.

c. The vast majoritiy of Swiss civil engineers confronted with structural design works in relatively small, privately owned consulting offices. This is also a consequence of the Swiss political system as public works (including national highways) are generally managed by the cantons which tend to prefer local consulting firms.

d. Swiss building codes, at least compared with German, are rather liberal allowing considerable freedom in choosing design methods. No state employee checking each single computation exists as this is the case in Germany with the so-called "Prüfingenieure".

e. Switzerland is a rich country with one of the highest computer hardware density of the world. Access to computer facilities is therefore in general easy. As an example the Swiss Federal Institute of Technology in Zurich with some 7200 mostly undergraduate students has main computer facilities worth over 50'000'000 Sfr. not counting many small computers scattered about the institutes.

f. Four years of undergraduate studies are needed to become a civil engineer. At the Swiss Federal Institute of Technology in Zurich out of 12 mandatory and 4 non-mandatory semester courses on statics and structures only one non-mandatory course in the 7th semester deals with computer methods for structural analysis. Graduate studies leading to anything similar to a master's degree do not exist and only some 5

or 6 students reach each year the Ph. D. degree in civil engineering.

Within this framework a small group of research workers and Ph. D. students headed by the author of this paper has been active since 1962 at the Swiss Federal Institute of Technology in Zurich in the field of computer applications to structural engineering. One of our objectives has always been the development of computer programs to be used both for teaching purposes at our school and for practical applications by consulting structural engineers.

2. BACKGROUND HISTORY: THE PROGRAM STRESS

In early 1964 a magnetic tape with the source code of the program STRESS was sent to us from the Massachusetts Institute of Technology. After two years of efforts a modified version of the program was installed on the main computer of our school and a course for practising engineers was announced. We had, however, so many inscriptions that a second course in 1967 had to be held bringing the total attendance to nearly 600 engineers, a very large number for Switzerland. The programm STRESS has been used ever since for teaching purposes at both Federal Institutes of Technology in Zurich and Lausanne and found wide acceptance among Swiss consulting engineers. It certainly contributed very much to the spread and to the understanding of computer methods in structural engineering in Switzerland.

The main reason for this success is due to the fact that STRESS was the first finite element program specially designed for civil engineering applications with limited but clearly defined objectives: it can handle only linear elastic frames and trusses which is what most structural engineers not only need but also clearly understand; it does not attempt to design anything which, even today, would be a rather hopeless objective for a general purpose program; is is easy to use due to its simple problem-oriented input language which in many cases allows the preparation of new inputs just by extrapolation from old ones without having to study each time the user's manual, a big advantage specially for sporadic users.

The developments which took place at the MIT after STRESS are well known: the much publicized ICES project failed to attain many of the extremely ambitious and clearly utopian objectives that were set and was finally abandoned at the MIT. ICES, however, specially in Europe, had some offsprings like the "integrated" (whatever that means) systems GENESIS in England, ITS in Germany or SYSFAP in Belgium and it would be interesting to hear some comments on such programs from people directly involved in their development and use. The writer is rather skeptical toward such integrated systems, unless the word "integrated" is used in a restrictive sense meaning a series of programs for a specific application (like highway design) where the output of a program serves as input for its successor and not in

his original ICES sense meaning a number of programs for totally different applications (like survey, structural analysis and project management) all working on the same data base describing the object to be designed and built in very general, application independent terms.

3. STRESS'S SUCCESSOR: THE PROGRAM STATIK

After STRESS our group was involved in basic finite element research which led, of course, to many programs for personal use but also to two programs for general use: a program called PLATTE for linear elastic plate bending analysis where, for the first time, syntax diagrams for input description, as explained later, were used as well as extensive graphical output of results (e.g. contour lines of bending moments envelopes) and a program called FLASH for plate bending, plate stretching and shall analysis. The program FLASH is today regularly used for teaching purposes and has also become quite popular among Swiss consulting engineers being efficient and very easy to use. However, the points we want to make in this paper are probably best understood if we consider the criteria which led to the latest and largest of our programs: the program STATIK.

Although the STRESS program was a success, we were quite unhappy with it during many years: it was very inefficient (which many users never noticed as they never tried to analyse large structures); it has never been completely error-free mainly due to the unnecessarily complicated internal data organization making it extremely difficult to find programming errors; it could only handle prestressing in a very primitive way; it had no graphical output, no restart capabilities, etc. Finally, we decided to write a new state-of-the-art program to be used directly by Swiss consulting engineers assuming from them very much the same degree of competence as needed for STRESS and handling the same kind of everyday's relatively simple design problems. With a total effort of approximately 5 to 6 man-years the program STATIK was then developed according to the following criteria.

When using STATIK no difficulty should arise concerning the mathematical model used, i.e. only those problems are to be handled where an exact solution for a clear and well understood approximation of reality is possible. This requirement excludes all kind of nonlinear, dynamic and two-or threedimensional problems where the choice of the element mesh has an influence on results. STATIK can only handle linear elastic framed structures under statical loads as well as different kinds of cross section calculations. Much attention was paid to the load case prestressing. We also tried (and it is not yet clear if it is relly being used) to implement some simple design procedures for prestressed and non prestressed symmetric reinforced concrete cross sections.

Input preparation should be very easy also for relatively inexperienced sporadic users who are only supposed to read a very short user's manual (48 pages including examples and appendices) once. This is achieved by using so-called syntax diagrams for defining and describing in a very concise and clear way the problem-oriented, free format input language of the program. Figures 1 to 4 show some of these syntax diagrams. They are easily understood observing a few simple rules: the sequence of input data is found by following the arrows; a thick stroke corresponds to the beginning of an input statement, a triangle to its end; the first letter of the upper case words has to be punched on cards or typed on a terminal as it is; lower case words are identifiers referring to numerical or nonnumerical problem data to be specified; what is written between brackets can be left out, etc. The syntax diagram showing the overall structure of the program consisting of seven different modules ("QUERSCHNITT PROGRAMM" to "AUSGABE VON EINFLUSSLINIEN") is given in fig. 1. Fig. 2 shows the syntax diagram of one of these modules ("STRUKTURELLE EINGABE") used for specifying the structural input data of very general three-dimensional space frames with straight and curved members. Fig. 3 shows the syntax diagram of the program module "RESULTAT AUSGABE" used for requesting numerical and graphical output of results. A similar syntax diagram of the program FLASH for the input of structural data for very general continuous plane and space structures is given in Fig. 4. Syntax diagrams have proved to be an extremely useful tool not only for describing input languages (they are also becoming a standard tool for describing and defining programming languages, e.g. the syntax of the new ANSI-standard FORTRAN77 is defined by means of similar so-called railroad diagrams), but also because they immediately show what a program can do. Today, we consider them an indispensable feature of all programs written for general use.

Input echo and numerical output of the program STATIK appear on numbered pages of standard format (A4) with one or two headlines at each page which are generally used for the name and address of the consulting firm using the program. Printer control characters are not used as they are not understood by most terminals. Such details are mentionened here because they are very welcomed by practising engineers and also because they complicate programming considerably.

The Program STATIK has extensive graphical output capabilities. In all cases the drawing area is assumed to be 37 x 27 cm which corresponds to the screen size of the large storage tube Tektronix terminals. In fact, the STATIK program produces a graphic file which can be viewed directly on such terminals, a postprocessor being necessary to obtain the same drawing in any size on a plotter or on any other graphic device. Figures 5 to 8 show some of these drawings.

The program STATIK is written in FORTRAN for a large CDC Cyber Computer with a conversational remote job entry system (not a time-sharing system). It is specially designed to be used from remote terminals connected by telephone. This has the advantage of requiring only minimal fixed hardware investments from the user: a cheap alphanumeric terminal is all it is needed at least to start with. Later, a faster printer, a graphic terminal possibly with a hard-copy unit or a small plotter can be added to improve speed and user's comfort.

The program STATIK, if requested, automatically saves all problem data at the end of the last program module executed. The computations for a specific object can therefore take place in any number of subsequent jobs being possible to change problem data at any time and to execute program modules in any order. Some kind of interactive "computer aided design" is therefore possible. It should be noted, however, that interaction between the program and its user takes place at the level of subsequent, often very short jobs, and not at the level of each single line of input text as this is the case when using a time-sharing system. In fact, we think that for most structural engineering applications time-sharing would be an unnecessary luxury.

In October 1977, a course on the programs STATIK and FLASH with an attendance of approximately 250 civil engineers was held. It is too early to know if the success we had with STRESS can be repeated. Among our students, however, STATIK has been intensively used since nearly two years for all kinds of applications (sometimes without any specific theoretical background knowledge) becoming quite popular indeed.

Of course, as always when a program has been written, there are some features of STATIK we are not so happy about today. We also had some critics.

Some think that university employees should stick to research work instead of writing commercial programs like STATIK. Our answer is that the efforts leading to programs like STATIK are indeed to be considered research work, not in the field of structural analysis of course, but in the field of computer science. In fact, we did our best to solve the interface problem between the computer and its users just like conventional computer scientist do, for different problems and different users, when they develop, say, a new programming language. Our goals, however, are only attained when many peoples use our programs as an instrument for their professional activities which is only possible in a commercial environment.

It would be very much in line with our purposes to have STATIK run on relatively small computers as it could then be made available to many more users (e.g. a PDP 11/60 would certainly be powerful enough for most applications). We must admit, however, that we made a mistake experienced programmers should not make: in order to improve efficiency we introduced many

machine dependent features greatly impairing program portability. Although STATIK is completely written in FORTRAN it would certainly be an extremely long and tedious task to develop today a general, machine independent version of the program.

Graphical output was planned having in mind Tektronix storage tube terminals combined with hard-copy units as we thought such terminals will have a great future being relatively cheap and easy to connect to a large computer using asyncronous low-spead (300 to 1200 Baud) data transfer. Today, we are not so sure about this anymore. In fact, inexpensive small plotters working in the same way already exist and recent advances in microcomputer technology may soon make graphical refresh terminals with a high level of built-in intelligence easily available. However, the advent of new graphic terminals will not really impair the use of the program STATIK being a simple task to have the program produce a hardware independent instead of a Tektronix-oriented graphical file. Postprocessors are then used to produce graphical output on different hardware units (this solution was already implemented in one version of the program STATIK running in a large computing center in Zurich).

Quite contrary to many other programs who use English for input and output (e.g. STRESS), STATIK and FLASH use German. Rather provincial reasons led us to this choice as we really did not want our programs to be used from people living too far away from us, whose problems, habits and level of competence we do not understand too well. We were also somehow afraid of the maintenance problems arising when too many versions of the program are running in different places. Unfortunately, Switzerland is a multi-lingual country, and we already had to hear the complaints of our collegues at the Swiss Federal Institute of Technology in Lausanne who would be much happier with French versions of the programs.

4. FURTHER DEVELOPMENTS AND CONCLUSIONS

No further development of our programs FLASH and STATIK are planned as this would contrast with the objectives we pursued with them. However, we certainly want to pursue similar objectives in the future for different classes of problems and assuming from our users a different level of competence. Graduate studies leading to something else then a Ph. D. degree shall be introduced soon at our school which will also have a direct influence on our activities. In fact, we feel that there is ample room for computer scientists involved with practical computer applications to try to span the gap between the real needs of today's structural engineering and much of the sometimes brilliant but often isolated university research work.

General conclusions shall not be drawn here, the "case history" presented being rather intended to raise questions than to anwer them. However, as a kind of summary, some of these questions shall be given hereinafter.

Does a classification of programs distinguishing programs for more or less personal use, programs requiring specialists help and programs to be used directly clarify the situation ? How should a programmer take into account the way his program will be used ?

What is the influence of local conditions like those given for Switzerland on the development of computer applications to structural engineering in different countries ?

In which circumstances can integrated systems be useful ?

What can be done in order to make the use of a program, i.e. the preparation of input data and the interpretation of results, as easy as possible ? In which cases is this of primary importance ?

How can computer-aided design procedures be helpful in everyday's structural design work ? What kind of an interaction between the engineer responsible for the design and the computer is needed ? Can automatic optimum design programs be useful for structural engineering applications ?

What will be the influence of recent advances in computer technology like the proliferation of minicomputers to be used both off-line and in-line with a host computer ?

What should civil engineering students learn in order to be able to use such new instruments properly ?

It is hoped that such questions will help to clarify the interface problems we are concerned with in this colloquium and lead to useful and more general discussions.

Notice: The User's Manuals of the computerprograms STATIK and FLASH can be obtained from the Institut für Baustatik und Konstruktion, ETH-Hönggerberg, 8093 Zürich, Switzerland.

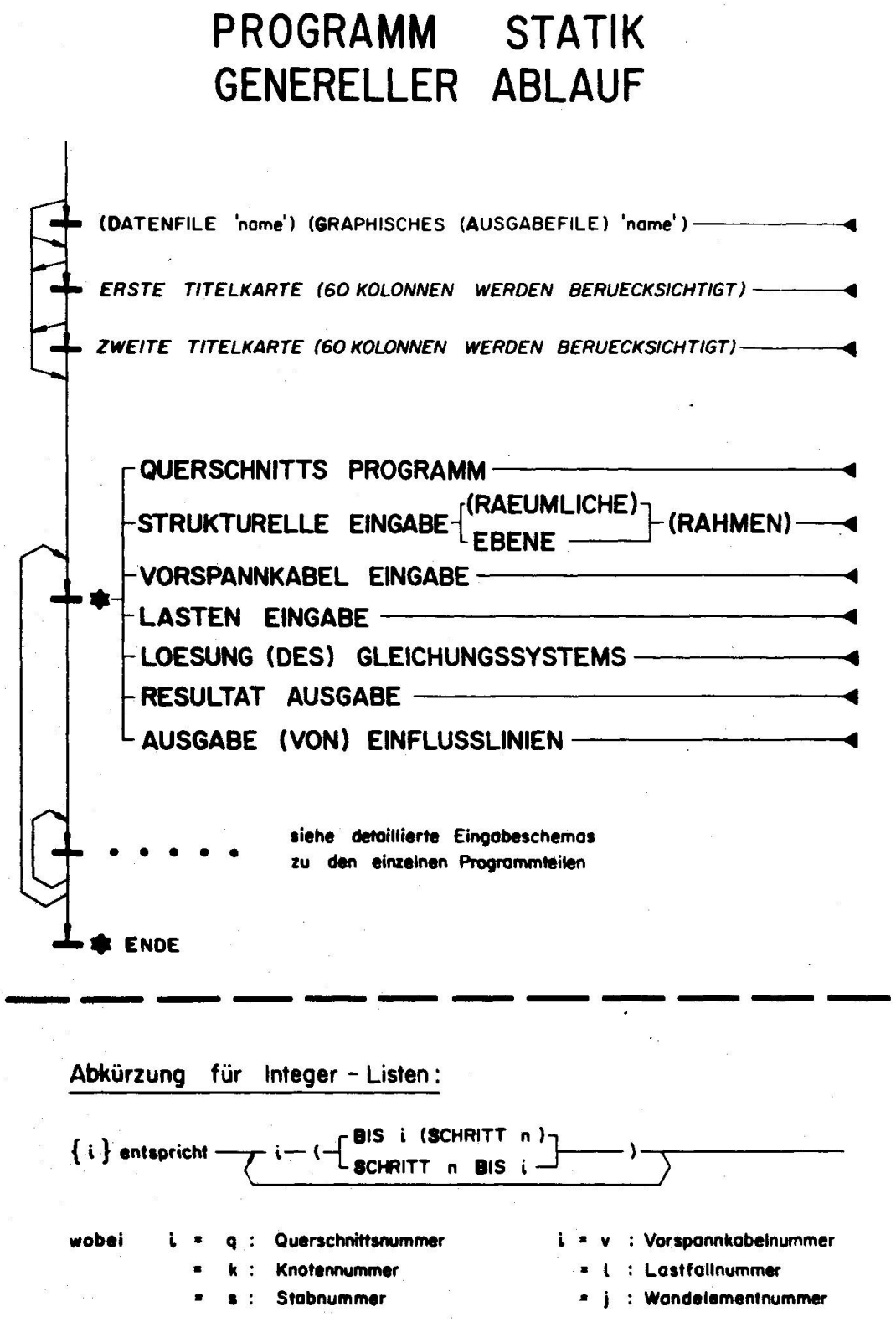


Fig. 1: Syntax diagram showing the overall structure of the program STATIK consisting of seven program modules callable in any order after specifying problem files and output headlines. The syntax of integer input lists used in subsequent diagrams is also shown.

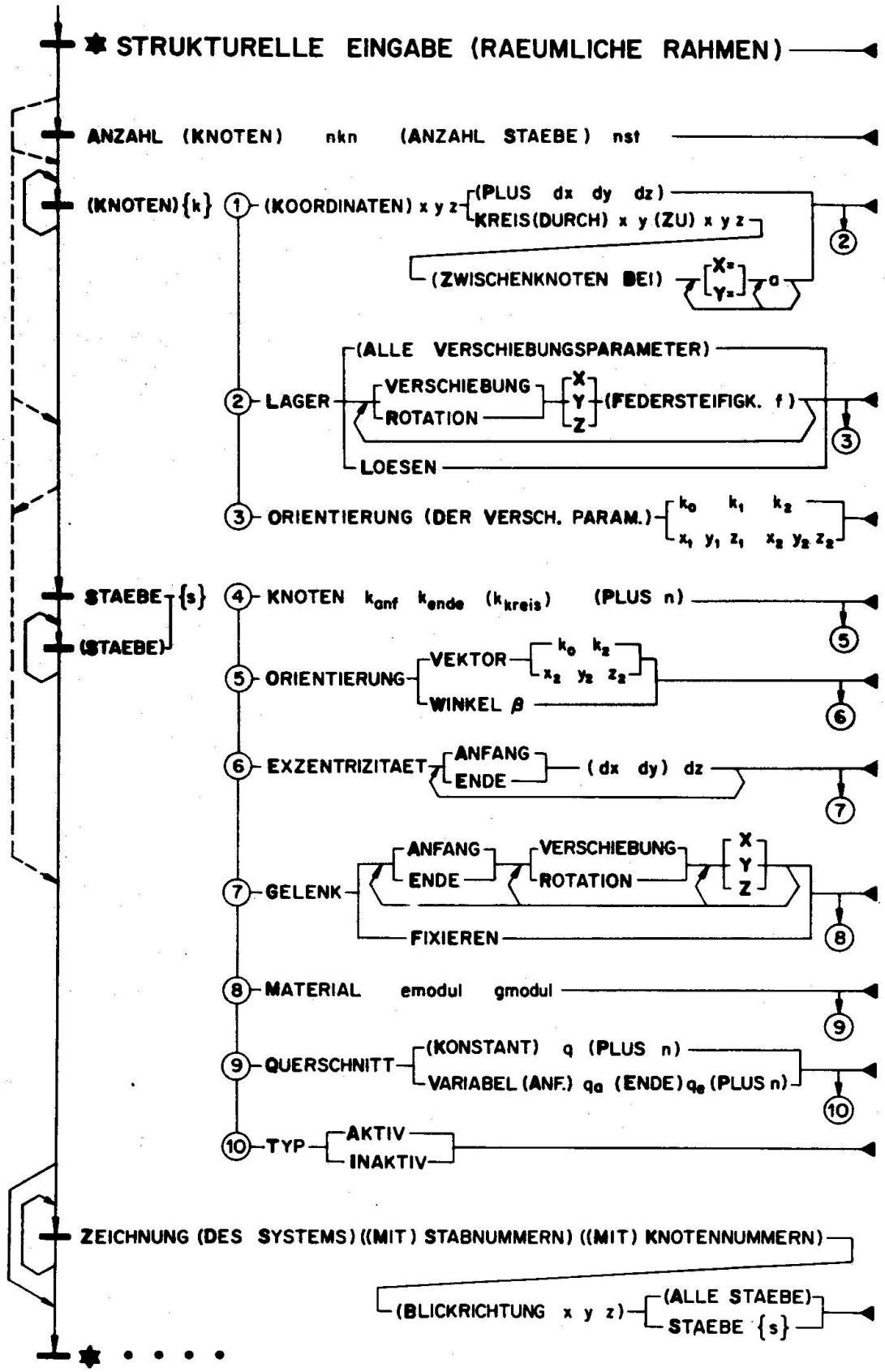


Fig. 2: Syntax diagram of the program STATIK for the input of structural data for space frames.

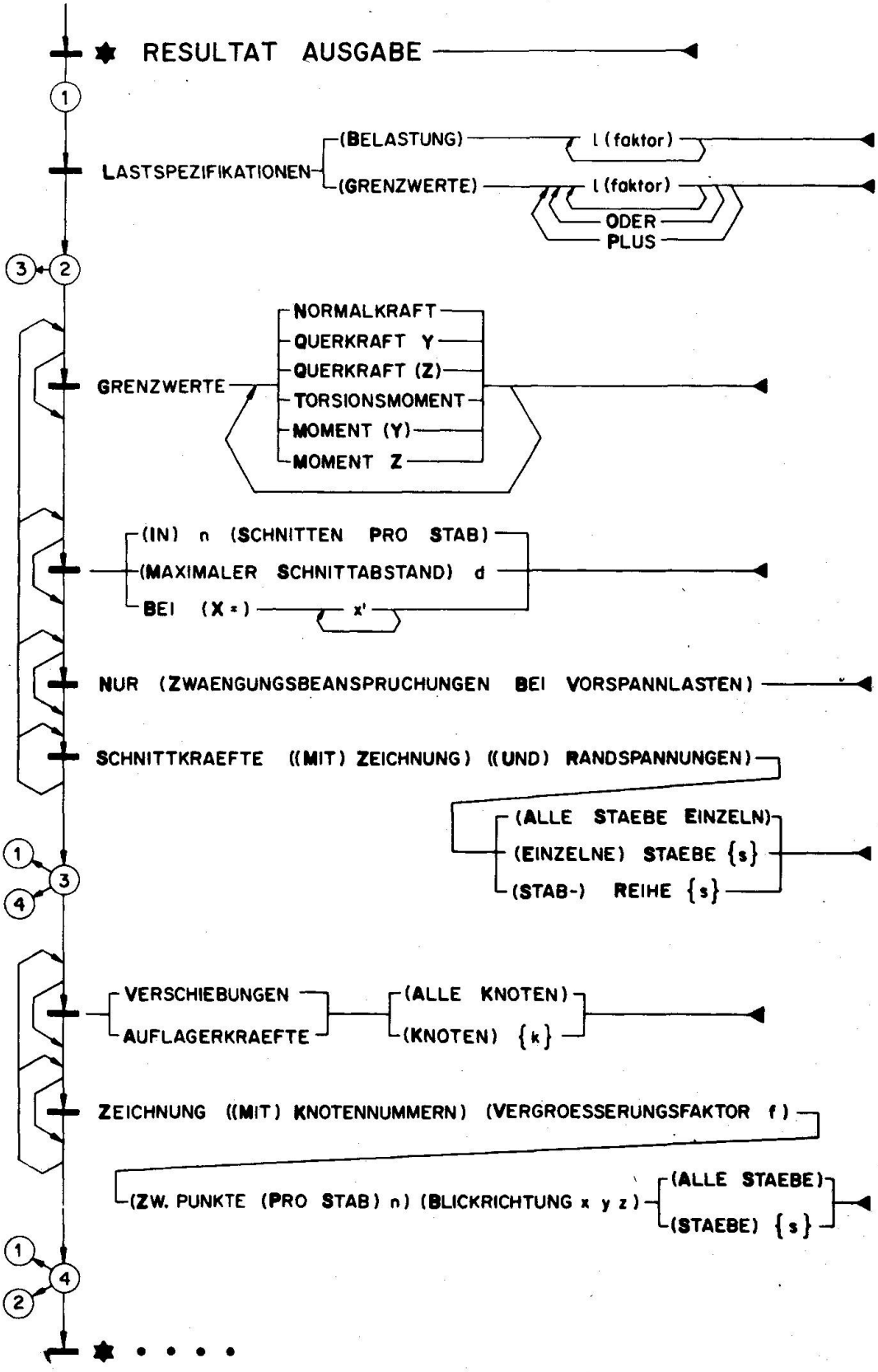


Fig. 3: Syntax diagram of the program STATIK for requesting numerical and graphical output of results.

I

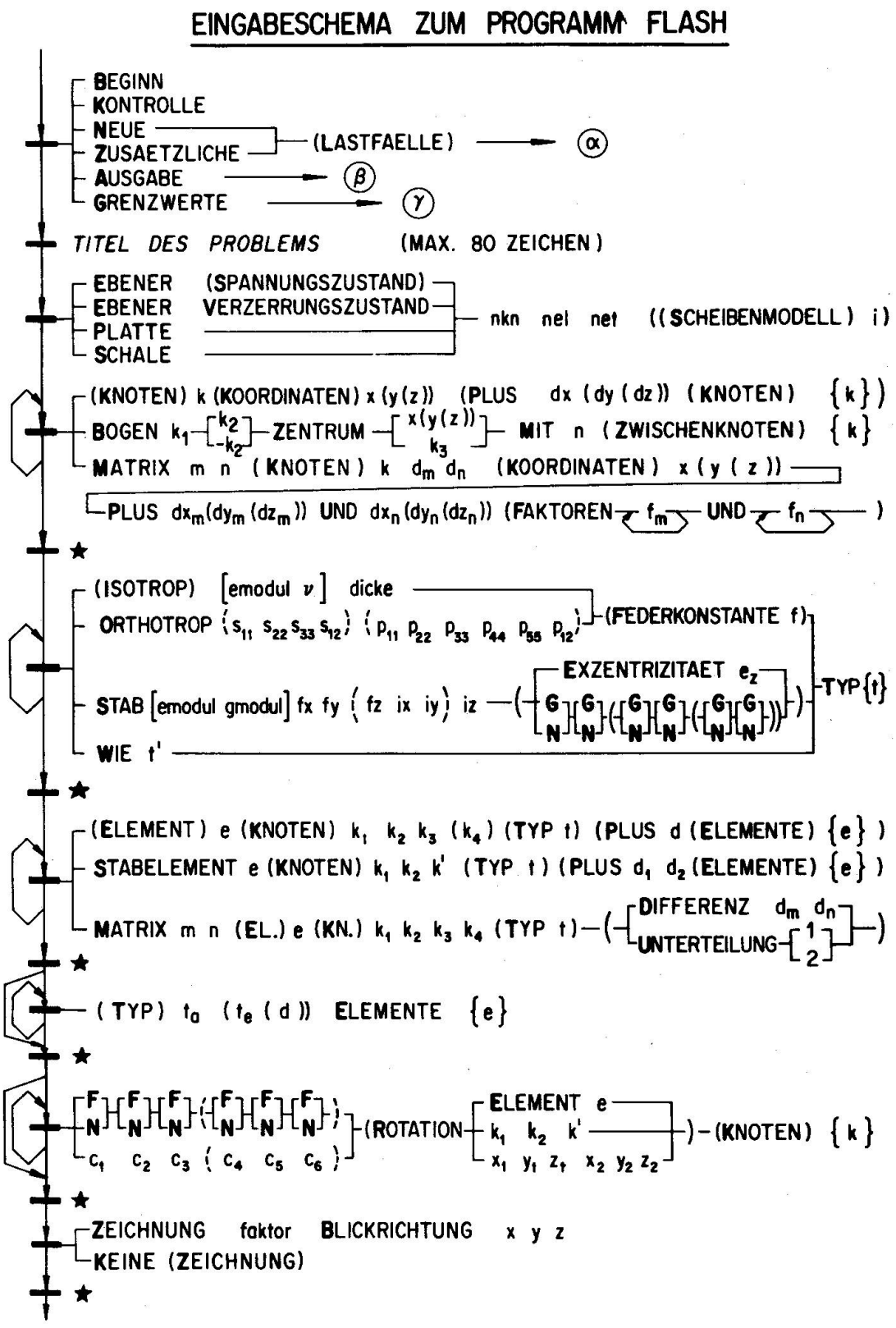


Fig. 4: Syntax diagram of the program FLASH for the input of structural data for plates and shells.

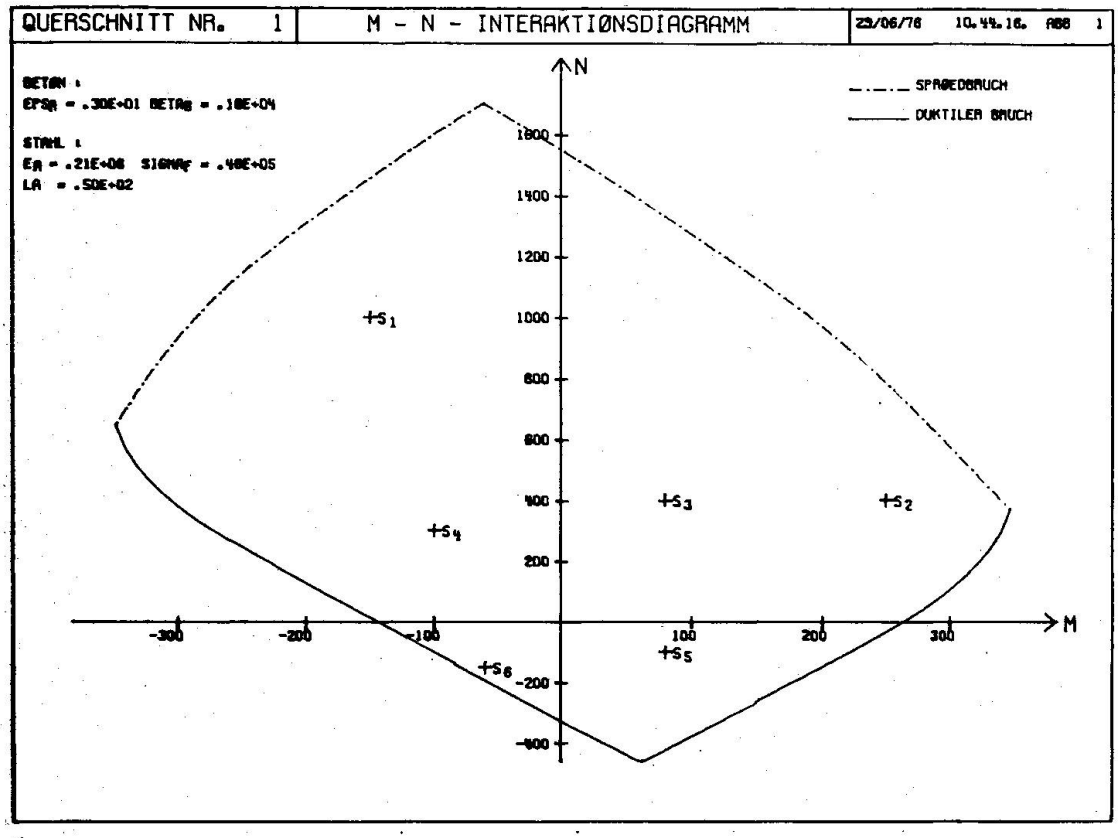


Fig. 5: Graphical output of normal-force bending-moment interaction for a reinforced concrete prestressed cross section.

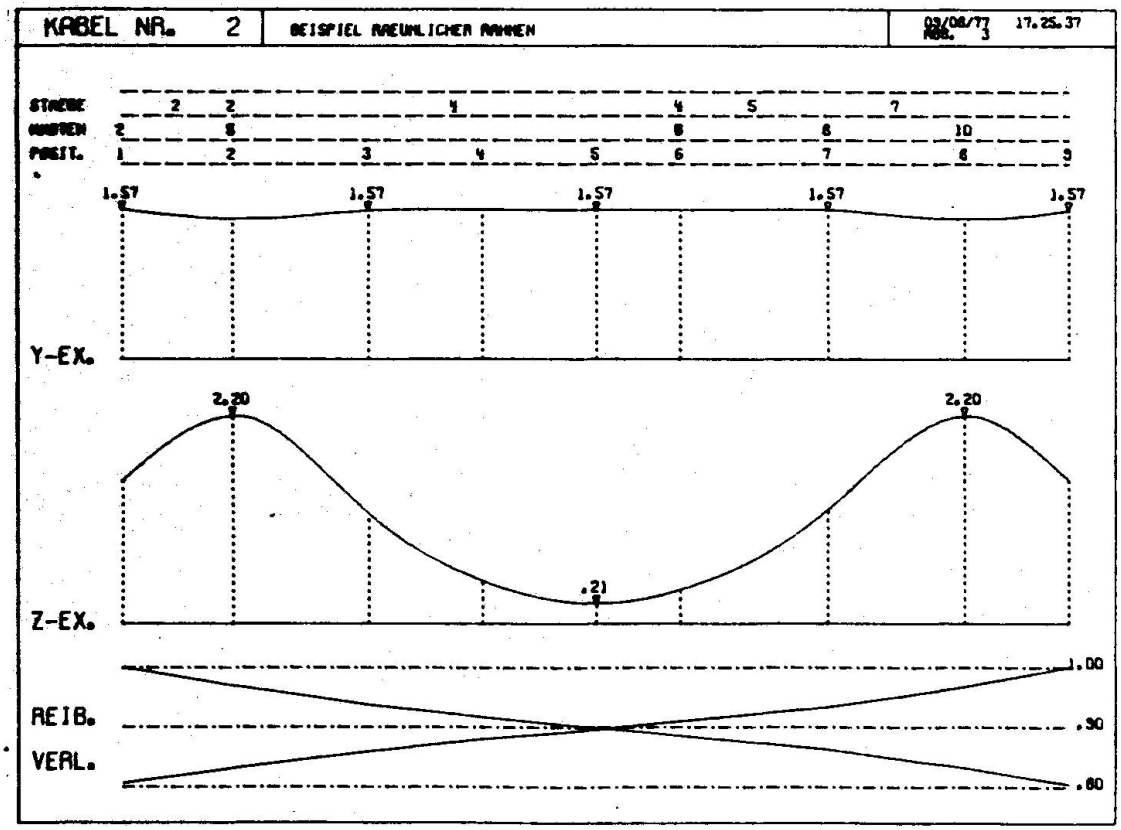


Fig. 6: Graphical output of prestressing cable geometry and friction losses.

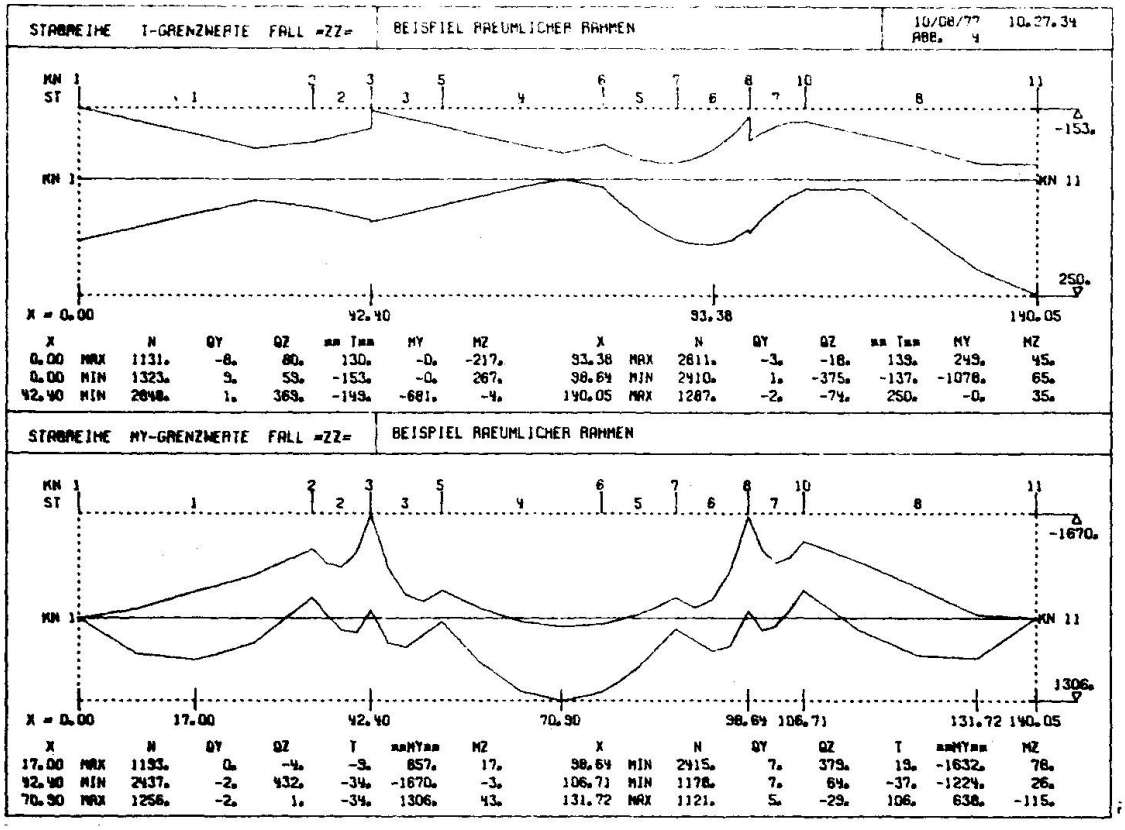


Fig. 7: Graphical output of torsional- und bending-moment envelopes.

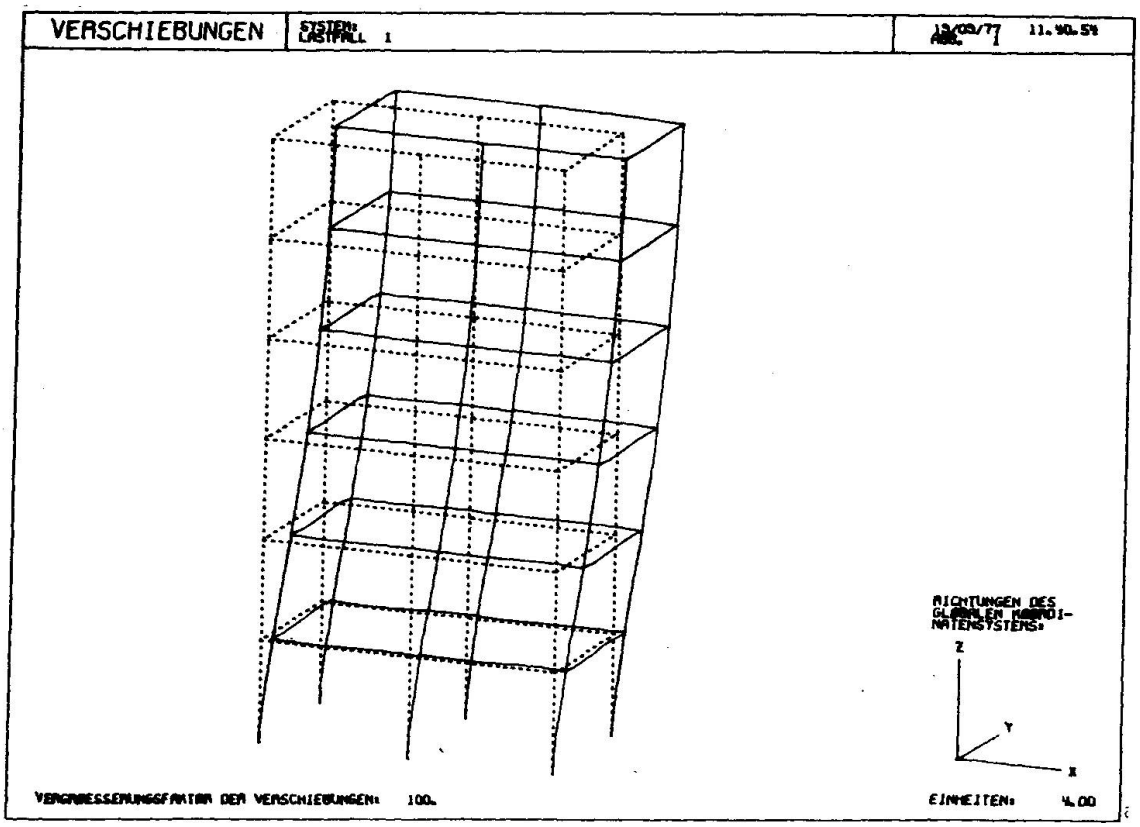


Fig. 8: Graphical output of the displaced shape of a multistory space frame.

Leere Seite
Blank page
Page vide

IABSE
A I P C
I V B H

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Considerations on Proper Usage of Design Programs

Considérations sul l'usage correct de programmes de calcul

Uber die richtige Anwendung eines Entwurfsprogramms

F. TAKINO

Staff Engineer
Musashino Electrical Communication Laboratory
Nippon Telegraph and Telephone Public Corporation
Tokyo, Japan

Summary

The correctness of program usage is as important as the correctness of programs themselves, especially in such a case that programs are offered to unexperienced users. This paper describes a data error check system and some materials for avoiding improper program usage in a building structure automated calculation system. These materials include a program application checklist and a guide on the preparation of structural calculation sheets.

Résumé

L'utilisation conforme de programmes est aussi importante que l'exactitude des programmes eux-mêmes, surtout quand ils sont utilisés par des usagers inexpérimentés. Cet article expose un système qui permet de vérifier des erreurs de données et quelques mesures à prendre contre de faux usages de programmes pour le système de calcul automatisé de structures. Ces mesures comportent un tableau vérificateur pour l'application de programmes, ainsi qu'un guide pour la préparation des fiches de calcul de structures.

Zusammenfassung

Die Richtigkeit der Anwendung eines Programms ist genauso wichtig wie die Richtigkeit des Programms selbst. In diesem Aufsatz wird ein Kontroll-System zur Erfassung von Datenfehlern beschrieben und auf Dinge hingewiesen, die ungeeignete Programmanwendung in einem automatisierten Kalkulationssystem zur Baukonstruktion vermeiden sollen. Diese Materialien beinhalten eine Checkliste zur Programmanwendung und Richtlinien zur Vorbereitung von Blättern für die konstruktionsberechnung.

1. INTRODUCTION

This paper deals with several considerations on avoiding improper program usage concerning building structure automated calculation programs.

Just like the correctness of programs, the correctness of program usage is important, especially in such a case that programs are offered to anyone without regards to his ability or his experience. It is a regrettable fact that users are not always experts or excellent engineers. It is a matter of deep concern, especially for building officials, that unexperienced engineers use automated calculation programs blindly and design structures directly under instruction from computers. Possibly there are some mistakes in input data, or some misunderstandings regarding user's manuals. Undue reliance upon computers can lead engineers to catastrophe.

Throughout Japan, several hundreds of building engineering firms subscribe to a time sharing system called DEMOS-E offered by Nippon Telegraph and Telephone Public Corporation, whereby many subscribers are frequently carrying out building structural design by using library programs through the telephone network. In this case, much attention has to be paid to ascertainment of proper program usage within its application range and to correctness of input data.

DEMOS-E BUILD is a series of building structure automated calculation library programs from preparatory calculation and stress analysis to member design. The design methods adopted herein are based on structural standards of the Architectural Institute of Japan (AIJ). Compared with manual calculation, engineers can treat in these programs more complex structural calculation, such as space frame analysis, with resultant reduction in necessary design manpower.

With a view to inspect and evaluate building structure automated calculation programs and to promote their correct usage, an official committee called Evaluation Committee of Structural Analysis by Computer has been organized in The Building Center of Japan (BCJ) since 1973. An outline of the committee is shown in the Appendix.

DEMOS-E BUILD programs were the first programs submitted to the committee. Under demand from the committee regarding DEMOS-E BUILD, a program application check-list and a guide on the preparation of structural calculation sheets to be submitted to officials for building permits were devised.

In this paper, an outline of DEMOS-E BUILD is described first, and then techniques to avoid improper program usage are presented.

2. DEMOS-E BUILD OUTLINE

2.1 BUILD Function

DEMOS-E BUILD is a series of building structure automated calculation programs which are furnished to DEMOS-E subscribers as library programs [1], [2]. In Japan, reinforced concrete structures are generally adopted for low buildings (20 meters high or less), steel-reinforced concrete structures are adopted for fairly high buildings (50-60 meters high) and steel structures are adopted for low buildings as well as for high buildings. BUILD covers these three types of building structures, having regular rectangular frames composed of columns and girders. Calculation and design methods used in BUILD are based on AIJ's structural standards.

BUILD is composed of 7 package programs, as shown in Figure 1. Outlines of these programs are:

(1) BUILDCK (Input data check): Input data are checked by real time processing.

(2) BUILD-P (Preparatory calculation): Loads, such as fixed end moments of girders, axial forces of columns, and lateral forces caused by earthquakes, are calculated. Quantities of members are also covered.

(3) BUILD-S (Stress calculation): Stresses and displacements of frames are calculated, either by plane frame analysis or by space frame analysis. In the former, vibrational characteristics of buildings are obtained which can be analysed in detail by using DYNA (earthquake response calculation library program).

(4) BUILD-M1 (Reinforced concrete member calculation): Reinforcing bars of girders, columns, and walls are calculated for bending and shearing forces.

(5) BUILD-M2 (Steel-reinforced concrete member calculation): Reinforcing bars and steel structures embedded in concrete of girders and columns are calculated.

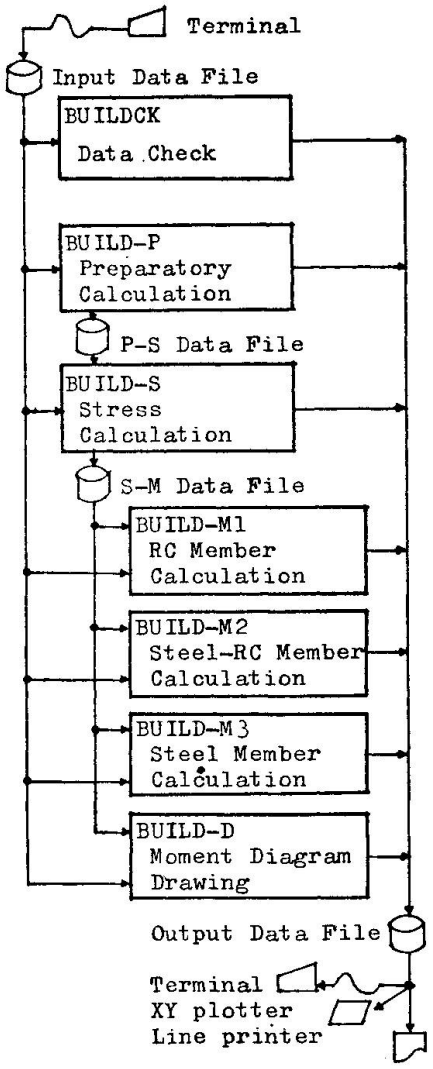


Fig. 1 BUILD Composition

(6) BUILD-M3 (Steel member calculation): Stresses are checked or suitable profiles are selected for various steel profiles of girders and columns.

(7) BUILD-D (Drawing moment diagrams): From the stress data output by BUILD-S, X-Y plotters draw moment diagrams for girders and columns.

In the case of simple structures, BUILD-P, S, and M1/M2/M3 are automatically processed in sequence, and required output can easily be obtained. In the case of complex structures, engineers use BUILD-P, S, and M1/M2/M3 repeatedly, as shown in Figure 2, by changing, for example, the sizes of structural members, or rigidity of walls. As input data are already stored in files, it is sufficient to transmit updating data only, thus reducing the engineer's work. By such repetition of calculation, more suitable sizes of members and arrangement of reinforcing bars are obtained.

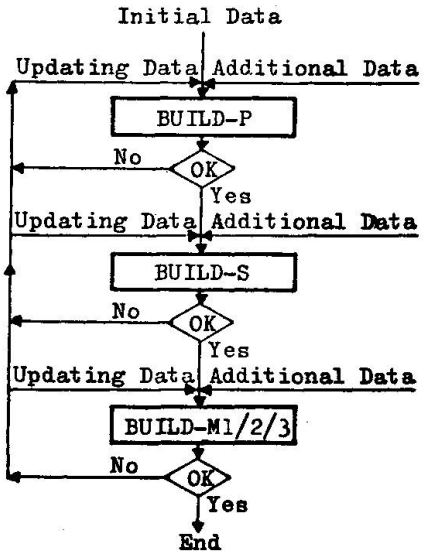


Fig. 2 Processing Flowchart

In 1973, when BUILD made its début, it was composed of 4 packages, namely BUILDCK, -P, -S, and -M1. Its size was 40K statements (mainly FORTRAN), but later its function was englarged and new programs were developed. At present, BUILD program size amounts to 110K statements.

2.2 State of Usage

BUILD programs are used considerably among building engineering firm subscribers throughout Japan. As an example, BUILD-P program is used several hundreds times in every month. Statistics indicate that frequencies of using other BUILD programs are, in comparison with BUILD-P usage frequency, 1.4 times in BUILDCK, 0.84 in BUILD-S, and 0.76 in BUILD-M programs (M1, M2, and M3).

An investigation regarding how BUILD programs were used among building engineering firms was held in 1976 [3]. Forty firms were chosen in Tokyo, Osaka, and Fukuoka for this purpose. Major results from the investigation are as follow:

(1) Main reasons why BUILD programs are used are: No. 1 — Shortening design time. No. 2 — Manpower reduction. Manpower reduction, in case of simple structures, amounts to 50% in preparatory calculation (BUILD-P), 50-90% in stress calculation (BUILD-S), and 20-50% in member calculation (BUILD-M programs).

(2) An investigation was made on how many times BUILD programs were operated in one case of structural designing because of data error or calculation modification (cf. Figure 2). Fifty cases were investigated. Results are shown in Figure 3. Major reasons for repetitive operation are: data error in BUILDCK, data error in 50% of the cases and calculation modification in 50% of the cases in BUILD-P, and modification of calculation in BUILD-S and -M programs.

(3) 40% of the total stress calculation cases in BUILD-S are calculated as plane frames, whereas 60% are calculated as space frames.

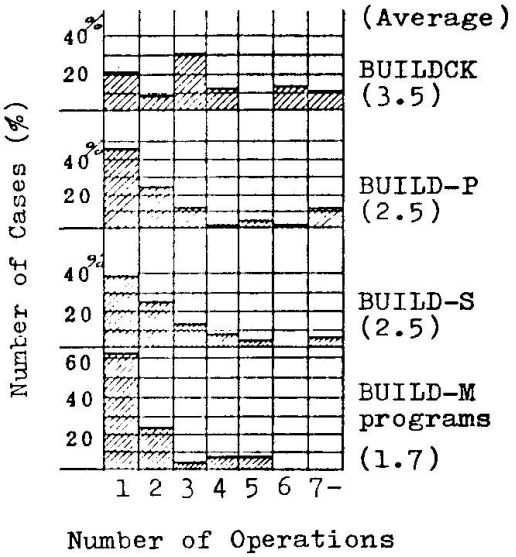


Fig. 3 Histogram on Number of Operations

3. PROPER DEMOS-E BUILD USAGE

3.1 Outline

It is desirable for users that program designers and suppliers prepare good input forms and output presentation, strict data check system, and good user's manuals. Moreover, in case of a building structure automated calculation program, it is recommended by BCJ's committee (cf. Appendix) that the program suppliers furnish a "Program Application Check-List", by which users can ascertain problems within the program application range, and a guide on structural calculation sheets preparation by which building officials can easily understand the contents of the structural calculation sheets submitted by users.

In DEMOS-E BUILD, several techniques to avoid improper program usage are considered, including a program application check-list and a guide on preparation of structural calculation sheets. By these techniques, users are considerably relieved from misuse and misunderstanding of BUILD programs.

3.2 Program Application Check-List

Purposes of this Check-List are:

(1) To ascertain that the structure concerned is within BUILD application range.

(2) To describe modelling of the structure to meet the requirement of BUILD, when the structure or a part of the structure is beyond the BUILD application range.

(3) To clarify and express important design conditions such as loads, materials, shape of the building, and foundations.

(4) To decide calculation methods, such as space frame analysis, consideration of foundations settlement, and consideration of rocking motion in earthquake resisting walls.

Using printed forms, users examine and check items and, if necessary, write their points of view in the forms. The Check-List is easily read by others and effectively used as a part of calculation sheet. Numbers of check items in BUILD's Program Application Check-List are, according to the above four categories, 13, 7, 27, and 39, respectively.

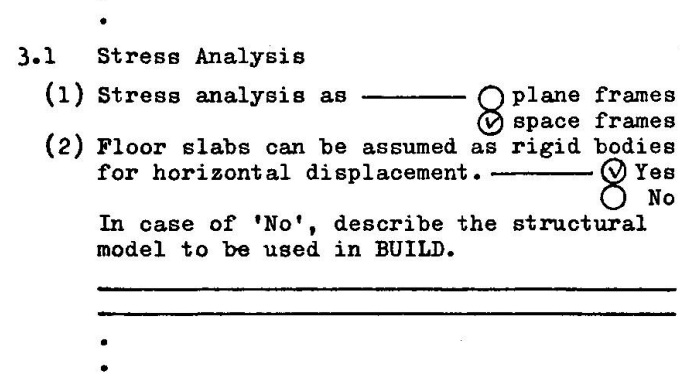


Fig. 4 Example of Program Application Check-List (originally written in Japanese)

Figure 4 shows an example of the Check-List translated into English from Japanese. In the near future, users will obtain BUILD's Program Application Check-List forms directly from BUILDCK program through user's terminals.

3.3 Error Check System in BUILD programs

There are three kinds of error check in BUILD programs, initial check at input data processing phase, intermediate check during calculation, and check after calculation.

(1) Initial check: Responding to each input record, syntax check, attribute check and limit value check are performed in the early stage of processing. Later, after all data are input, they are mutually checked to isolate contradictions between themselves. They are compared with building codes and structural calculation standards.

(2) Intermediate check: During structural calculation, the occurence of some irrational performances is watched for.

(3) Final check: Calculation results are usually checked to meet the requirements of building codes and structural calculation standards, especially in case of structural member design.

When input data or calculation results are found not to comply with the requirements of building codes and structural calculation standards, warning messages are output, such as: "Slenderness ratio of the steel column exceeds 200", and"Breadth thickness ratio of the flange plate exceeds limit value". Important warning messages, which have been output at proper

places, are gathered and listed again in the last part of output to make them conspicuous. BUILD programs include 110 prepared warning messages of above mentioned type and 260 other error messages.

Besides warning and error messages, BUILD calculates key values to express characteristics of structures such as each floor area and space volume, total weight and weight per unit floor area, quantities of concrete and steel per unit floor area, center of gravity and rigidity at each floor, slenderness of members, etc. By examining these values, users can verify the structural model, as well as check the input data comprehensively.

3.4 Input Data Abstract

The whole input data are output arranged into tables or appropriate forms. These arrangements aim at good documentation and are not always suitable for input data checking by users. Moreover, input data tables sometimes amount to a fairly great volume, in case of large structures.

"Input Data Abstract" is a summary of input data compiled in several pages by BUILDCK real time program using the whole input data for BUILD-P, -S, and -M1/2/3. It is useful for input data checking by users, as well as for concise description of the building structure, design conditions and calculation methods.

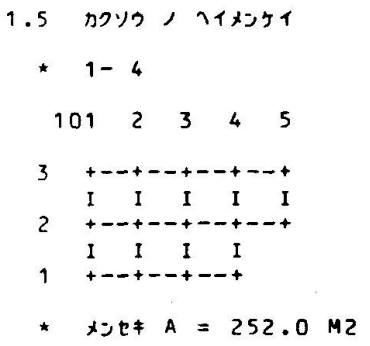


Fig. 5 An Example of Input Data Abstract (Floor Plan)

Several output items are as follow:
-Figure 5 shows a concise floor plan with floor area.
-Sizes of girders and columns are not output individually. Instead, maximum and minimum sizes and depth-length ratios in each story are output.
-When member sizes are altered among BUILD-P, -S, and -M programs, old and new member sizes and altered member positions are shown.

3.5 Graphically Arranged Output

Even in case of printers, graphically arranged output is more easily read and understood than lines of mere characters. On the recommendation of BCJ's committee, many graphically arranged output forms are introduced into BUILD programs, such as member arrangement plans (slabs, girders, columns and walls), column axial force list, member rigidity list, earthquake induced horizontal loads distribution in columns and walls, frame stress diagrams, and reinforcing bar schedules for girders and columns.

Figure 6 shows an example of a frame stress diagram.

3.6 Guide to Preparation of Structural Calculation Sheets

To use computer output alone as a formal document often arouse troubles for non-users, especially for building officials. Major causes for such troubles are:

(1) Usually, descriptive statements in the output are few in quantity and inferior in quality.

(2) Knowledge regarding the program's calculation functions and methods is needed.

(3) Often output amounts to a large volume, in case of a large structure.

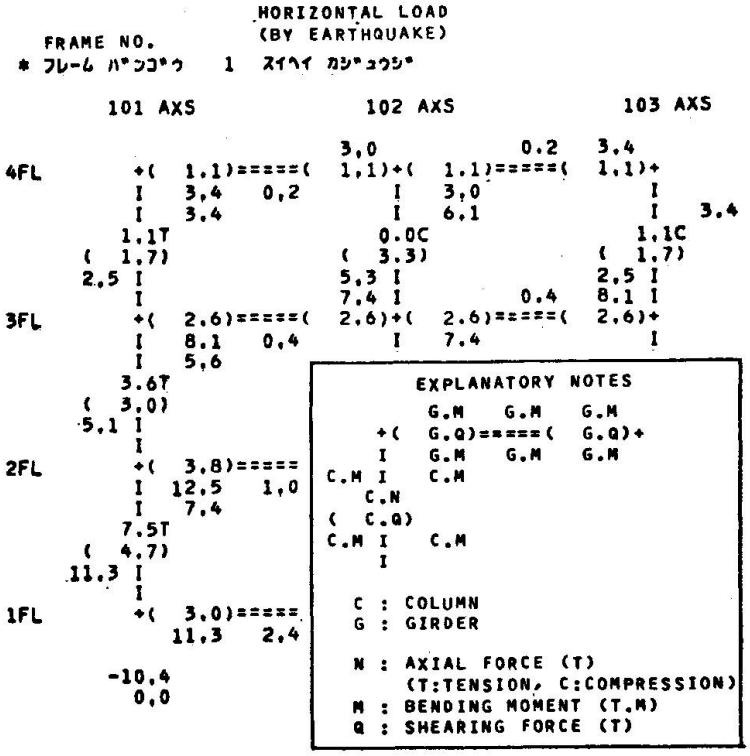


Fig. 6 Frame Stress Output

On the recommendation of BCJ's committee, a Guide on the preparation of structural calculation sheets was devised. According to the Guide, structural calculation sheets are composed of two parts, main text and supplementary data. Computer output is treated as the latter. The main text, written by hand, must contain the following items.

a) Structural modelling description.

b) Main or representative values from the output, in order to understand the calculation outline.

c) A summary of structural design and calculation, as well as remarks on structural safety, especially for earthquakes.

4. CONCLUSION

This paper describes several techniques for avoiding improper usage of programs adopted in DEMOS-E BUILD, such as error check system and other means, namely, program application check-list, output of input data abstract, graphically arranged output, and a guide on the preparation of structural calculation sheets. The program application check-list and the guide are furnished by the service supplier and are expected to be fully utilized by subscribers.

Considering the tendency for computer programs to become increasingly extensive and complicated with the progress of computer

hardware, it is important to develop proper methods for ascertaining the correctness of program usage so as to promote reliable and pertinent computer application to the structural designing.

ACKNOWLEDGEMENT

The author wishes to thank Professor M. Fujimoto, Chairman of the Evaluation Committee of Structural Analysis by Computer in The Building Center of Japan, for his advice on the preparation of this paper.

APPENDIX

Evaluation Committee of Structural Analysis by Computer in The Building Center of Japan

Several papers regarding this Committee, written in Japanese, have been published [4], [5], [6]. The following is a brief on the Committee using these papers.

The Committee was organized in 1973, and more than ten programs have been examined since then. Programs to be examined by the Committee are, for the time being, limited to building structure automated calculation programs. Members of the Committee are university professors, govermental institute researchers and building officials.

At the request of an applicant, the Committee examines the program concerned and the way of using it, regarding the following points:

(1) Within the application range of the program, are there any errors regarding methods of analysis and calculation as well as programing?

(2) Is there any possibility of the program being used erroneously? What is the pertinent precaution method?

(3) Is there any possibility of output data being used erroneously? What is the pertinent precaution method?

(4) When input and output data are used in structural calculation sheets to be submitted to officials for building permit, are styles of these data suitable for this purpose?

(5) In order that users master the program and its usage, are there any training courses available for users?

The result of examination is publicized and delivered to building officials. Users are recommended to use the program according to the way proposed by the Committee. Building officials are no longer required to check the program by themselves.

REFERENCES

1. TAKINO and KUWAGATA: Building Engineering Library Programs in DEMOS; Japan Telecommunications Review, Oct. 1974

2. TAKINO et al: Reinforced Concrete Building Structure Design Program; Review of the Electrical Communication Laboratories, NTT, Japan, Vol. 23, Nos. 9-10, Sep.-Oct., 1975

3. TAKINO and KUWAGATA: DEMOS-E Building Structure Design Library Programs, Investigation of Usage and Some Problems; The 23rd Structural Engineering Symposium, Tokyo, Feb. 22, 1977, by JSCE and AIJ (in Japanese)

4. FUJIMOTO: Technical Evaluation on the Structural Analysis Program with Electronic Computer; The Building Letter, No. 102, Jan. 1978 (in Japanese)

5. MORI: Procedure Based on Building Standard Law for Structural Analysis Program with Electronic Computer; The Building Letter, No. 102, Jan. 1978 (in Japanese)

6. OHTANI: The Reliance on Continuous Treatment Programs of Structural Calculation for Buildings; The 23rd Structural Engineering Symposium, Tokyo, Feb. 22, 1977, by JSCE and AIJ (in Japanese)

IABSE
AIPC
IVBH

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Development and Maintenance of Design-Supporting Software

Developpement et maintenance de programmes de calcul de structures

Entwicklung und Wartung entwurfsunterstützender Programme

H. WERNER

Professor, Dr. Ing.
Techn. Univ. Munich
Munich, German Federal Republic

Summary

Mature design-supporting software has to pass through four important phases of development:
1. design 2. Coding and test 3. Pilot installation(s) 4. Marketing, maintenance and enhancement. To neglet one phase endangers the success of the next one. From practical experience, the author states the aims of these development phases and offers advice on how to reach them.

Résumé

Les programmes de calcul de structures doivent passer par quatre phases pour être véritablement opérationnels:
1. Concéption 2. Realisation et essais 3. Installation pilote 4. Marketing, maintenance et développement. La négligence dans une phase compromet la suivante. Des remarques et reccomandations, résultant d'expériences pratiques, sont faites pour chacune de ces phases.

Zusammenfassung

Marktreife entwurfsunterstützende Software durchläuft vier wichtige Entwicklungsphasen:
1. Entwurf 2. Erstellung und Tests 3. Pilotinstallationen 4. Vermarktung, Wartung und Weiterentwicklung. Vernachlässigungen in einer Stufe stellen die erfolgreiche Durchführung der nächsten in Frage. Ausgehend von praktischen Erfahrungen werden Anmerkungen und Empfehlungen zu den einzelnen Entwicklungsstufen gegeben.

1. INTRODUCTION

1.1 Definition of Design-Supporting Software

In the process of structural design, the following steps can be defined [1]:

- collection of input information (e.g. preliminary dimensions, loads, construction stages);
- development of a model for analysis (e.g. an FE model);
- analysis
- interpretation and modification of the results according to standards and practical experience;
- calculation of the members;
- construction, i.e. preparation of drawings, details etc.

In each step, information gained by the preceeding steps is used in addition to new data.

Fig. 1 shows a scheme of this process.

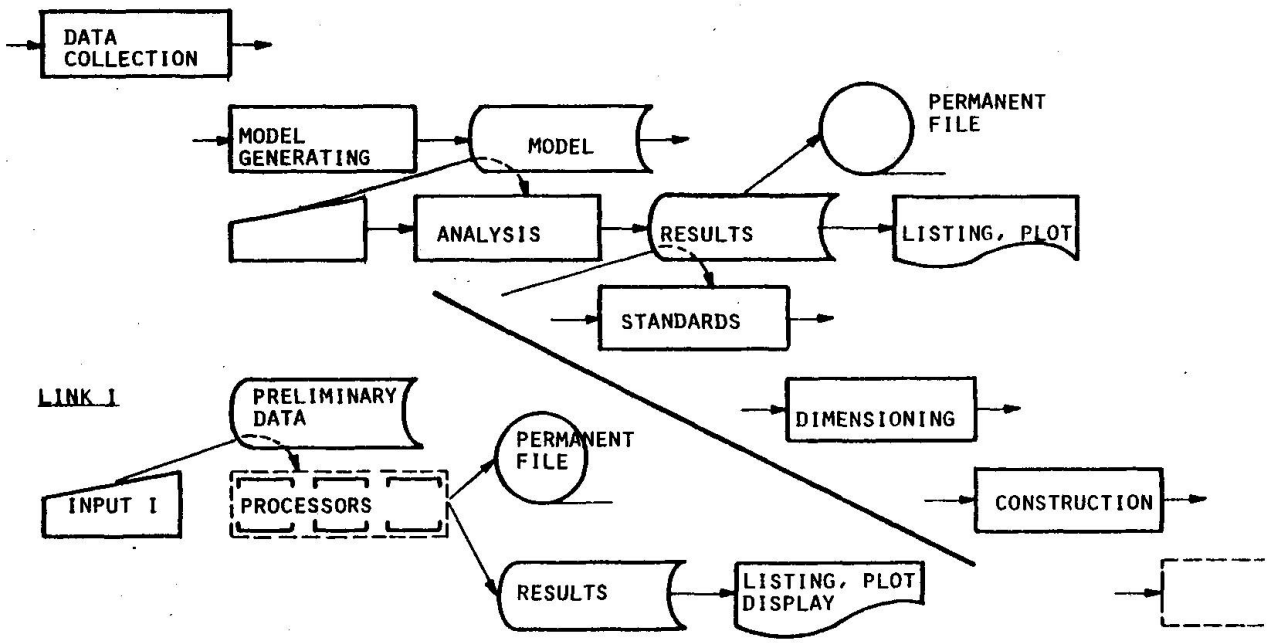


Fig. 1 Scheme of the Design Process

Software meant to automate parts of this chain of processes which are connected with each other to a certain degree could be called "process oriented software".

For every type of construction there is a different process because there are different aspects to the design e.g. of bridges or of soil structures.

Solutions for problems in different processes are often similar (i.e. use of the FEM in structural analysis). Software has been developed to solve special problems (i.e. mechanical or graphical). This type of software may be called "problem oriented software".

Take a matrix which contains horizontally the steps of the respective design processes and vertically the different constructions. CAD-Programs (CAD = Computed Aided Design) are an example of the former group (rows) and FE-Programs [2] are an example of the latter group (columns). The expression "design-oriented software" may be used to comprise both groups.

1.2 Hardware - Software - User Interface

Design-oriented software provides an interface between computers and engineers (fig. 2).

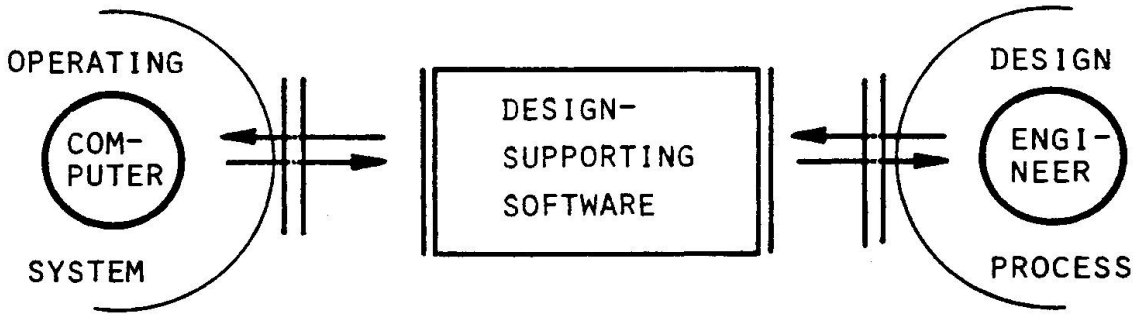


Fig. 2 Interfaces between Computer, Software and Engineer

Software has to be adaptable both

- to the computer and its operating system and
- to the engineering design practice.

These "interfaces" become important in the transition from software "made to measure", for one computer system and one firm, to "standard software", intended for many applications on different makes of computers.

The expression "software", used in this paper, means a complete set of information describing a program, comprising

- short description of the program,
- user manual,
- data processing manual,
- source code.

Hardware exists in many forms; it can be classified approximately into [3]:

- mainframe computers,
- minicomputers,
- desk-top-calculators or microcomputers,
- terminals (without any computing capabilities)

Computers of different sizes are increasingly linked together, i.e.

- terminals are used as peripherals for micro- or minicomputers or connected to commercial timesharing-systems via telephone lines,
- microcomputers serve as intelligent terminals to prepare data for larger computers,
- minicomputers and mainframers work together in computer networks (i.e. minis as front-end-processors).

The smaller (and cheaper) a computer, the greater the number of installations; design-supporting software consequently has to meet the needs of mini- and microcomputers in order to find an adequate market.

Similar to the hardware hierarchy, engineers can be classified according to their knowledge of data processing. The majority of structural engineers are strangers regarding the requirements of hardware and the methods and models of software. They need the help of "interpreters" once their involvement is beyond that of theoretical background.

Much rarer is the engineer who combines knowledge on software applications and structural design, the "finite element engineer". Therefore, software must be aimed at for the "normal" engineer in order to attain wide distribution.

1.3 Software Examples

In the following chapters two software examples will be used for illustration:

1. SET - a chain of programs for geotechnical problems [1]. SET contains components to
   - generate structural and system data (GENSET);
   - analyse FE-structures with non-linear material behaviour (NONSET);
   - calculate reinforced concrete members, i.e. tunnel linings, tied-back walls (CONSET);
   - analyse seepage and groundwater movement (SISET);
   - print and plot the results (PRINSET, PLOTSET);

   All components are linked via a common data base (fig. 3).

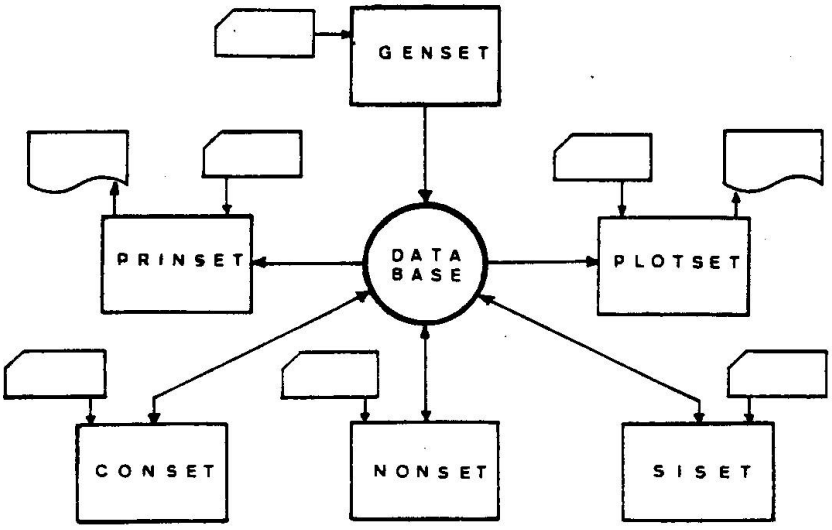


Fig. 3 Components of Program SET

2. HOPP - a program package for the structural engineer.

As a consequence of the wide distribution of desk-top calculators and of their growing performance, engineers require software for their run-of-the-mill problems; HOPP, a package in BASIC, fits this need. Standardization of input (interactive), output and interfaces for all components has led to its wide distribution.

Components exist for the analysis of RC beams, slabs, columns, foundations, plane frames etc. and perform structural analysis, superposition of member loads and dimensioning according to the standards.

2. STEPS OF SOFTWARE DEVELOPMENT

Software development comprises four stages from the problem definition (a description of the desired capabilities, the methods and the "boundary conditions" of a program) to the mature product:

- program design,
- coding and tests,
- pilot installation(s)
- marketing, maintenance and enhancement.

Each stage must be completed before the next one is started; omissions and carelessness in one stage lead to expensive corrections in the next or endanger the whole project.

In [4] a great number of topics to be considered has been summarized.

3. PROGRAM DESIGN

According to the German CAD standards [5], concepts must exist for

- input,
- output,
- solution methods and algorithms,
- program structures and data flow,

before coding can begin. The results of this first stage of software development should be fixed in provisional "User's Manuals" and "Data Processing Manuals".

3.1 Input

Input and output are the interface between design-supporting software and user. A user must not be asked to adapt to new input forms for every new program; a strange form of input can build a barrier against the introduction of new software. An interface between the input component and the rest of a program allows the adaption of the input component to a form the user is familiar with. Some firms do well with traditional input (with fixed columns) - especially, where large amounts of data have to be transcribed by data typists from form sheets to storage media. On the other hand, free format input using a problem oriented language [6] leads to ease of use on terminals. "Guided" input is useful for a dialog with

the computer if it is coupled with immediate validity and plausibility checks. This gives guidance to the engineer who does not constantly use the computer but may be superfluous for the experienced and constant user.

3.2 Output

The results of a calculation should be preserved in a storage file; this is advantageous for the following reasons:

- a subsequent program can use these values as input (fig. 1);
- on a permanent file, the results can be kept as long as they are needed;
- subsequent processes, i.e. superposition of load cases, can be started immediately;
- similar to many input forms, several forms of output can be selected (fixed or free formats, degree of condensation, printed or graphical output).

In Munich work is currently being done on a program which lets the user choose the contents and form of the printed output.

On the subject of print output, some obvious facts are mentioned here because they are often overlooked:

- To ensure easy transition between batch mode and dialogue, line numbers should be counted and the number of lines per page should be limited by a variable (which can be set according to the printer);
- every page should automatically be marked by a heading (project identification, date, page number etc.);
- print output should adhere to standard paper sizes (in Germany: DIN A4);
- empty lines waste paper without leading to greater clarity;
- every print output must be self-explanatory without the help of manuals etc.

3.3 Solution Processes and Algorithms

With the application of data processing in design practice, many new solution processes and algorithms have been developed. The practical engineer as program user is often not familiar with these methods; he must, however, check the results and take responsibility for them. Here we have a difficulty which we could meet by the following means:

- Full documentation of the theoretical background of a program, in such a way that the manual becomes a textbook complete from references to the actual implentation;
- introduction of an adviser between software and user. This could be an engineer familiar with the dp methods in a firm, a computer center or an institution concerned with marketing software.

If a new technical model or a solution algorithm is to be introduced, it must be checked for numerical stability, limits of applicability and practical reliability. Such research ought to occur before a design-supporting program is planned. Only tried and proved algorithms should be used for broad (and often uncritical) applications. On the other hand, new techniques are necessary. Their development must be supported if software development support is to be of long-term effect. The result of such research cannot be mature programs but algorithms, which are turned into programs for practical use in the next step (i.e. SAP IV). If the result of such research is a source program, then this source program should be published together with all necessary explanations.

Design-supporting programs refer to standards, recommendations etc. once they leave the area of classical mechanics. Standards, however, usually do not give an unequivocal answer for every detailed problem; interpretations are necessary. Engineers are confronted daily with interpretations; often they decide open questions by experience and engineering judgement.

Design-supporting software has to decide some of the questions in advance, which puts the responsibility on the program author; he has to fill the holes in the standards. In a way he sets a standard himself because possibly hundreds of users may trust his decision implicitly. This is another problem which must be solved.

Some recommendations are:

- design-supporting software development needs the intensive assistance of engineers versed in design practice;
- interpretations of standards must be documented and the standard-committees must be made aware of the problem;
- software engineers should participate in standard-committees in order to adapt standards to dp needs.

3.4 Program Structure and Data Flow

A program consists of routines; each routine should have a clearly defined task and a clearly defined data interface to the rest of the program. CAD standards [5] recommend a size not exceeding 200 statements per FORTRAN routine. It is therefore necessary to analyse the problem, to structure it into sub-problems, until each routine meets these requirements.

At the same time, the data flow between components of the program must be planned.

Interfaces can and should be defined, not only between input and output components and the rest of the program, but between large blocks of routines as well. For instance, the following program and data structure has served well for FE analysis of construction stages (for bridges and tunnels) [1]:

1. "System data" (nodal point coordinates, material parameters, element descriptions, loads and boundary conditions) are input, checked and stored on a permanent file ("original qualities").
2. The FE model representing a certain construction stage (or design alternative) is built up using pre-defined members; data concerning this "current structure" are stored in a working file.
3. The current structure is analysed; results are printed and stored on a working file.
4. If the user so decides, results are stored with the "original qualities" as "acquired qualities".

In this way, the user can model the construction sequence, analyse alternatives or control non-linear processes (i.e. system creep).

A number of programs can be linked to a chain if the external files are clearly defined.

In order to curb the proliferation of programs it seems to be absolutely necessary to design program components to be interchangeable between different program packages (i.e. FE-solutions or graphics packages).

Expensive interfaces become superfluous when data bases are standardized; in any case general recommendations are very much needed. With SET the author offers an initiative in this direction (see chapter 1.3).

4. CODING AND TEST

4.1 Programming Language Considerations

The choice of a programming language for engineering programs rests largely between FORTRAN and BASIC. Up to now, FORTRAN is the language generally recommended for technical programs because a standardized version (ANSI-FORTRAN) exists giving a high degree of portability. For smaller programs, desk-top calculators programmable in BASIC only, offer the widest market. The problem with BASIC is its lack of standardization. Each computer has its own dialect; this in not prohibitive however, because the popularity of desk-top calculators makes it worthwhile to adapt programs to the different dialects. Experience with the program package HOPP has shown that an adaption to each type of computer is necessary but that these alterations account for only a small percentage of the development effort.

4.2 Reduction of Software Errors

Coding means the translation of ideas (collected in the manuals) into statements in a programming language. In practice, however, more time is spent in testing. Hierarchial structures and clearly defined interfaces make it possible to code and test program segments independently, one at a time. By this method, one can be reasonably assured that an error (or an alteration) in one segment has no unforseeable repercussions in another.

Software errors may be defined as:

- errors in the source code,
- errors in the manuals (i.e. description of input),

Clearly, it is efficient to use routines already tried and proved in earlier projects because the effort for testing can be much reduced.

Testing can be made easier if the search for errors is planned in advance. Valuable tools are:

- clear structuring;
- options to print intermediate values ("trace"); it is helpful to print input and output parameters for routines and data transferred to or from files; it must be possible to pinpoint certain areas to reduce the bulk of data;
- redundant validity checks whereever possible; every rountine should "defend" itself against incorrect parameters.

Errors in the manuals are often caused by program alterations not noted in the manual. For this reason disciplined updating is essential.

Errors in the solution algorithms occur if cases are analysed not considered in the design stage. Here the solution process must be enhanced or these cases must be specifically excluded in input description and

by input checks.

5. PILOT INSTALLATIONS

Pilot installations following the implementation stage quickly detect:

- installation difficulties,
- errors not found by previous testing,
- practical cases not concidered in the development.

One of the most important sources of information for the program author is the user feed-back. Every recommendation to extend the scope of the program, every misinterpretation of the manuals, every question as to the theoretical background (or to the use of the program or to the meaning of the output) has to be analysed carefully and will usually lead to alternatives in code or manuals. Changes in large FE programs (i.e. ASKA, NASTRAN) are tested for one year by pilot installations in large and experienced firms before being released to the public.

Pilot installations involve a lot of human effort and are therefore expensive; these costs have to be planned into the budget.

6. MARKETING, MAINTENANCE AND ENHANCEMENTS

Only a few large firms have the means to develop design-supporting programs on their own; hardware firms have more or less stopped to develop application software. These days most software must be bought.

The customer is confronted by the following problems:

1. Orientation: Often the information concerning software is insufficient.
2. Choice: Reference lists of users probably provide the best judge of performance and reliability.
3. Installation: This can be particularly difficult with software designed for a special type of computer. Once again, references are helpful.
4. Training of staff: Effort here can be reduced considerably if standards in manuals [5], in input form [6] and in output are generally accepted.
5. Maintenance: Correction of errors needs effort in direct relation to the number of applications. Software has to be adapted constantly to changing standards, techniques or solution processes, if it is not to become obsolete. If software is accepted by the engineers of a firm, then the number of applications grows and also the demands for expansion of the original scope of a program. Who is to maintain and evolve a program? To do it in one's own firm, the user needs dp staff who are willing and able to analyse the program. Otherwise, maintenance should be done by the vendor or the author.
6. Training and advice: New software often contains new algorithms, different interpretations of standards or new theoretical background. Users must familiarize themselves with these novelties and decide whether a new program is applicable to their problems. In this they need help.

These questions demand answers from the vendors:

1. New software products must be published. Mere descriptions are not enough; examples of practical applications awaken interest. Software lists, short information, publications in engineering journals are possibilities. A lack of suitable marketing organizations is obvious.
2. Sample runs of data prepared by the prospective user are often expensive; on the other hand they are a welcome addition to the testing process.
3. Installation effort is reduced considerably if
   - ANSI-FORTRAN is used,
   - the data processing manual contains clear information concerning the installation procedures.
4. User's access to new software can be simplified by:
   - detailed examples in the user manual (first program applications usually take their pattern from examples);
   - adaption of input to a well-known form, i.e. by use of form-sheets familiar to the staff.
5. Software always contains errors; every new installation brings new maintenance problems. Maintenance and enhancement must be guaranteed. This stage of development never stops once a program is on the market; product without maintenance is soon obsolete.
6. The author of software acquires new knowledge in the process. As a natural consequence he has to give advice and publish his new knowledge. On the other hand, councelling by phone can be time consuming, especially if the program is widely used. A requirement becomes necessary for somebody who can answer standard questions and give normal advice to customers while retaining valuable hints for transmission to maintenance staff. This person should ideally be with the marketing organization.

These activities are quite time consuming and expensive. They increase with the success (i.e. distribution) of software. The development including the pilot installation accounts for about 25% of the costs, the rest is spread over the above mentioned points. Approximatively 1 - 3 years time lag occurs between the end of phase 2 (coding and test) and the first sales.

7. CONCLUSIONS

As mentioned before, design-supporting software is never finished; it needs constant maintenance and constant adaptions to changing surroundings. If a program is funded in some way, gains from sales etc. should be fed back into maintenance and expansion.

Software engineering is a young disclipine certain to play a key role in future scientific and technical prograss. It should be supported and helped to stand on its own feet. On no account, however, should subsidizing public agencies shackle the free development by unnecessary restrictions.

ACKNOWLEDGEMENT

The author wishes to thank Dipl.-Ing. K. Axhausen, member of the Group on Electronic Computation in Structural Engineering of the TU Munich, for his collaboration on this paper.

REFERENCES

1. WERNER, H., AXHAUSEN, K., KATZ, C.: Programmaufbau und Datenstrukturen in entwurfsunterstützenden Programmketten. Tagungsbericht: Finite Elemente in der Baupraxis, Hannover, 1978
2. DUNDER, V.F.: Finite Element Programs or Software Engineering Library? Proceeding of the 4th Intern. Seminar on: Computational Aspects of the Finite Element Method, Long Beach, 1977
3. NOPPEN, R.: Technische Datenverarbeitung bei der Planung und Fertigung industrieller Erzeugnisse. Informatik-Fachberichte Nr. 11: Methoden der Informatik für Rechnerunterstütztes Entwerfen und Konstruieren, Springer-Verlag, Berlin, 1977
4. BLAND, R.L.: Engineers and Computers-Interaction or Reaction. 6th National ASCE-Conference on Electronic Computation, Atlanta, 1974
5. LANG-LENDORFF, G.: CAD-Guidelines, CAD-Bericht KFK-CAD 6. Gesellschaft für Kernforschung, Karlsruhe, 1976
6. AHN, M., BÖCKELER, K.H., HAAS, W.: Eingabekonventionen für CAD-Programme. CAD-Bericht, KFK-CAD 39. Gesellschaft für Kernforschung, Karlsruhe, 1977

Leere Seite
Blank page
Page vide

IABSE
A I P C
I V B H

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Aspects on Input and Output for Finite Element Programs

Considérations sur l'input et l'output de programmes avec des éléments finis

Aspekte des In-und Output von Programmen mit finiten Elementen

H. TAGNFORS
Res. Eng., Cent. for Comp. Calc.
Chalmers Univ. of Technology
Gothenburg, Sweden

N.-E. WIBERG
Professor, Dept. of Struc. Mech.
Chalmers Univ. of Technology
Gothenburg, Sweden

Summary

For an engineer in a structural design office one of the main requirements for a finite element program is that it must be simple to use in order to minimize the total work in the design process. In order to diminish the gap between man and machine, a general finite element program SITU has been developed. The program, suited for the small and medium sized office has a simple and logical input, logical program structure and an output with high readability.

Résumé

Le projeteur d'un bureau d'ingénieurs a besoin d'un programme avec des éléments finis qui soit simple à l'emploi afin de réduire au minimum le travail total dans l'étude du projet. Le programme SITU a été développé dans ce sens et particulièrement pour les bureaux d'ingénieurs moyen et petits: l'input est simple et logique, la structure du programme est logique et l'output est facilement lisible.

Zusammenfassung

Der Entwerfer eines Ingenieurbüros braucht ein Komputerprogramm mit finiten Elementen, welches so einfach sein soll, dass die ganze Arbeit in der Entwurfsphase am geringsten ist.
Das Programm SITU wurde zu diesem Zweck besonders für kleinere und mittlere Ingenieurbüros entwickelt; der Input ist einfach und logisch, die Programmstruktur ist logisch und der Output ist lesbar und verständlich.

1. INTRODUCTION

The finite element method has become a powerful tool for the analysis of complex structural engineering problems. The input and output parts of the computer programs are often not as developed as the calculation part. For an engineer in a structural design office one of the main requirements is that the program must be simple to use, in order to minimize the total work in the design process. This is of utmost importance for all the engineers who do not use the computer every day. The discussion in this paper concerns mainly a small office, with a small equipment: a non-graphic terminal connected to a large computer.

The paper will discuss the use of finite element programs in the design process and different requirements including aspects on safety for the preparation of input data. The need for effective solution techniques, for equation systems and non-linear time dependent problems, is stated. Different ways of result presentation are compared and aspects on readability are laid.

In order to diminish the gap between man and machine a general finite element program, SITU, which is SImple To Use is under development. The program, suited for the small and medium sized office, has a simple and logical input, logical program structure, and an output with high readability. As a demonstation problem the analysis of a wood diaphragm, where the cover is attached to the frame by nonlinear nails, is presented.

2. GENERAL ASPECTS ON USE OF FEM-PROGRAMS

2.1 Computational cost

The aim of an industrial process is to minimize the total cost for a product of a required quality. In structural analysis, extensive use is made of the finite element method (FEM), [1], especially if complicated shape, properties and loading are to be simulated realistically. The finite element analysis may be made in three main

steps:

- Setting up the calculation model and the relevant data
- Numerical calculation
- Presentation and evaluation of results

In practice, the first and third steps usually needs much more effort than the second step. Fig . 1 shows a general estimation of different costs in connection with computer calculations.

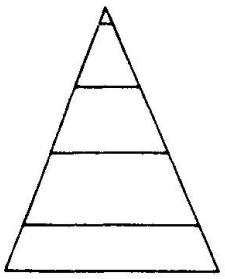


Fig. 1 Cost triangle for FEM-calculations

As seen in Fig, 1 the cost for input and output may be 10 times the direct computer cost. Errors in input data gives large total costs. Reading the manual for the first time may cost another ten times, depending of the type of program . The numerical calculations need very little action by the user. The input and output steps rely heavily on the effort of the analyst to set up, handle and evaluate large data sets. A logical structure and a logical input and readable output and well written manuals may diminish the user cost to a great extent.

In addition we have costs for the equipment at the office. In the discussions below the cost for investments are very limited, in order to suit small construction units. The equipment is a non graphic terminal. Large lists of results are supposed to be mailed.

2.2 The engineer and the computer

The engineers main concern is to solve his particular problems in the easiest and quickest manner, preferably at as low cost as possible. In order to simplify the use of a computer some basic

requirements must be fullfilled:

- Physically easy to use (terminal on the desk)
- Simple instructions to the computer
- Special and general programs, easy to use

A very important matter is to make the contact with the computer very easy by having terminals on the desk of each engineer. At the terminal the engineer can create a job and submit it to the computer. Limited amount of result can be studied. Large output can either be taken to a minicomputer-terminal at the office or can be mailed from the computer centre.

The necessary instructions to the computer for running a program must be extremely simple with for example direct questions and answers which need a minimum of knowledge of the job control language of the computer. More people in practice would use computers in practice to a greater extent if the instructions to be given were more user oriented, to suit people who do not use the computer every day.

A program library ought to cover two main types of programs, special programs and general programs. A special program, which may calculate a frame, a plate or a 3D-body, must be so simple to run, that any engineer directly can run the program from a terminal and get the result almost immediately with minimal possibility of all types of errors. A general program is supposed for complicated problems e.g. coupled structures, (different element types mixed together), non-linear problems and time dependent problems.

The program package CHALMFEM, [2] can to a certain extent meet the requirements above, Fig. 2. After identification (/ID) conversive instructions are given to the operative system concerning used FEM-program, files on secondary memory etc. An input file for the program is generated. The chosen FEM-program reads data from the input file and produces results on a result file. By inspection of the result file, changes of the structure can be decided, and performed by changing the input file.

/ID

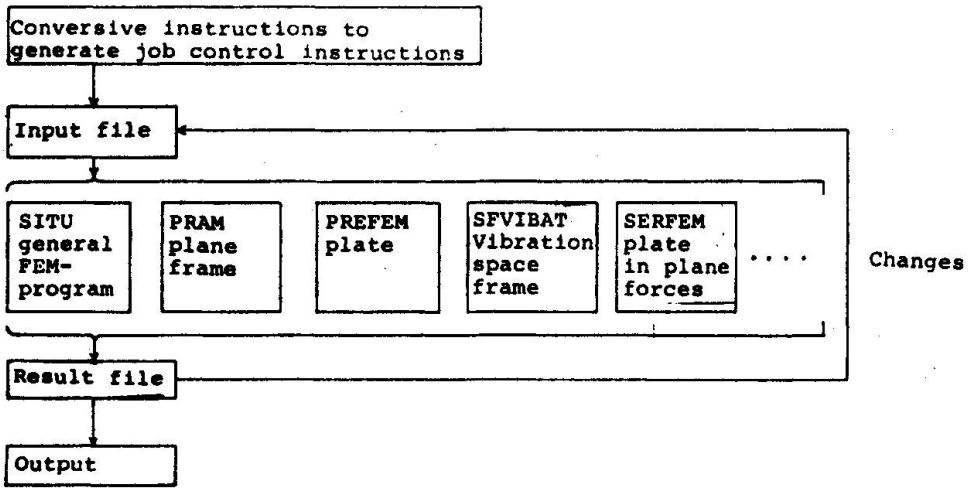


Fig. 2 Calculation with programs in CHALMFEM

2.3 Demands on a general program

From the engineers point of view he want to describe the problem in a simple way and obtain a correct result which contain only what he is interested of. To meet these demands some requirements must be fullfilled:

- A language for giving input data, close to the engineers
- Safe algorithms for the calculations
- A variety of output selections.

The structure of a general FEM-program must be logical and divided into a number of well defined modules.

In general; input data, calculation and output must be made so that a "safe" result is obtained. The old manual way of writing input data which for example use fixed format, node- and element numbering as input, steering of the calculation by a number in a special input position, is time consuming, tedious and error-prone. However, it is possible to construct a simple language for giving input data even for the complicated problems.

In nonlinear and time dependent problems there are a need for effective routines in order to minimize the computer cost and to obtain

a numerically safe solution. Such routines are time-stepping routines which gives an automatic selection of time steps based on a local error estimation [3]. Other routines are direct and iterative solution routines for equation systems [4]. Use of substructuring will simplify input, calculation and output.

A simple and readable output with possibilities of output selection is also a very important part in a FEM-program. A list of results, referred to the structure by node numbers and variable numbers are difficult to read and errors might be undetected. Plotted diagrams are of course very useful, but has to be complemented by the results presented at the terminal or the line printer.

A general program SITU [5], under development, has a very simple and logical input in the language of an engineer.

3. SITU, A FEM-PROGRAM SIMPLE TO USE

3.1 The program structure in SITU

The total structure is built by plane substructures S, see Fig. 3.

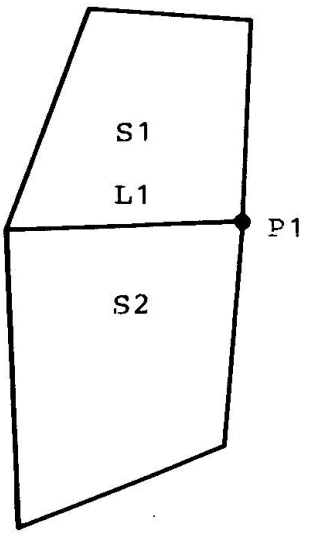


Fig. 3 Structure built by substructures

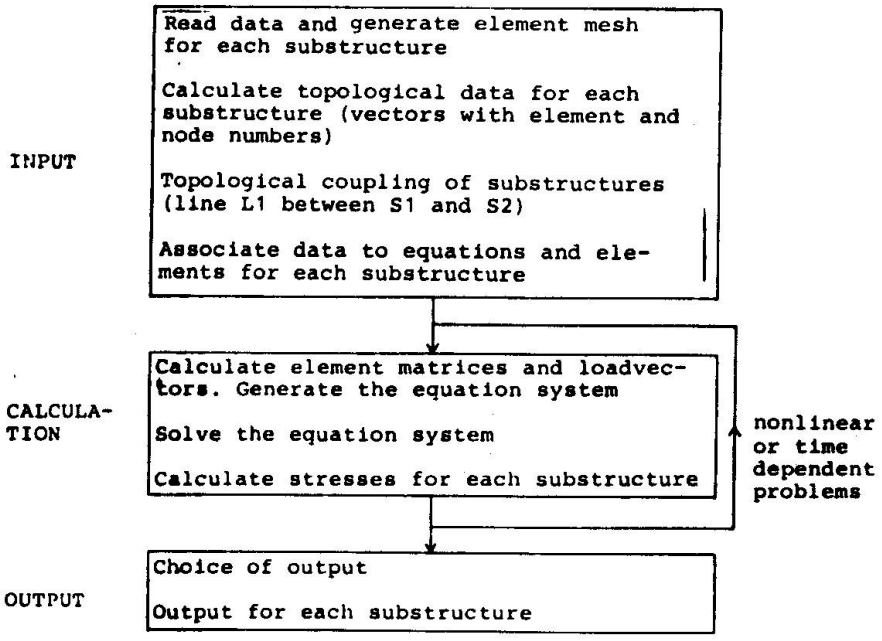


Fig. 4 Program structure in SITU

Each substructure is geometrically defined by points (P), lines (L) and areas (A). Loads, boundary conditions and material properties etc are defined and assigned to the structure by given geometrical properties. Positions where results are wanted are referred to by given names of geometrical properties. By this method the user does not have to concern with node numbers, variable numbers etc, because these numberings are only used in the program and does not appear in input and output. The main program structure in SITU is given in Fig. 4. Input and output are described below in an example.

4. NUMERICAL CALCULATION OF A WOOD-DIAPHRAGM

4.1 Problem description

The stresses and displacements in the wall in Fig. 5a are to be calculated by FEM. The wall is a wood-diaphragm consisting of a frame with a nailed sheet, see [6]. From the calculation point of view, the wall is separated into three substructures, frame, sheet and nails, Fig 5b.

The frame is divided into beam elements, the sheet into plane elements (8 node element based on a biquadratic displacement field). The frame and the sheet are coupled in a number of nodes by nail elements. The nails are concentrated to these points. The material in the frame and sheets is linear elastic. The nail has a nonlinear force-displacement (T, t) relationship, given by a polygon, see input data. This structure gives a very complex description

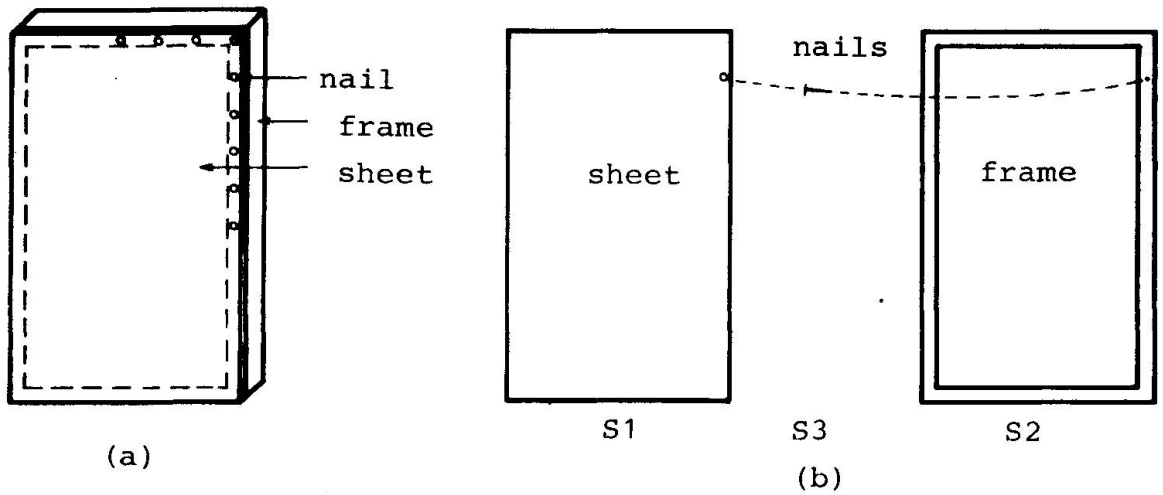


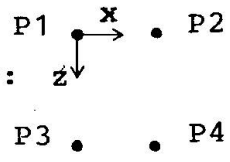
Fig. 5 Wood-diaphragm

4.2 Input data

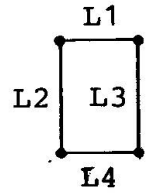
The input data for the example in Fig. 5 will be described below. The element mesh is here chosen very coarse in order to make the description small in size. The needed input is given below together with some explaining figures.

```
SHEET + NAILS + FRAME
SUBSTRUCTURE S1     SHEET
POINTS
P1 0. 0
P2 2000. 0.
P3 0 3000
P4 2000 3000
LINES L1 P1 P2
L2 P1 P3
L3 P2 P4
L4 P3 P4
AREA A1 L2 L4 L3 L1
NET TYPE 21 2 2
ELEMENT 21
DEFINITION MATERIAL AREA MA1 A
2000 0 13
DEFINITION LOAD POINT PL1 A
1000
ASSIGN MA1 A1
ASSIGN PL1 P1 X
PRINT DISPLACEMENTS

SUBSTRUCTURE S2     NAILS
POINTS ACCORDING S1
LINES ACCORDING S1
NET TYPE 1 5 ELEMENTS ON LINE
ELEMENT 1 X Z X1 Z1
DEFINITION MATERIAL POINT MA1 B
0. 0.
0.1 100
0.8 200
2 300
3.8 400
7 475
14 500
14 0
20 0
ASSIGN MA1 L1 L2 L3 L4
PRINT NAIL FORCES

SUBSTRUCTURE S3     FRAME
POINTS ACCORDING S1
LINES ACCORDING S1
NET TYPE 11  4 ELEMENTS ON LINE
ELEMENT 11 X1 Z1 Y1 X1 Z1 Y1
DEFINITION MATERIAL LINE MA1 A
10000 4275 721400
DEFINITION MATERIAL LINE MA2 A
10000 2025 341720
DEFINITION BOUNDARY POINT BO1 A
0
ASSIGN MA1 L1 L4
ASSIGN MA2 L2 L3
ASSIGN BO1 P3 P4 X1 Z1
PRINT INPUT DATA

STRUCTURE S4
ADD S1 S2 S3
EQUAL BY NAME OF LINES
SOLVE
LOAD 1. INCREMENT 1.
END OF DATA
```

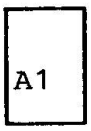
Point geometry:



Line definition:

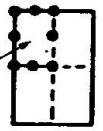


Area definition:



Net definition:

Element:

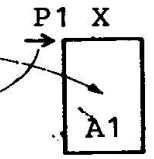


Material:

MA1 = (E, ν, t)
(2000, 0, 13)

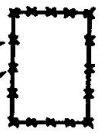
PL1 = (P)
(1000)

ASSIGN:



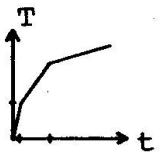
Net definition:

Element:



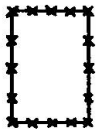
Material:

MA1 =



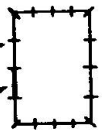
ASSIGN:

MA1



Net definition:

Element:



Material:

MA1 = (E,A,I)
(1000, 4275,
721400)

MA2 = (E,A,I)
(1000,2025,
341720)

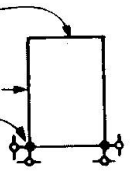
BO1 = (p)
(0)

ASSIGN:

MA1

MA2

BO1



COUPLING OF SUBSTRUCTURES

NUMERICAL CALCULATION

4.3 Input data check

Results from the input analysis are shown in matrix as a receipt. It is given for each substructure when requested. Fig. 8 shows the material properties and boundary conditions for the frame (substructure 3).

MATERIAL PROPERTIES FOR ELEMENTS

P1	MA1	MA1	MA1	MA1	P2
MA2					MA2
MA2					MA2
MA2					MA2
MA2 P3	MA1	MA1	MA1	MA1	MA2 P4

MATERIAL	E	A	I
MA1	10000.	4275.	721400.
MA2	10000.	2025.	341720.

(a)

PRESCRIBED DISPLACEMENTS AT THE NODES

PRESCRIBED DISPLACEMENTS IN X-DIRECTION

P1				P2
.	.	.	.	.
.				.
.				.
P3 0	.	.	.	P4 0

PRESCRIBED DISPLACEMENTS IN Z-DIRECTION

P1				P2
.	.	.	.	.
.				.
.				.
P3 0	.	.	.	P4 0

(b)

Fig. 8 Input data receipt for the frame (substructure 3)
(a) Material properties
(b) Boundary conditions

4.4 Output from the numerical calculation

As an example of the output from the numerical calculations the displacements at the nodes of the sheet (substructure 1) are given in Fig. 9a, and the nail forces (substructure 2) are shown in Fig. 9b.

DISPLACEMENTS AT THE NODES

DISPLACEMENTS IN X-DIRECTION
MULTIPLY PRINTED VALUES BY 10 RAISED TO -3

P1 2475	2398	2332	2321	P2 2319
1852		1867		1859
1364	1359	1353	1359	1363
863		863		859
P3 374	377	378	379	P4 377

NAIL FORCES

MULTIPLY PRINTED VALUES BY 10 RAISED TO -3

P1 -305015	-65986	-15757	72125	P2 323505
-409697				358136
-387989				389872
-344986				470977
P3 -155991	135957	263397	397993	P4 695672

Fig. 9 (a) Displacements at the nodes of the sheet
(b) Nail forces

5 REFERENCES

[1] O.C. Zienkiewicz: "The finite element method in engineering science", Mc Grow-Hill, London 1971

[2] "CHALMFEM: Computer program for FEM-analysis of static, dynamic and transient problems, developed at Chalmers University of Technology", Computer Centre in Gothenburg, Gothenburg 1977

[3] K Runesson, N-E Wiberg and A Wolfbrant: "A new Rosenbrock-type method applied to a finite element-discretized nonlinear diffusion-connection equation", MAFELAP 1978, The matematics of finite elements and applications, 18-21 April 1978, Brunel 1978

[4] R Glemberg and N-E Wiberg: "Iterative solution of finite element equation", Chalmers University of Technology, Department of Structural Mechanics, to be published, Gothenburg 1978

[5] H Tägnfors and N-E Wiberg: "SITU, A general finite element program, SImple To Use". Chalmers University of Technology, Department of Structural Mechanics, to be published, Gothenburg 1978

[6] H Tägnfors and N-E Wiberg: "Wood-diaphragms analysed by the finite element method", (In swedish), Chalmers University of Technology, Department of Structural Mechanics, to be published, Gothenburg 1978.

IABSE
A I P C
I V B H

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Optimization of Structures Using Decomposition

Optimisation des structures au moyen de la méthode de décomposition

Optimierung von Tragwerken mittels Dekompositionsmethode

Z.K. LESNIAK

Dr. Professor
Technological University
Bialystok, Poland

Summary

The application of the decomposition method to the optimization of structural systems consists in dividing the system into parts and coordination of them to assure adequacy to the reality. After general formulation of optimization the conditions for applying decomposition have been discussed. A case study - the optimization of the steel structure of industrial sheds - illustrates the successful application of the decomposition method. The computer system OSY which performs this optimization, based on the decomposition princimple, has been mentioned.

Résumé

L'application de la méthode de décomposition pour l'optimisation des systèmes de construction consiste en une division du système en parties, en leur coordination pour assurer leur concordance avec la réalité. Après une formulation générale de l'optimisation, on a discuté les conditions de l'application de la décomposition. L'optimisation de la structure métallique de toitures en shed est presentée comme un exemple de l'application de la méthode de décomposition. On mentionne aussi le système informatique OSY avec lequel cette optimisation, basée en principe de la décomposition, est faite.

Zusammenfassung

Die Anwendung der Dekomposition zu der Optimierung von Tragwerksystemen besteht in der Teilung des Systems in Subsystem und ihre Koordination zwecks der Beibehaltung der Verträglichkeit des Systems mit der Realität. Nach allgemeiner Formulierung der Optimierung werden Voraussetzungen für die Anwedung der Dekomposition erörtert. Eine Fallstudie - Optimierung der Stahlkonstruktion von Industriehallen - zeigt eine erfolgreiche Anwedung der Dekomposition. Das Computer-System OSY, das diese Optimierung auf der Grundlage des Dekompositionsprinzips vornimmit, wird erwähnt.

1. INTRODUCTION

Let us start by general formulation of the optimization problem.
Given:
1/ a decision space X' , consisting of the elements (decisions) x: $x \in X'$;
2/ a set of permissible decisions X, defined as

$$X' \supset X = \{x : g(x) \leqslant 0,\ h(x) = 0\}$$

3/ a functional f optimum criterion , which maps the decision space X' into the space of real numbers R, i.e.

$$f : X' \to R$$

The task of the optimization is to find such an element

$\hat{x} \in X$ that $(\forall x \in X)\ f(\hat{x}) \leqslant f(x)$

In this way we can formulate many optimization problems, in particular the problem of optimization of systems. It does not mean, however, that we are able to solve every mathematically formulated optimization problem.

The question arises, how to solve large problems occuring in practice? The concept of solving such problems is to apply decomposition, which consists in dividing the problem into parts and solving them parallelly and independently by assuring the existing connections between these parts by so called coordination.

2. DECOMPOSITION

Large optimization problems occuring in practice are, as a rule, the problems of the optimization of systems. The main feature of systems is that they consist of distinct components, called subsystems, and - because of the conflict of interests arising between the subsystems and the systems as a whole - the optimum of the system is not the sum of the optima of subsystems. The partitioning of the large optimization problems, which is necessary for applying decomposition, goes on in natural way in the case of the optimization of systems, because the system is composed of distinct parts, i.e. subsystems.

Let us now formulate the decomposition mathematically. For simplification let us consider the optimization problem in the Euclidian space. The primary problem is:

$$\inf_{x \in X} f(x), \qquad X \subset E \qquad (1)$$

To solve the problem (1) by decomposition we divide the vector x into 2 parts (y,z), where y are coordination variables and z are the decision variables of the decomposed parts of the whole problem. The decomposed problem can be then formulated as follows [1] :

$$(2)\quad \inf_{x=(y,z)\in X} f(x) = \inf_{\substack{y\in Y\\ z\in Z\\ (Y,Z\subset X)}} f(y,z) = \inf_{y\in Y} \varphi\left[p_1(y,z^1), \ldots, p_N(y,z^N)\right]_{z^1\in Z^1(y),\ \ldots,\ z^N\in Z^N(y)}$$

where: $z^1, z^2, \ldots, z^N$ are separate parts of the vector z,
$Z^i(y)$ for $i = 1,2,\ldots,N$, is the set of permissible vectors z^i depending on y,
f is the objective function of the not decomposed problem,
φ is the global objective function of the decomposed problem,
p_i, for $i = 1,2,\ldots, N$, are the subsystem-objective functions.

If certain conditions are fullfilled, the solution of the decomposed problem (2) is at the same time solution of the primary problem (1).

Let us now discuss the conditions which are to be fullfilled in order to apply the decomposition principle. Let us assume that there is a set X' (decision space) of the discrete decision variables which is decomposed into 2 parts. In order to apply the decomposition, the objective function must be separable with respect to decomposed parts so that elements of one part of the set must not be included into the objective function of the second part. Some of the restrictions which are imposed upon the decision variables of the first decomposed part concern also certain variables of the second part. These decision variables which have common restrictions will be treated as the coordination variables. Let us present it in the matrix form (fig. 1). Let us assume that objective function is additive, which is the premise of the decomposition of the problem. Let us consider the restriction matrix with 1 lines, each of them representing an restriction R_i ($i = 1,2,\ldots,1$) imposed on decision variables. When the restriction matrix is completely filled up (fig. 1a), any decomposition is impossible. The problem must be then treated as one inseparable whole. The opposite extreme case arises, when the restriction matrix has the block-diagonal structure (fig. 1b). In that case one has not to do with one general problem but with separate problems which can be solved independently.

In the case of systems there always exists partly overlapping of the restrictions relating several subsystems (fig. 1c). These decision variables which occur in more than one restriction groups, let us call them y, do the coordinating and controlling with regard to the decomposed parts of the problem. The remaining decision variables of the decomposed parts will be called z.

In order to solve the decomposed problem the method of the two-stages-optimization (parametric optimization) can be sucesfully applied [1]. It consists in optimizing in two levels. In the lower level a multiple optimization of the decomposed parts is done with regard to their decision variables for consecutive values of the coordination variables which are assigned in the upper level and - in the lower level - treated as paramaters. The results of the lower level optimization ($\hat{z}(y)$) are availed in the upper (coordination) level for searching optimum values

of the coordination variables y. As optimization procedures the searching methods are used in both levels.

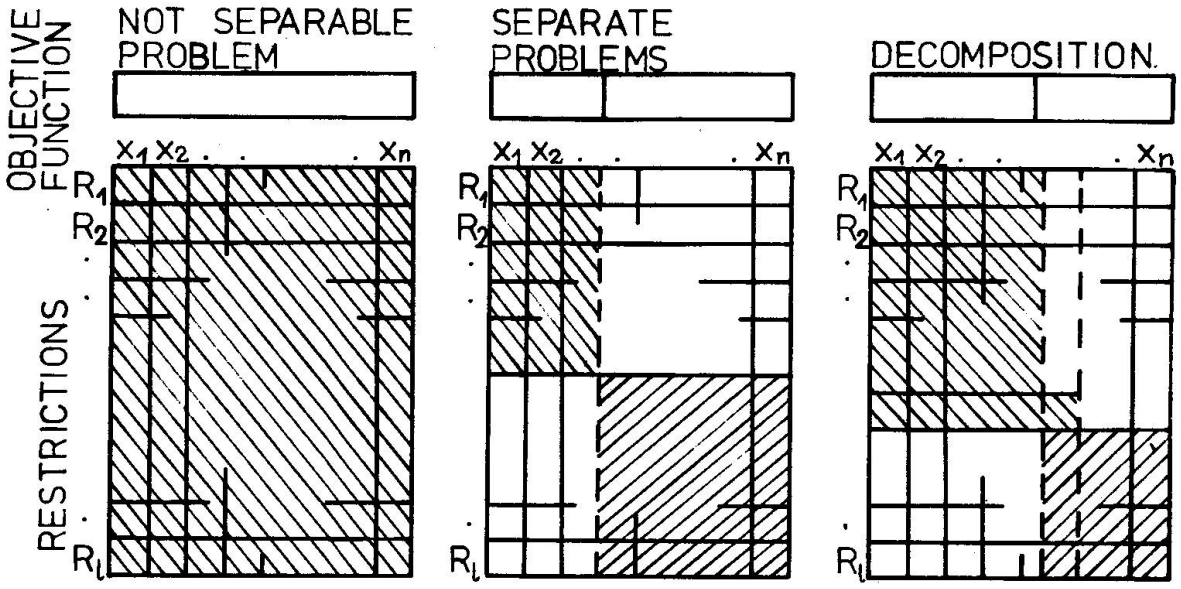


Fig. 1 Matrix representation of optimization problems

The condition for applying the two-level-optimization is that in the upper (coordination) level the restrictions imposed upon the coordination variables y must not include the variables z. In the lower level, i.e. in the decomposed parts, the restrictions can concern as well the variables z and y.

3. CASE STUDY

An example of the application of the decomposition to the optimization of a system is the optimization of the steel structure of industrial sheds (fig. 2).

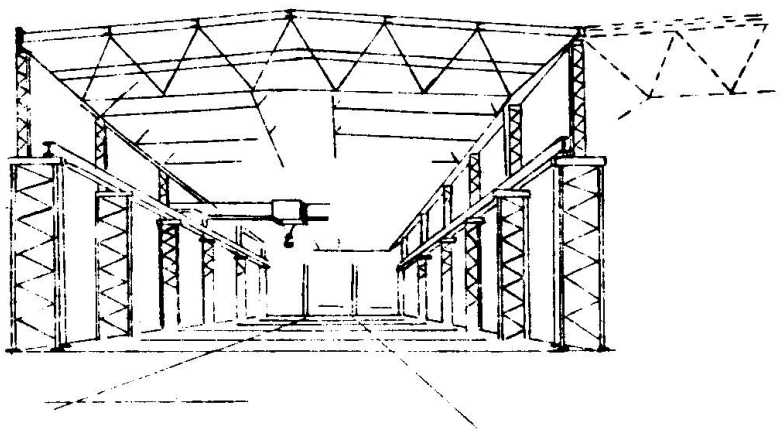


Fig. 2 Steel structure of an industrial shed

The structure of an industrial shed is a system which consists of a number of structural elements (subsystems) , see fig. 3. If each element is optimized separately this can lead to contradiction from the view point of optimum of the system as a whole. E.g. the optimum height of the cross-section of the gantry beam, found for the beam as a separate structure, can lead to the enlarging of the cubage of the shed and to the greater use of steel in higher colums. The optimization of a system cannot be therefore done on just assembling optimum solutions of the subsystems.

For solving the problem of the optimization of the structure of industrial sheds the author applied the decomposition method [2 - 6] .

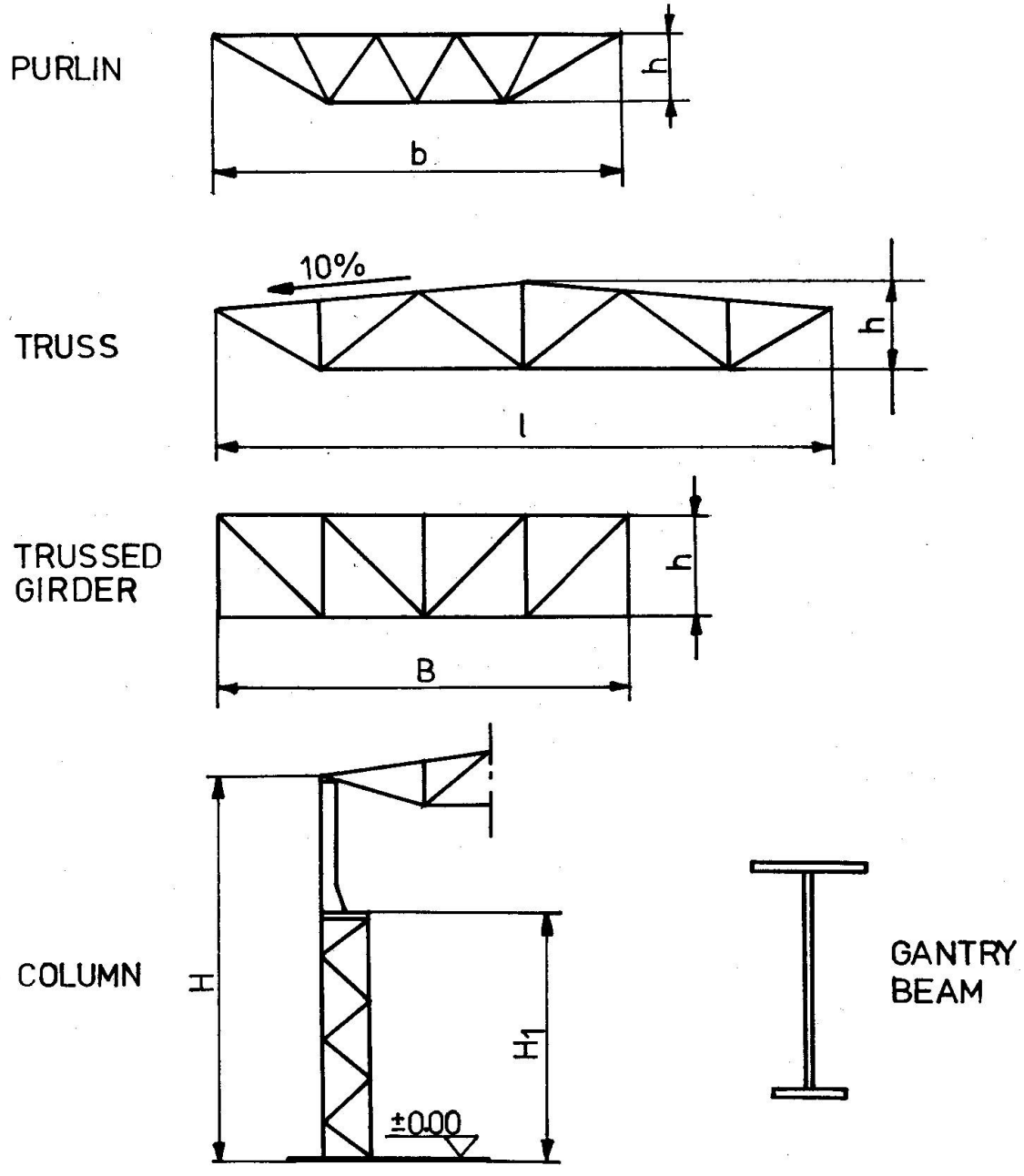


Fig. 3 Elements of the steel structure of an industrial shed

As the decomposed parts following subsystems were separated:

1. purlin,
2. truss,
3. trussed girder,
4. gantry beam,
5. column.

The optimum criterion was the cost of the carrying structure of the shed, objective function was therefore the sum of the execution cost of the subsystems. This function was separable with respect to the subsystems. The decomposition was therefore possible.

In the restrictions three decision variables refer to more than one subsystem. These variables stand for coordination variables. The mathematical model of the optimization problem of the shed structure - after decomposition - is:

$$\inf_{\substack{x\in X\\ y\in Y}} Q(x,y) = \inf_{y\in Y} \sum_{i=1}^{n} \inf_{x^i \in X^i(y)} Q_i(x^i,y) \qquad (3)$$

where: Q - the objective function of the whole system,
Q_i - the objective functions of i-subsystem,
x - vector of all subsystem-decision variables,
x^i - vector of the decision variables in the i-subsystem,
y - vector of the coordination decision variables,
X - permissible set of x-vectors,
Y - permissible set of y-vectors,
$X^i(y)$ - permissible set of x^i-vectors. This set is a function a function of y-vector, $\left(\bigcup_{i=1}^{n} X^i = X\right)$.

The substance of the formula (3) can be expressed in words as follows: The constrained infimum of the vectors x and y, is equal - after decomposition - to the constrained infimum (with regard to the coordination variables y) of the sum of the constraines infima of the subsystem objective functions Q_i, defined respectively for their decision variables x^i and suitable coordination variables y.

In the decomposed problems of finding infima of the subsystem--objective functions Q_i the decision variables are parameters. The decision space of the optimization structure of a shed is rather a general kind. It was not possible to reduce it to the Euclidian space, because several decision variables were of topological kind (e.g. truss type, profile type) and they could not be represented by numbers. This fact limited substantially the amount of suitable optimization procedures. To solve the problem of the optimization of the shed structure the exhaustive enumeration was mainly applied.

The problem of the optimization of the steel structure of one aisle industrial shed is the 33 - dimensional problem. Using the decomposition the dimension of the problem was reduced to the threedimensional coordination problem and 5 parallelly solved decomposed problems from 6 to 7 dimensions.

A computer system called OSY, developed by the author and his team, was based on the above presented decomposition principle. This system performs the optimization of the steel structure of industrial sheds. The system OSY was implemented on Polish ODRA and British ICL computers.

REFERENCES

1. FINDEISEN, W.: Wielopoziomowe układy sterowania PWN, Warszawa, 1974, 193 - 212 (in Polish).
2. LESNIAK, Z.K.: Optimization of the steel structure of industrial sheds, lecture held on November 13, 1970 in Warsaw (not published).
3. LESNIAK, Z.K.: Optymalizacja systemu budownictwa stalowych hal przemysłowych, II Conference on Applications of Computer Science, Krynica, 1971, 321-326 (in Polish).
4. LESNIAK, Z.K.: Optimierung der Stahlkonstruktion von Industriehallen, Wiss. Zeitschr. der Techn. Universität Dresden, vol. 21, no. 5, 1972, 957-958 (in German).
5. LESNIAK, Z.K.: Optimierung der Stahlkonstruktion von Industriehallen, Der Stahlbau, vol. 43, no. 3, March 1974, 93-94 (in German).
6. LESNIAK, Z.K.: Some Practical Applications of Structural Optimization, Optimization in Structural Design, IUTAM - Symposium, Warsaw 1973, Springer-Verlag, Berlin - Heidelberg - New York, 1975, 563 - 569.

Leere Seite
Blank page
Page vide

IABSE
AIPC
IVBH

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Tools for Computer-Aided Design (CAD)

Outils pour le projet à l'aide de l'ordinateur (CAD)

Mitteln für Computergestützte Berechnung (CAD)

G. MIANI

Dr., Tech. Comp. Software Dep.
ENEL - Ente Nationale Energia Elettrica
Milan, Italy

Summary

Provided that a great number of computer programs for automatic analysis of structures is available it is a worthwile attempt to look for tools which enable the civil engineer to overcome two critical steps in the use of these programs: the generation of an analytic, computer oriented, description of the physical model and the transformation of the results of computation in a synthetic, man-oriented form. The paper is devoted to enable structural engineers to get a picture of how they could be helped by the use of software for computer-aided design, and to give a glance, at the same time, to the problems involved in the design of this kind of software.

Résumé

Aujourd'hui un grand nombre de programmes peuvent être utilisés pour le calcul automatique des structures; par conséquent l'ingénieur civil est très intéressé d'avoir à sa disposition des outils qui lui permettent de surmonter deux phases critiques pour l'emploi des programmes de calcul: la génération d'une description analytique, "computer-oriented", du modèle physique et la transformation des résultats du calcul dans une forme synthétique et facilement interprétable par l'utilisateur. Ce rapport passe en revue les différents moyens par lesquels un logiciel de CAD peut aider un ingénieur civil et, en même temps, des problèmes qu'on doit résoudre pour l'établissement de cette sorte de logiciel.

Zusammenfassung

Da eine grosse Anzahl von Computer-programmen für die automatische Berechnung von Tragwerken zur Verfügung steht, ist es der Mühe wert, nach Mitteln zu suchen, die es dem Bauingenieur ermöglichen, zwei kritische Phasen bei der Benutzung dieser Programme zu überwinden: die Schaffung einer analytischen Computer-orientierten Beschreibung des physikalischen Modells und die Transformation der Resultate der Berechnung in eine synthetische, Gebraucher-orientierte Form. Dieser Artikel soll den Bauingenieuren zeigen, wie die Anwendung von Software für CAD ihnen behilflich sein könnte und ihnen gleichzeitig erlauben einen Blick auf die Probleme zu werfen, die mit einem Projekt dieser Art verbunden sind.

1. INTRODUCTION

The availability of computers had a great impact on structural engineering as well as on other branches of engineering during the past twenty years. The methods of analysis underwent a rapid change which allowed design engineers to overcome the problems involved in the integration of differential equations and especially in the correct formulation of the boundary conditions. This evolution took place step by step. First of all a general approach was introduced which could be used for every structural idealization and exhibited two complementary aspects, called respectively forces method and displacements method. They are quite similar if we look at the analytic formulation and both base themselves upon the effects superposition. Afterwards the problem of analysing the behaviour of continua was faced by developing first the method of finite differences and then the finite elements method. The latter drew the attention of design engineers upon itself because of its effectiveness and flexibility.

At the present time the finite element method is the most sophisticated and powerful computer-oriented procedure available for the static and dynamic analysis of structures, and a great number of computer programs based on this technique is used in all major industries.

On the other hand the use of this kind of programs is strongly hindered by the need of generating an analytic, computer-oriented, description of the physical model and of translating the results of computation in a synthetic, man-oriented, form. We must realize that the procedures for data or instructions input to the computer and information output have been the main problem on the way of the efficient use of the computer since it appeared. Man thinks indeed in terms of words and numbers, or diagrams and pictures, whereas computer can operate only with coded digital representations of those entities. This fact causes difficulties in the communication between creative, slow and error-liable man and literal, fast and error-free machine.

Conventional interfaces between man and computer are punched cards for input and printer lines for output. Keyboard terminals and plotters represent an improvement of these conventional interfaces, but the introduction of third generation computers offering the availability of communication and interaction with the operator and the development of rather inexpensive storage C.R.T. graphic terminals give the opportunity to enable the civil engineer to overcome the critical steps in the use of structural analysis programs.

In the past few years tools for computer-aided design based on interactive computer graphics were developed and they have been transformed from an expensive curiouseness into a low-cost, useful and sometimes necessary instrument for structural engineers.

Now we will get a picture of how these people could be helped by the use of software for computer-aided design and we will give a glance, at the same time, to the problems involved in the design of this kind of software.

2. CHECK FOR DATA ERRORS

The conceptual simplicity of the finite elements method transforms the user's main problem from the need of finding an analytic solution for the mathematical model of a structure to breaking-down the geometry of the structure under consideration into suitable elements of regular shape (called the idealization).

A considerable amount of data must be prepared with corrispondingly large manpower requirements, particularly for three-dimensional analysis. Thus, when this whole task is performed manually, human errors are introduced adding further cost for abortive analysis runs and corrections. All the structural analysis programs automatically detect most of the input data errors before a large amount of internal computation is performed. However the automatically detectable errors are the formal ones or those which lead to some inconsistencies. Some typical user's errors, as wrong nodal points coordinates, wrong elements connections, overlapping or missing elements and so on, could not be detected by the errors cheking systems based on numeric evaluations.

In these cases only a graphical representation of the idealization of the structure would be helpful.

Such a representation should allow the user to :
- select the part of the idealization which has to be displayed,
- rotate, shift and scale the selected portion of the model,
- produce orthogonal, isometric and perspective views of the previously defined assembly of elements, with the ability of shrinking elements (a useful device for ensuring that internal elements are actually present) and displaying after magnification small areas of the selected portion of the model so that detailed investigations may be made (zoom),
- obtain elements and nodal points numbering,
- display any input numeric information regarding materials data, boundary conditions, substructures connections and so on,
- correct on line wrong data,
- add ar delete or modify elements.

It is quite clear that only the last two operations require the use of interactivity. However we must realize that the use of an off-line device, such as a plotter, is good only for final documentation. Selection and representation parameters may be wrong as well as corrections at modifications of the idealization. Therefore interactivity allows the user to try and try again until a satisfactory condition is achieved.

Moreover graphic interactivity can be very helpful in communicating with the computer because it works in a man-oriented fashion. It allows indeed the user to give or ask informations via a graphic device : by means of a light pen (a pencil-like device which the operator can use to point at something of interest on the screen of the graphic display terminal) or a joy-stick, a graphic tablet or other manually operated devices, the user can indeed communicate with the computer in a very direct manner without using any special language.

As far as the effectiveness of the picture is concerned, we must keep in mind that isoparametric elements commonly available in structural analysis programs have curved edges and curved surfaces defined by particular internal nodal points. Therefore a good treatment of curved lines in necessary : the points which define an edge are interpolated by special spatial curves, new points are considered on these curves and finally the edge is substituted by a sequence of straight lines. This operation must be performed in the 3D space when we deal with solid or shell elements, before any transformation to the 2D space representation is done.

Moreover, if we wish to eliminate hidden lines in the picture of spatial idealization also a good treatment of curved surfaces is necessary, after the general problem of hidden lines elimination has been solved. It is clear indeed that a surface can be visualized only by means of its edges : if it is plane no problems arise, otherwise new lines could delimitate its 2D representation depending on the point of space from which the user wish to look at it.

All these features of the visualization tool which enable the user to have a good picture require special efforts in designing and developing the necessary software.

3. DISPLAY OF RESULTS OF ANALYSIS

Corresponding to the large amount of input data required, many pages of output can be expected containing a great number of numerical informations. It is quite clear that this kind of informations has no physical evidence, therefore the engineer must examine the numbers in order to transform the analytical results of the analysis into significant patterns in graphic form.

If we consider this fact the usefulness of an automatic output data reduction is apparent. Its fundamental aim should be to produce pictures which compress the relevant informations.

As far as the output graphic device is concerned, also in this case we must realize that conventional permanent copy plotting is good for final documentation purposes while interactivity is preferable when looking for the most significant displays of the results. The choice of magnification factors to apply to deformed shapes, for example, or the best density of contours lines and so on, are not likely to be defined without some trials.

The capabilities of an efficient post-processor for structural analysis should not be different from those of a pre-processor with reference to the representation of the deformed shape of the idealization. Moreover the post-processor should allow the user to :

- display the deformed shape, magnified as much as required, superimposed on the undeformed one to give a clear impression of the overall distortion,
- display stress or strain contours with any desired density in two-dimensional idealizations and in any prescribed section, not necessarily flat, of three-

dimensional idealizations,
- produce diagrams showing displacements, stresses or strains along any line through the body.

Looking at the deformed shape it is useful that the whole mesh is represented to give informations about the displacements which occur in the interior part of the body. On the contrary, when representing the isostress or isostrain lines, the subdivision into elements is a disturbing fact and therefore it is necessary to eliminate the internal boundaries between elements while retaining the external boundary of the whole object along the chosen cut.

The diagrams of the various results of the structural analysis can be of different forms : they can have the standard aspect or a vectorial representation. In the latter case a set of straight segments are traced at right angles to the cut line, every segment having a lenght proportional to the size of the evaluated quantities.

These possibilities of representation require new efforts from the developers of the related software. It is necessary to provide for cutting of three-dimensional bodies along not flat surfaces besides implementing diagrams representations techniques.

4. BULK MESH GENERATION

The subdivision of the geometry of the structure into finite elements can be performed automatically. This means quick and easy generation of input data for programs based on the finite element method.

The characteristics of pre-processors devoted to this purpose should be :
- the availability of facilities in describing the geometry of the idealization by means of points, curves, surfaces, bodies, etc.,
- the automatic generation of nodal points coordinates and nodes and elements numbering,
- the easy change of element types and mesh coarseness.

If we restrict our interest to the meaning of an easy definition of surfaces, assuming that similar techniques can be used for curves and bodies, we must realize that a surface can be described in many ways. We can define many points, and then interpolate or approximate them by some suitable functions. For general surfaces this is the most appropriate method, but for commonly encountered structures it is too complicate. These structures consist indeed of parts of planes, cylinders and spheres. Using the analytical descriptions of these shapes a little amount of informations is needed to define a structure. A sphere, for example, is defined by its center radius. In the same way the most commonly used curves, as straight lines and circular arcs, can be defined by two or three points. When only a part of an analytically described surface is wanted, it should suffice to describe the border of the wanted surface. Because of the existence of an infinite number of surfaces having the same border, artifices

must be provided to make the user sure about the surface that has been chosen.

At this point it is clear that serious problems arise for the developers of the software for this kind of automatic mesh generation. In any case the user must be enabled to build up the idealization from simple and natural objects like points, lines, surfaces and so on. If all these objects are given names, new objects can be defined merely by reference to those defined earlier. This natural geometric language should allow the user to define almost any kind of structure and to use any type of elements.

Some experiences achieved in producing software for automatic mesh generation showed that two rather opposite approaches are possible :
- a poor automatic discretization, requiring a refined subdivision of the body into many parts of rather simple geometric shape, but strongly subjected to the wishes of the user and therefore likely to give satisfactory results in almost any case,
- an effective automatic discretization with a reduced intervent of the user, but not always completely satisfactory.

From the user's point of view the difference between tho two approaches lies in the possibility of describing only the external boundary of the whole idealization against the necessity of subdividing it into smaller parts all requiring the description of the related boundary. It is apparent that, from the software developer's standpoint, the latter approach presents more difficulties than the former due to the presence of wider regions with irregular borders. In these regions of arbitrary shape it is not possible to use previously defined discretization patterns. To clarify this point we may spend some words to give a glance at how the first approach works. A given surface in three-dimensional space is described in parametric form such as to establish a close correspondence with a quadrilater or a triangle defined in the bi-dimensional space of the parameters. After the user has established the type of required elements and the distribution of elements coarseness, the discretization takes place in the parameters space and then is transferred to the spatial surface. The same happens for solid bodies. This way of working requires obviously a subdivision of the idealization into parts which can be described analytically as results of a transformation of very simple geometric forms. Being the discretization applied to these forms we can speak of an almost predefined discretization which strongly simplifies the problems encountered in the design of the related algorithm.

Reference was previously made to the capability of automatic nodes and elements numbering. It is a quite trivial operation if we do not consider the relevance of a numbering oriented to reduce the time required by the solution of the set of simultaneous linear algebraic equations built up by the structural analysis program. Most of these programs use direct methods which yield the solution by performing a fixed number of arithmetic operations. This number heavily depends upon the order in which the particular method eliminates the equations. When a front solution algorithm is used the order of elements is the important factor in increasing the efficiency of the elimination process while the node numbers

are merely unique identifiers. This order should be such that elements topology is defined progressively through the structure in rows, in a way suited to make the longest row as short as possible. When other band algorithms are used it is necessary to order the nodes according to the previously explained criterium. In any case the automatic mesh generator should provide for a suitable numbering of elements and nodes which could take place after the completation of the discretization.

As far as the choice of elements is concerned it has to be remarked that an easy change of the type represents a valuable capability. The use of higher order elements where a discretization using lower order elements has already taken place is a common operation which wastes time when performed manually. All elements connections must be changed, new nodes coordinates must be defined, a general renumbering of elements and nodes is necessary. When performed by the computer this operation is rather simple and the development of the related soft ware does not present difficulties.

Interactivity has a remarkable role in automatic mesh generation both in facilitating the description of the geometry of the idealization and in allowing the user to modify the automatically generated meshes when they are not satisfactory.

5. PROBLEMS IN DEVELOPING SOFTWARE

It is quite obvious that an interactive system must assure a response from the computer in a comfortable time, that is, balanced to the expectation of the user according to the complexity of the operations requested. Considering this necessity the algorithms for hidden lines elimination are likely to be critical while automatic mesh generation is not strongly limiting. We must consider indeed that the individuation of the parts of the idealization which are hidden from other parts requires many investigations. First of all these algorithms must recognize the so-called internal faces, i.e. those elements faces which constitute boundaries internal to the discretization and are therefore shared between elements contiguous one with other. After these faces have been removed the outer skin of the idealization must be examined to determine which parts could be viewed by the observer if they would not be hidden by other parts which can in turn be viewed. When we deal with elements having curved surfaces it could be necessary to split the curved faces into two or three parts delimited by curved edges which are not present in the idealization's description. Finally there is the need of considering all the edges which are common to faces which can be viewed and faces which cannot be viewed to establish the correct relations between them and determine what can be seen and what is hidden. Taking into account the possible existence of line elements, bi-dimensional elements and solid elements, we can easily realize that hidden lines elimination algorithms are quite complicate and require a considerable amount of time to give the desired results. For this reason when developing the related programs, we take care of showing intermediate results : first the outer skin of the idealization is represented, then the part of outer skin that

can be viewed is shown (at this point the hidden line elimination is complete if we deal with a convex object), finally the remaining operations are performed until the elimination reaches its ultimate objective point.

Moreover the software developed for the previously described purposes must be tailored to the computer and the display devices used. We must recognize that the software of this kind offered tends to be computer-independent and output display device dependent at a very low degree. But this approach is valuable for those who develope software which has to be sold. When developing software for in-house applications it could be a worthwhile attempt to look for efficiency without taking care of a wide applicability. Generality can be achieved at a higher cost and therefore it may be disregarded. Anyway it must be clear that this choice affects only the structure of the data-base and the way of operating, while the algorithms of representation, mesh generation and hidden lines elimination constitute a know-how achievement valid for every new software development.

IABSE
A I P C
I V B H

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Structural Design Implications of Analytical Techniques in Computing

Influence des techniques analytiques dans la programmation sur le projet des structures

Auswirkungen von analytischen Techniken in der EDV für die Berechnung von Tragwerken

A.R. CUSENS

Professor of Civ. Eng. Univ. of Dundee
Consultant, Posford Pavry & Partners
Dundee, United Kingdom

Summary

This paper reviews the advantages and limitations of current analytical approaches used in the computer analysis of structures. Some specific sources of error are indicated. The techniques covered include harmonic methods, the grillage method and the finite element method. The division of responsibility between: 1) computer program writer, 2) user manual writer, 3) computer bureau, 4) designer (and program user) is discussed in the light of the characteristics of the techniques covered in the paper.

Résumé

L'article passe en revue les advantages et les limites des approaches analytiques conventionnelles utilisées dans le projet des structures à l'aide de l'ordinateur. Quelques sources d'erreur typiques sont mentionnées. Les techniques considérées comprennent les méthodes harmoniques, la méthode du grillage et la méthode des éléments finis. La répartition de la responsabilité entre: 1) l'auteur du programme d'ordinateur, 2) l'auteur du manuel d'utilisation du programme, 3) le centre de calcul, 4) le projeteur(et l'utilisateur du programme) est envisagée en fonction des caractéristiques et des techniques présentées dans l'article.

Zusammenfassung

Der Artikel beschreibt die Vorteile und Grenzen von gewöhnlichen analytischen Annäherungen, welche im komputergestützten Entwurf von Tragwerken benützt werden. Einige typischen Fehlerquellen werden aufgezeigt. Die betrachteten Techniken sind die harmonische Methode, die Gittermethode und die Finiteelementen-Methode. Die Teilung der Verantwortung zwischen: 1) Autor des Komputerprogramms, 2) Autor des Programmhandbuchs, 3) Rechnungszentrum, 4) Entwerfen (und Programmbenützer) wird anhand der im Artikel dargestellten Eigenschaften und Techniken besprochen.

Leere Seite
Blank page
Page vide

INTRODUCTION

This paper reviews the advantages and limitations of current analytical approaches used in the computer analysis of structures. Some specific sources of error are indicated.

The techniques covered include harmonic methods, the grillage method and the finite element method. The division of responsibility between

1) computer program writer
2) user manual writer
3) computer bureau
4) designer (and program user)

is discussed in the light of the characteristics of the techniques covered in the paper.

HARMONIC METHODS

The use of methods based on Fourier series solutions have been quite widespread particularly in the analysis of bridge decks. The main techniques have been

a. Orthotropic plate theory. This usually involves the solution of the fourth order differential equation for elastic orthotropic plates by the Levy-Nadai method[1]. Guyon[2] and Massonnet[3] set up a simple tabular technique in which the first term only of the sine series representing the elastic deflection curve was used to characterise the distribution of bending moments due to wheel loads on bridge decks. The advent of the computer has permitted more accurate solutions using a relatively simple program. The Highway Engineering Computer Branch of the British Department of Transport has issued a program ORTHOP[4], based on work by the author, for application to right slab and pseudo-slab decks with edge-stiffening beams. This enables the calculation of bending and twisting moments and shear and reactive forces.

b. Finite strip method. Cheung[5] originated this hybrid method which is useful for bridge and roof structures of uniform cross-section. The structure is divided into longitudinal strips running the full length of the structure. The longitudinal variation of displacement is characterised by trigonometric series functions and the transverse variation by polynomial

functions (as conventionally used for the finite element method). The method has considerable advantages in terms of simplification of input data and economy of computer storage and run time as compared with the finite element method.

c. Folded plate method. The matrix formulation by Scordelis[6] of the Goldberg and Leve analysis for folded plates has been programmed and applied to bridge and roof structures. It is a series solution but is less versatile than the finite strip method which has largely replaced it.

d. Limitations and practical difficulties. The harmonic methods are all based on functions of the form

$$p(x) = \sum_{n=1}^{r} \left(H_n \sin \frac{m\pi x}{L}\right) A_n$$

where $p(x)$ represents the load distribution

H_n is a load function (in trigonometric form)

A_n is a function embodying geometric and stiffness parameters and hyperbolic and trigonometric functions

Hence the displacement

$$w = \sum_{n=1}^{r} \frac{L^4}{n^4 \pi^4} \left(H_n \sin \frac{m\pi x}{L}\right) B_n$$

where $\frac{\partial^4 B_n}{\partial x^4} = A_n$

Because the series for w converges with $\frac{1}{n^4}$ it converges very rapidly. The series for bending moments depend upon terms in $\frac{\partial^2 w}{\partial x^2}$ and thus converge less rapidly with $\frac{1}{n^2}$. The series of shear force converges very slowly with $\frac{1}{n}$. These essential differences are not always understood by users of programs based on this method.

e. Examples

1. Fig. 1 illustrates the relationship between the shearing force diagram and the number of harmonic terms used for the finite strip solution of one span of a continuous box beam under uniform loading. As the number of terms increases, the harmonic solution approaches the correct linear distribution. However values of reaction at the support are very significantly lower than true reaction values.

   It is possible to obtain reasonable assessments of shear force at the supports by considering a position near rather than at the support point but users who fail to understand this characteristic of the method can make errors.

2. In a development of orthotropic plate theory the ORTHOP2 program is currently being developed to analyse right slab and pseudo-slab bridge decks with unequal edge beams. The edge beam stress parameters are developed in terms of the deflection profile at the slab-beam boundary. The expression for bending moment in the edge beam develops a tendency to oscillate about the true solution if more than a small number of harmonic terms are considered. This is clear from Fig. 2, which shows comparisons with finite element and finite strip results for the same problem. Fig. 3 illustrates the divergence which occurs above 6 harmonic terms. Here then is a circumstance which is the opposite of Example 1, i.e. now, a large number of terms does not improve accuracy.
   These two examples provide an object lesson to users on the necessity to understand the characteristics of analytical techniques before applying them to design situations.

GRILLAGE ANALYSIS

The representation of a slab or pseudo-slab structure by a grid or grillage of beams interconnected at rigid joints has always been a popular technique with bridge engineers and it has been given new impetus by the availability of the computer.

The method is relatively simple in terms of data preparation and economical in computer storage and run time.

a. Choice of beam spacing

   Each beam of the grillage replaces a finite width of the deck. Thus the choice of the relative positions of the beams is important as also is the allocation of bending and torsional rigidity to each beam. West[7] recommends that there should be odd numbers of longitudinal and transverse beams and that as far as possible they should be at equal spacing and of equal stiffness. The gross torsional rigidity of the deck should be assigned in equal parts to the longitudinal and transverse beams. An orthogonal pattern of beams is desirable even for a skew planform - even though this conflicts with the recommendation for equal beam spacing.

b. Limitations and practical difficulties

   1. The grillage method is relatively insensitive to concentration of stress and, for example, will underestimate the peak stress below a small patch of load.
   2. The grillage method often underestimates torsional moments and overestimates values of bending moment at positions remote from a load concentration.

These two points are illustrated by Figs. 4 and 5, which show values for four alternative skew grillage arrangements for the analysis of a model skew deck tested by Rusch and Hergenroder.

3. Results are difficult to interpret for decks which have non-parallel edges because the beams now represent variable widths.
4. Results are unreliable for curved decks where the angle is greater than about 20 degrees between supports.

FINITE ELEMENT METHOD

The finite element method is now well-known to most structural engineers through the work of practitioners such as Zienkiewicz[8] and others. It is the most general of the methods available for structural analysis and large packages (e.g. PAFEC and NASTRAN) are now available with a variety of alternative elements for two- and three-dimensional stress analysis.

Limitations and practical difficulties

1. The finite element method is both costly and complex to use. Its cost makes it particularly desirable to avoid errors and abortive computer runs, but its very complexity makes it prone to misunderstandings between design engineer and analyst. The method should only be employed where simpler techniques are inappropriate, e.g. for non-uniform members, non-standard geometry, or inelastic materials.
2. The volume of input data for a finite element analysis is usually large and errors are often difficult to spot.
3. Hinton[9] points out the difficulties of interpreting stress distribution from a finite element program output because of discontinuities between elements. Interpretation is largely subjective and can therefore be inconsistent and irrational.
4. Simple equilibrium checks should always be made to ensure that gross errors or misunderstandings are not present. A recent case is known to the author, of a skew bridge deck analysed using a large finite element package, where elements were chosen which for skew axes imposed a degree of restraint at the nominally simple line supports. This effectively reduced the mid-span bending moment and design of the bridge was well-advanced before a perceptive engineer made a simple check of equilibrium and discovered the mistake.

RESPONSIBILITY

There are four groups of personnel who carry responsibility for the computer program and its use in the analysis and design of a structure. These are:

1) Computer program writer
2) User manual and program manual
3) Computer bureau staff
4) Engineer designer (program user)

Frequently one individual will have written both the program and the manuals and this is a desirable state of affairs. At least there must be very close collaboration between the program writer and the author of the manuals. Computer bureaux usually take over proven standard programs and their staff may have a very limited knowlege of the structural principles underlying the program. However they should gain a thorough knowledge of the manuals and the program input and output. They should be ready to seek advice from the originator of the program in cases of difficulty.

The structural designer chooses (or sanctions the choice) of the program to be used for a particular problem. He must therefore be aware of the general characteristics, limitations and costs of each of the alternatives. He must make extensive checks to ensure that the results are structurally valid.

If shearing stresses are likely to be critical in a particular structure then the finite strip method is a poor choice. If local moment peaks are important (for example in fatigue situations), the grillage method is not the best choice; the finite element method will only give reasonable results with a very fine mesh arrangement; on the other hand a harmonic method, if applicable, would be accurate and inexpensive.

Most major computer programs used in structural analysis have been written by engineers whose understanding of the implications of the technique used is beyond question. However errors of logic do occur in programs and can often lie undetected until a particular problem solution or a change of computer system brings them to the surface. These must be guarded against and although they might be said to be primarily the responsibility of the original program writer, the bureau staff and program user must also be on their guard against such an occurrence.

In general the designer inevitably bears the main responsibility over the use of computer programs. He chooses the program (which implies knowledge of the underlying method of analysis); he must be able to know if results are substantially in error. Computer bureaux staff have responsibility as sub-contractors to ensure that the program is working as intended and that input data are checked. If they recommend a particular program they must be familiar with the limitations of the program. Manual writers have the responsibility of ensuring that both the capabilities and the limitations of a program are clearly stated. They must provide guidance to aid avoidance of

errors and misunderstandings but cannot be held responsible for users who ignore manual instructions. Program writers are responsible for translating a method of analysis or a design procedure into a logical and unambiguous computer program. They cannot be held responsible for subsequent misuse of programs which they have written. However if their program is written under contract for a particular purpose subsequently not fulfilled, or if they are extracting a royalty for use of a program, there is an obligation to provide help and advice to users.

REFERENCES

1. Timoshenko, S.P. and Woinowsky-Krieger, S. Theory of plates and shells. McGraw Hill, 1959.

2. Guyon, Y. Calcul des ponts larges à poutres multiples solidarisées par les entretoises. Annales des Ponts et Chaussées, 24, 1946, pp. 683-718.

3. Massonnet, C. Methode de calcul des ponts à poutres multiples tenant compte de leur resistance à la torsion. Publ. IABSE, 10, 1950, pp. 147-182.

4. Bakht, B. and Bullen, R.C. ORTHOP User Manual. HECB/B/15 Department of the Environment, 1975.

6. Cheung, Y.K. The finite strip method in structural analysis. Pergamon, 1976.

6. De Fries-Skene, A. and Scordelis, A.C. Direct stiffness method for folded plates. Jnl. Struct. Div. ASCE, 90, ST4, Aug. 1964, pp. 15-48.

7. West, R. Recommendations on the use of grillage analysis for slab and pseudo-slab bridge decks. Report 46.017 Cement and Concrete Association and CIRIA, London, 1973.

8. Zienkiewicz, O.C. The finite element method. 3rd Ed. 1977, McGraw Hill.

9. Hinton, E. The finite element method. Lecture notes to PTRC course on bridge deck analysis, London, Nov. 1977. Planning and Transport Research and Computation Company.

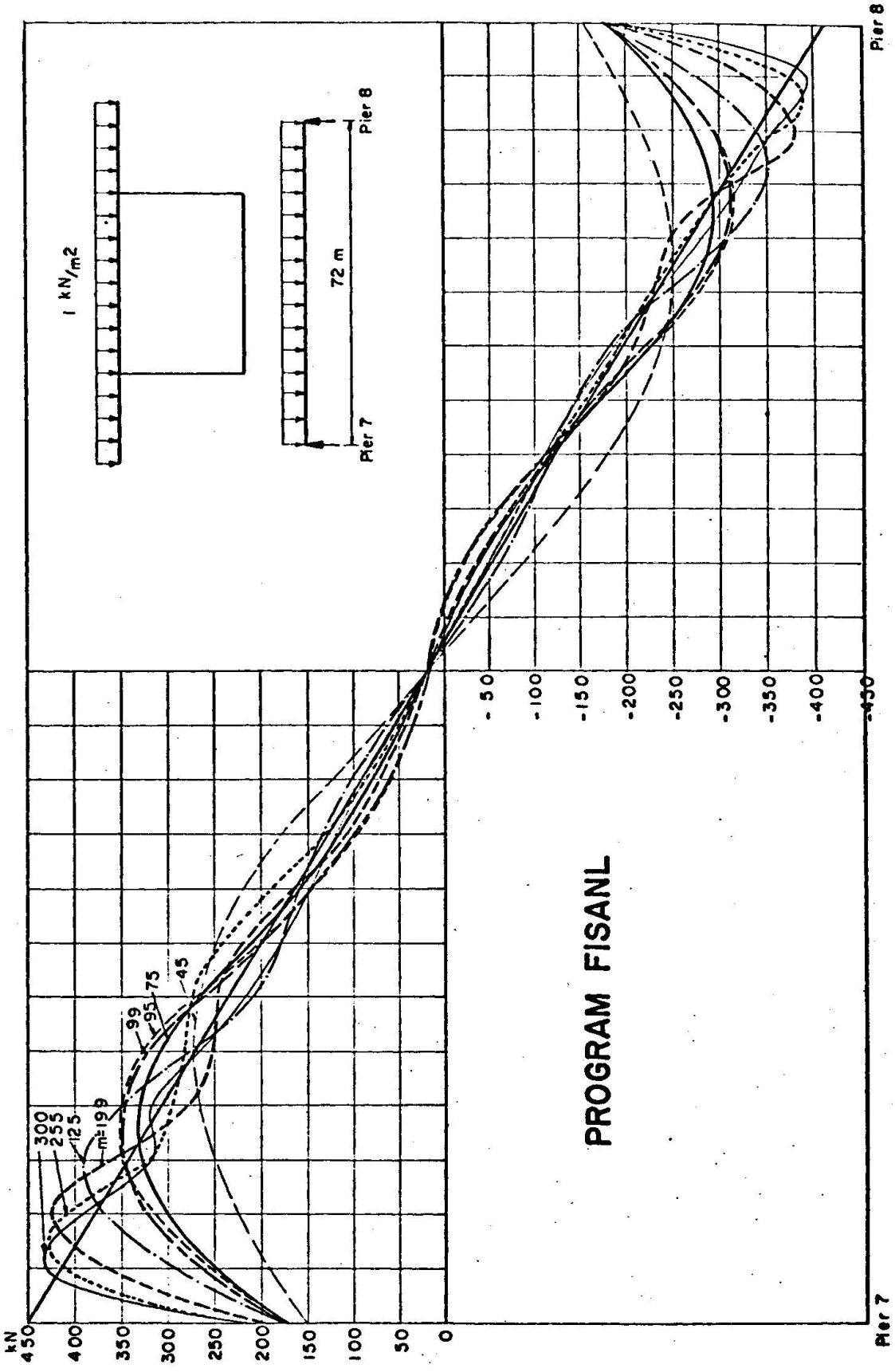


FIG. 1 SHEARING FORCE DISTRIBUTION ALONG ONE SPAN OF CONTINUOUS BEAM

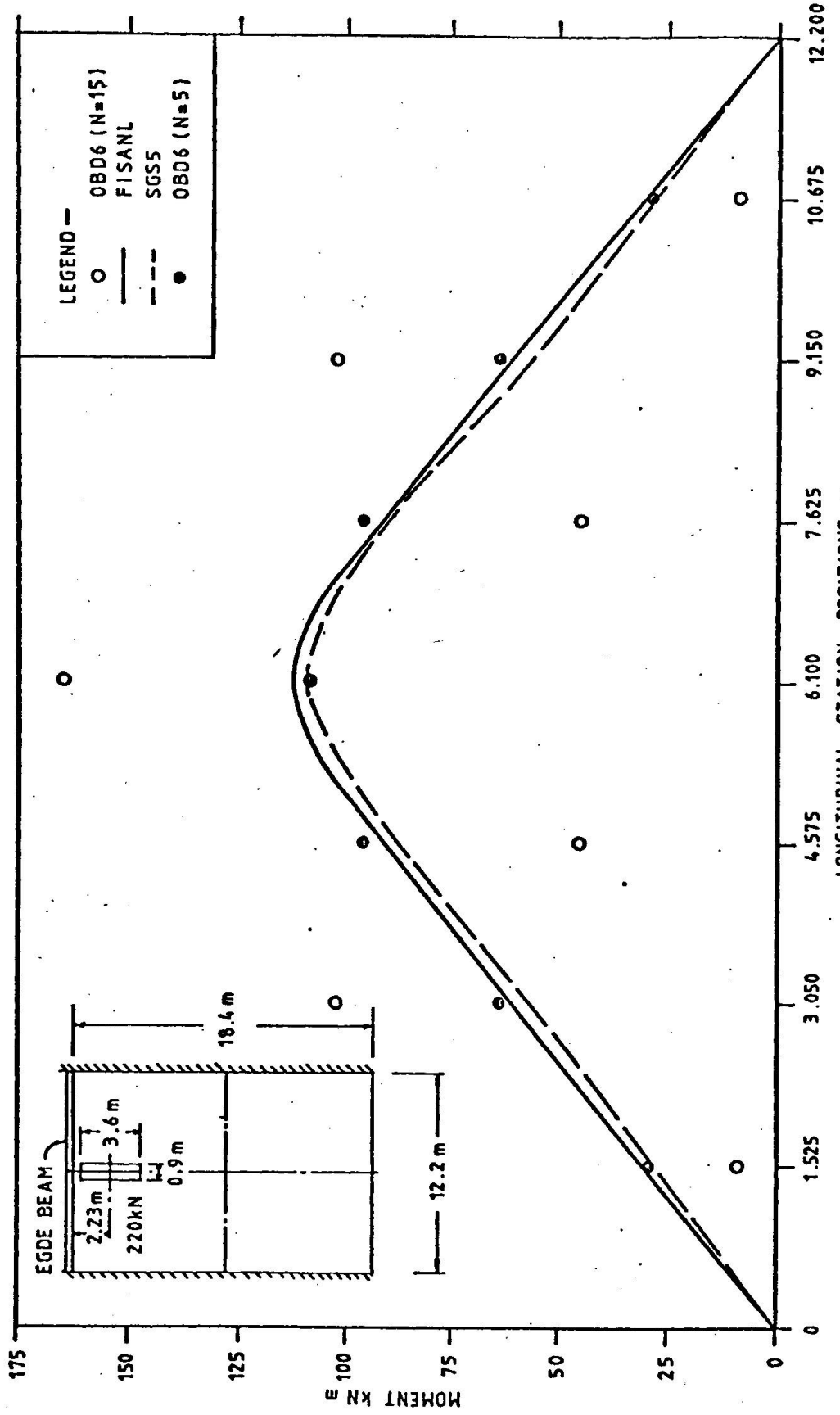


FIGURE 2 EDGE BEAM BENDING MOMENTS 220kN LOAD AT 2.23 m FROM THE EDGE

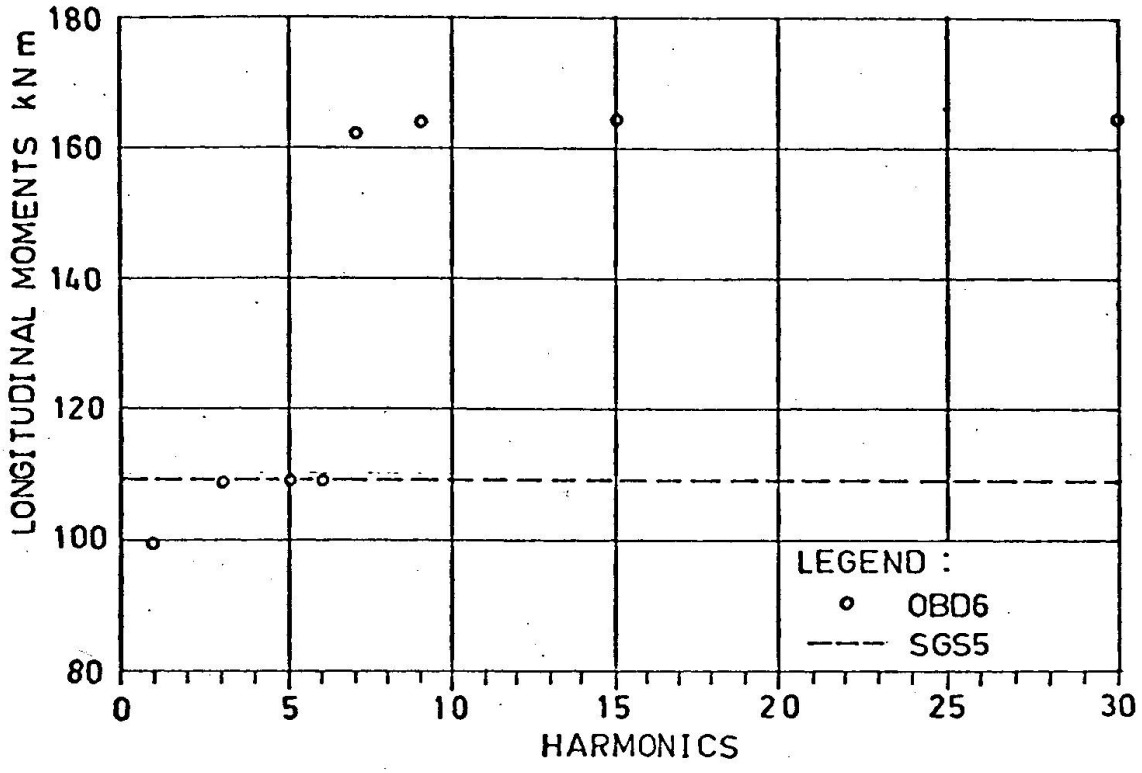


FIGURE 3 VARIATION OF MID-SPAN EDGE BEAM BENDING MOMENT VALUES WITH HARMONIC SUMS

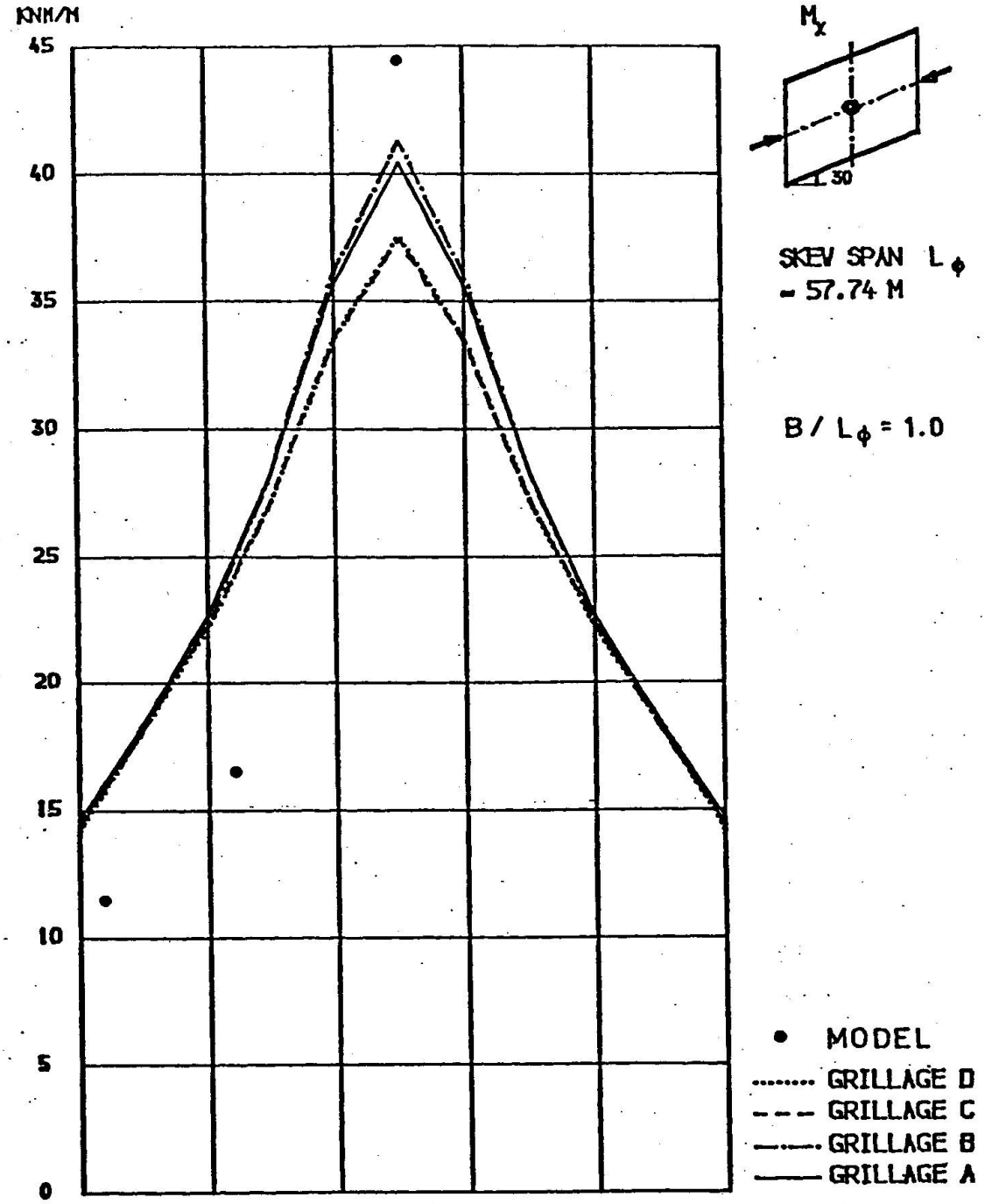


FIG. 4 LONGITUDINAL MOMENT PROFILE AT MID-SPAN

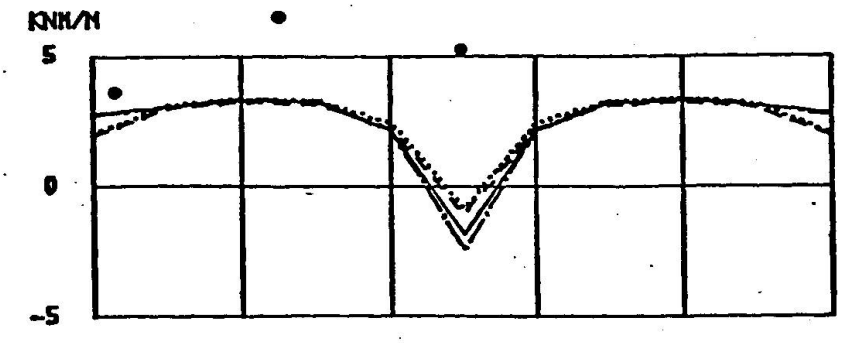


FIG. 5 TWISTING MOMENT PROFILE AT MID-SPAN

IABSE
AIPC
IVBH

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Elasto-Plastic Stage by Stage Analysis

Analyse elasto-plastique des stades de construction

Elastoplastische Bauvorgangsberechnung

I. UHERKOVICH

Head of the Design Office
Losinger Ltd - VSL International
Berne, Switzerland

Summary

The latest knowledge of the physical properties of materials and of increasingly complicated construction methods is fairly soon mentioned in design codes. The paper analyses the possible ways of solution. The suitability of an extensive simulation of the physical processes in an engineering calculation is discussed. The requirements with regard to the capacity of the computer and therefore the cost of the calculations may become very high, whereas the possibilities of checking the results remain small.

Résumé

Les connaissances nouvellement acquises sur les propriétés physiques des matériaux ainsi que les méthodes de construction de plus en plus complexes sont mentionnées assez rapidement dans les normes de calcul. L' exposé analyse l'opportunité de disposer d' un modèle de calcul reproduisant avec fidélité les processus physiques. Ces calculs exigent des ordinateurs de grande capacité et impliquent par conséquent des coûts relativement élevés, alors que les possibilités de contrôle des résultats restent faibles.

Zusammenfassung

Die kenntnisse über die physikalischen Eigenschaften der Baustoffe wie auch die Anwendung von ständig komplizierteren Bauvorgängen finden heute einen sehr schnellen Niederschlag in den Entwurfsvorschriften. Im Beitrag wird anhand eines Beispieles über die Zweckmässigkeit der treuen Nachbildung der physikalischen Vorgänge bei Ingenieurberechnungen diskutiert. Die Anforderungen an die Kapazität der Rechenanlage und damit auch die Kosten der Berechnung können aber sehr gross sein, die Kontrollierbarkeit der Resultate klein.

1. INTRODUCTION

There is a very close correlation between the possibilities of calculation and the designed structures.

In the past, when we had recognized the influence of certain executional solutions or material properties, we chose such a constructive detailing in the design so as to limit this influence to a negligible value in order to avoid calculations which, at that time, were practically not executable. Today we have computers which, according to the opinion of many people, seem to be means of unlimited power.

This leads to the design of structures in which the influences we neglected in the past have to be considered as accurately as possible. This, on the other hand, leads to the development of computer programmes which not only represents a quantitative increase and acceleration of the existing calculation algorithms developed for hand calculating, but it also forms an absolutely new and different working level.

In this contribution I would like to try to clarify the development having occurred by means of an example, as I think that the findings from the analysis of a particular problem will lead to general conclusions.

2. SOME WORDS ON CONCRETE CREEP AND ITS EFFECTS

As already mentioned in the introduction, certain material properties can highly influence the volume of a calculation. This, for instance, is true for the stress-strain behaviour of concrete which does not have a linear, time-independent character, as it does for steel, but which is very much dependent on the environment and time. In the last two decades the research of this behaviour, generally known as shrinkage and creep of concrete, has become a favorite field of activity for numerous scientists to whom we also owe the quick introduction of new findings into the standards.

In the new "CEB/FIP Recommendations for the design and construction of concrete structures" for example, the concrete deformation from a stress increase $\sigma_{(t_o)}$ in the period t_o to t is given as:

$$\varepsilon_{(t,t_o)} = \sigma_{(t_o)} \left\{ \left[\frac{1}{E_{c(t_o)}} + \frac{1}{E_{c(28)}} \right] \left[\beta_{o(t_o)} + \varphi_d \, \beta_{d(t-t_o)} + \varphi_f \left(\beta_{f(t)} - \beta_{f(t_o)} \right) \right] \right\}$$

where:

$$\beta_{a(t_o)} = 0{,}8 \left(1 - \frac{f_{c(t_o)}}{f_{c\infty}} \right)$$

$$\varphi_f = \varphi_{f_1} \cdot \varphi_{f_2}$$

$$\varphi_d = 0{,}4$$

The individual coefficients are found from the following figures (diagrammes) and tables:

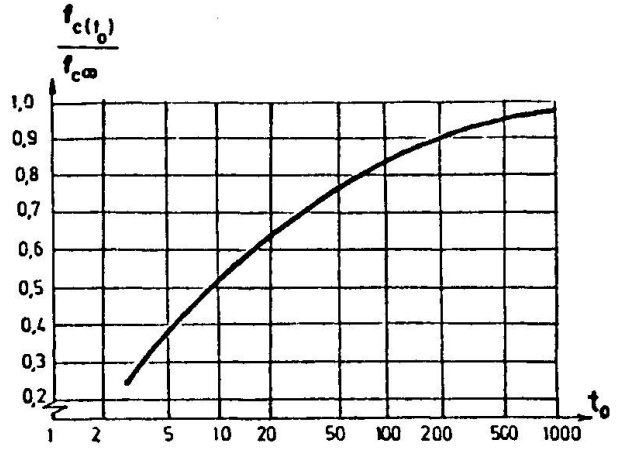


Fig. 1: Coefficient $\frac{f_{c(t_0)}}{f_{c\infty}}$

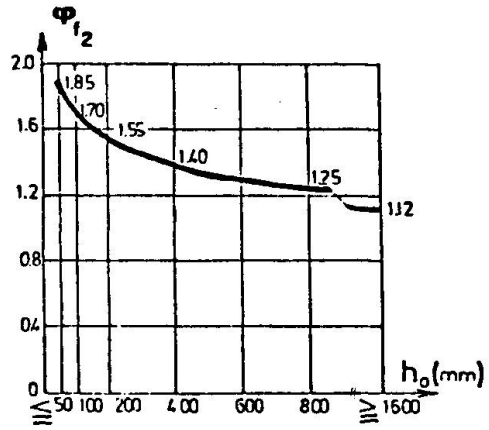


Fig. 2: Coefficient φ_{f_2}

Environment	Relative humidity	φ_{f_1}
Water	100 %	0,8
Very humide atmosphere	90 %	1,0
Exterior in general	70 %	2,0
Very dry atmosphere	40 %	3,0

Fig. 3: Coefficient φ_{f_1}

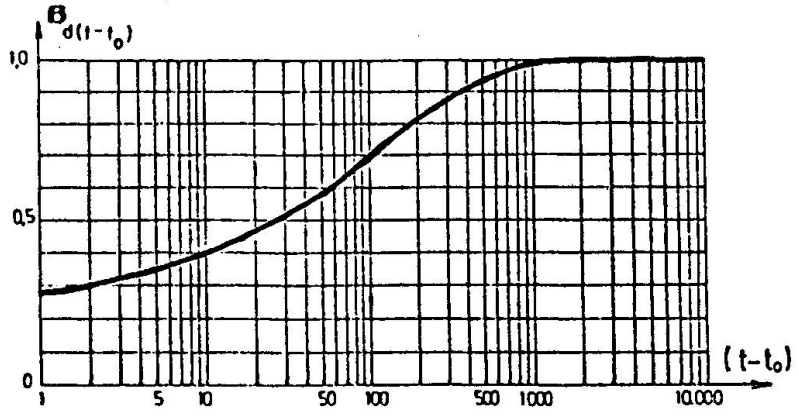


Fig. 4: Coefficient $\beta_{d(t-t_0)}$

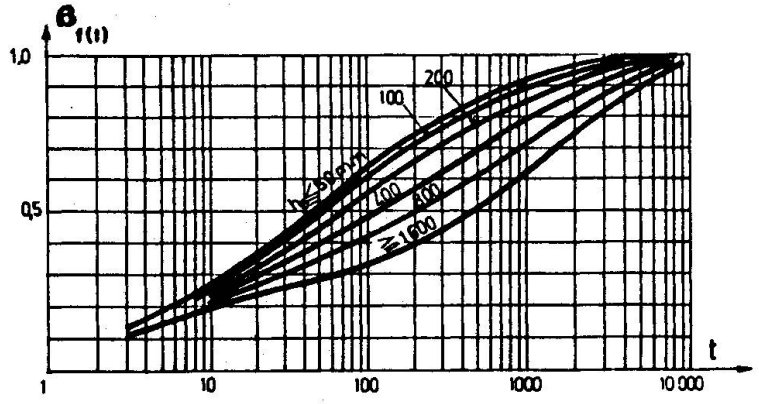


Fig. 5: Coefficient $\varphi_{t(t)}$

I do not intend to elaborate on concrete creep but I would like to emphasize a fact which is of definite importance for my further considerations.

The deformation of an element in a time period is not only given by the stress present in the element in this period, the time origin of the stress also plays a role. This means that in the superposition of several stress stages of different time origins in one element,(i.e. a structural part of homogeneous deformation behaviour),a different creep coefficient has to be introduced for everyone of these stress stages. If we pursue the deformation of this element for a series of several time periods we have to elaborate a matrix of coefficients in which the number of elements is equal to the product of the "(number of stress stages) and (the number of time periods considered)".

If a structure consists of parts with different properties (which in general applies) - different properties here mean for example different age, dimensions, reinforcement content a.s.o. - then for each of these parts ("elements") a separate matrix of creep coefficients is to be established.

Without going into details, two of the numerous existing rheological models will be shown for completeness and for a better understanding of what a creep function is:

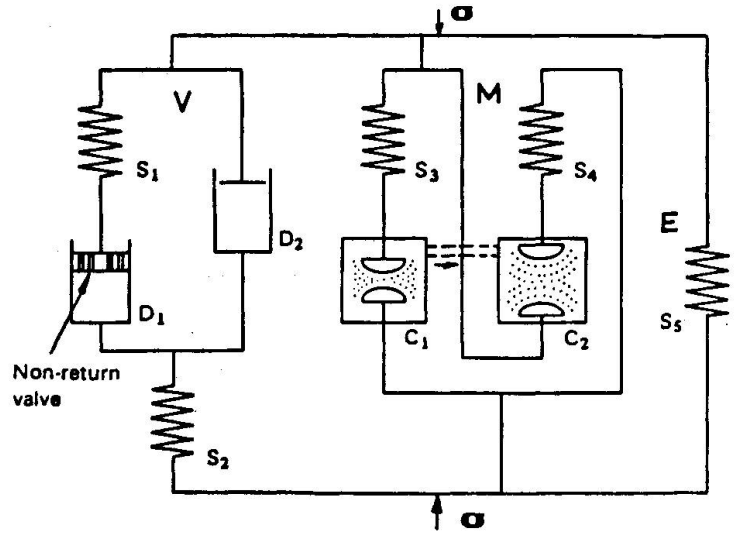


Fig. 6: The Gopalakrishnan-Neville-Ghali model

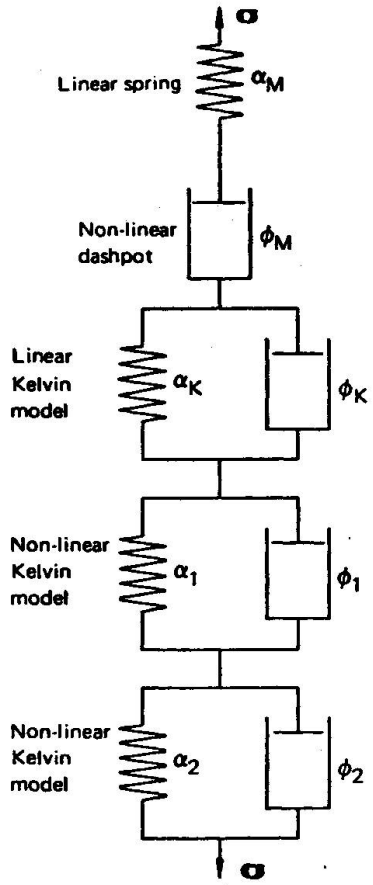


Fig. 6: Freudenthal Roll's model

A digital simulation of such processes is fairly difficult. To complicate the matter, it must be mentioned that independent of the stress-dependent deformation (creep) there is also a purely time-dependent deformation of the concrete (shrinkage) which superposes with the creep deformation. The total time-dependent deformation of an element in the actual structure (I am mainly thinking of post-tensioned concrete) leads to different but coupled effects:

2.1. By axially distorting the structural parts, the elongation-dependent post-tensioning force changes ("loss due to creep"); this change of the post-tensioning force and the resulting change of the stress distribution then influence the distortion itself.

2.2. With elements composed of parts with different deformation properties (so-called "composite sections" = i.e. concrete of different age plus steel) the stress shifts from elements of high deformability to elements of lower deformability (for example from young concrete to older concrete or from concrete to steel). This, however, influences the deformation properties of the whole element and thereby also the magnitude of the deformation. We therefore talk of a so-called "inner shifting of the stresses due to creep".

2.3. With structures where the boundary conditions vary with time (e.g. due to changing bearing conditions) secondary forces due to time-dependent deformation properties of the elements develop without a change of the active load condition. They in turn influence the deformations through changes of the stress distribution.

Here we are dealing with the so-called "outer shiftings of the bearing forces due to creep".

3. STRUCTURAL DESIGN PROGRAMMES TAKING INTO ACCOUNT THE CREEP OF CONCRETE

From the above presentation of the problem which probably was not short but the shortest possible, we come back to our theme which concerns the possibilities of calculating such processes.

If we intend to use a fully automatic processing, the programme must consist of the following main parts:

- structural design (e.g. by finite elements)
- calculation of stresses in the cross section including computation of post-tensioning
- calculation of creep and shrinkage

These parts must together form one system and run under a superimposed programme control. The Fig. 7 schematically represents such a programme system.

All data concerning the time of construction of the individual structure parts, the time of loading or unloading a.s.o. are stored in the part called "programme control". This part then regulates the sequence of calculation processes. After each time period a stress calculation has to be done for each element. This is used as a starting point for the creep calculation which must also be done for each element in each time period. Together with shrinkage the element deformations are computed which then serve as a starting point for calculating the changes of the prestressing force and the redistribution of the stresses in the section. The calculation of the change of secondary forces again has to be done by a new structural calculation (the calculation of compatibility). When the mathematical bases of this programme are elaborated, attention must be paid to the fact that the inner and outer force equilibrium conditions are strictly observed.

This brings considerable difficulties as there is no longer a direct relationship between deformations and forces.

As already mentioned, every stress stage in every element has to be stored as an independent cause of creep during the whole calculation in order to allow a creep computation to be made.

From the processing point of view some problems arise in the run, especially the long computation time and the need for large storage. It may be mentioned here that, in addition, for a highest possible accuracy in calculations, the (elastic) E-Modulus should be introduced as a function of concrete age and

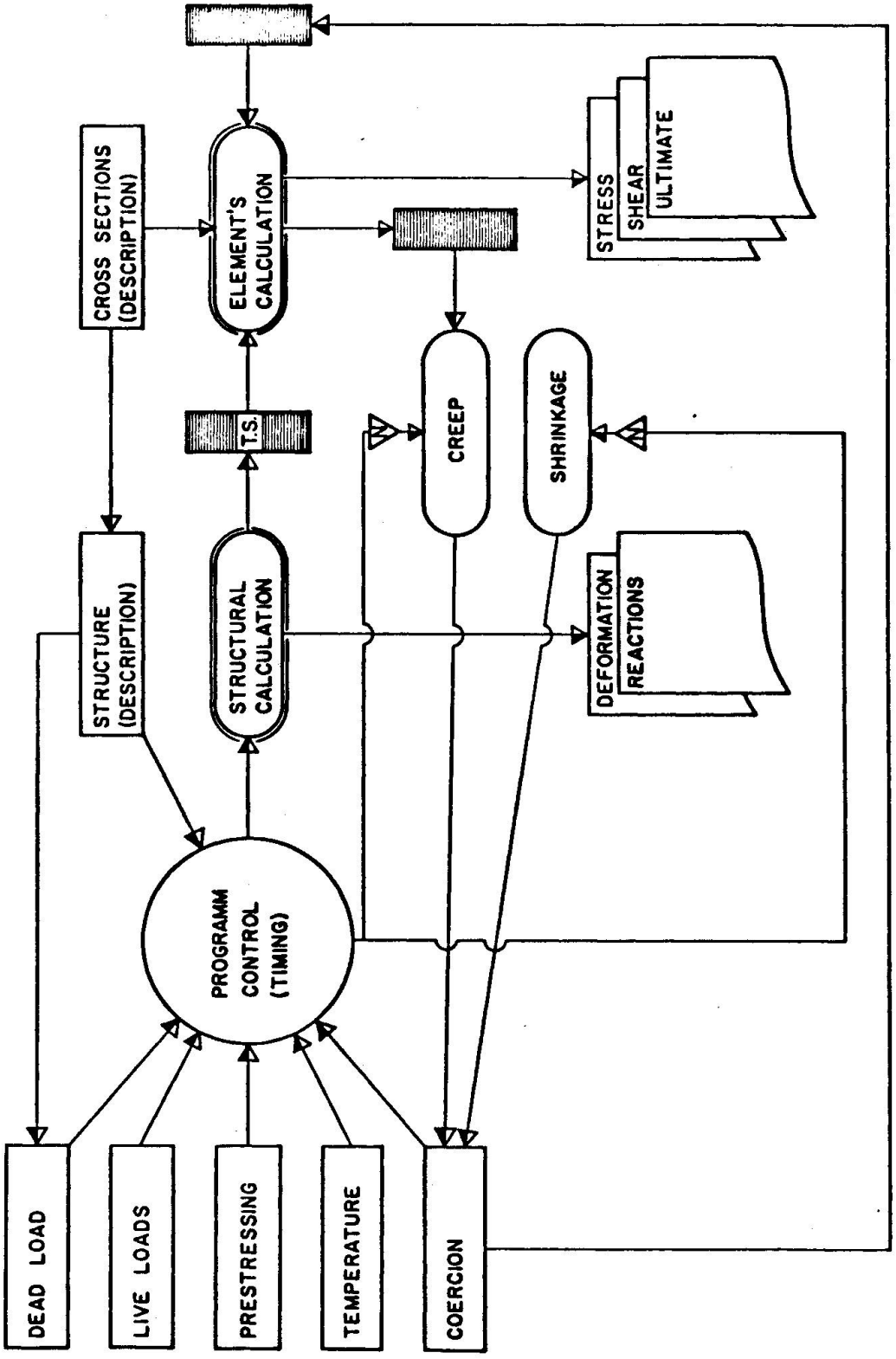


Fig. 7: Programme system for stage by stage calculation

that the real construction times have to be adjusted in the calculation according to the seasonal climatic conditions.

We tried to establish such a programme by strictly applying the creep formula mentioned in the introduction, with some simplifications in the process given in Fig. 7. We did not consider the "inner shifting of the stresses due to creep" for example, and the prestressing losses were calculated from the same initial data, and not from the deformations of the elements. The processing also was not fully automatic.

We neither considered a time-variable E-Modulus nor the adjustment of the time data from temperature influences. In the programme we made use of the fact that the elastic structural calculation forwards inherent element deformations; it therefore was sufficient to multiply these by the corresponding creep coefficients and to reintroduce the element deformations so obtained into the structural calculation as a loading case again in order to produce the compatibility of the deformations.

Hence the constraints ("outer shifting of the bearing forces") could be calculated which appear as a new loading case as they are also subjected to "decreeping" (relaxation).

The programme block scheme is given in Fig. 8.

The calculation of a fairly complicated case with more than 40 time periods could thus be done. Each time period practically led to a change of the structure. As "heart piece" the member programme "STATIK" (developed at the Swiss Federal Institute of Technology in Zurich under the direction of Prof. Anderheggen) was used. The proper elaboration of the full creep programme was done by Thomas Friedrich, Eng., in the surprisingly short period of 3 months.

4. DESIRABLE AND OBTAINABLE ACCURACY

The question whether such a high precision in calculations is reasonable, is very interesting. I would like to point out first that a highest possible accuracy, e.g. in prestressed concrete bridge construction, is not so much of interest for the stability design than for the deformation considerations of a structure. A calculation of the forces in the structure to an accuracy of $\pm$ 5% or $\pm$ 10% allows for a justified comparison with the allowable values. This is not the case for the form of a structure where not limiting values but real and most accurate possible values have to be considered. With a free cantilever bridge, for example, an up to 100 m long cantilever steadily moves up and down during construction. A calculation of accuracy usually required for statical calculations would lead to deformations that cannot be undone later and thus the bridge surface would become uneven.

For illustration I would like to present the following values: Thanks to the accurate manner of calculations in the previously mentioned example we were able to obtain all movements of a bridge cantilever with an accuracy of 20 mm compared to the effective values obtained on site (with respect to the centre of the 108 m long span). A hand calculation of reasonable volume (based on

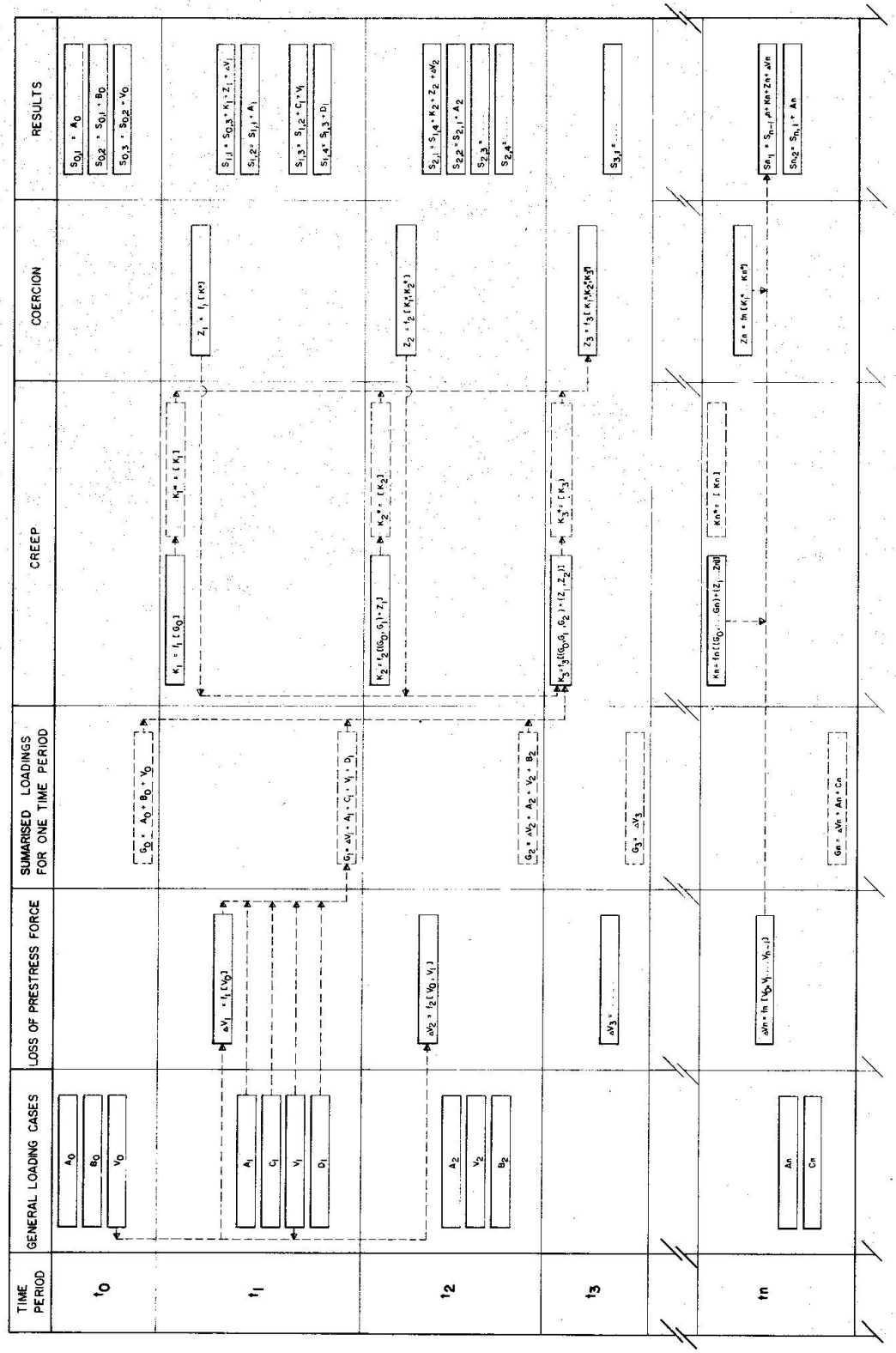


Fig. 8: Block scheme of realized programme

elastic deformations calculated with a computer) gave a deviation of 60 mm. With an increasing span length this value would rise with the power product of the span ratios.

One can imagine that a calculation without simplifications would bring the accuracy to below 10 mm in this case.

Considering increasing span lengths, more slender sections (and thus bigger deformations) and increasingly complicated construction procedures which take only little of the calculations into account, we can answer the question about the right of a highest possible accuracy in the affirmative. Obviously a calculated accuracy only makes sense if it is not beyond the possibilities of the actual conditions. In the mentioned case of the prestressed concrete bridge not only the effective material properties but also the influence of the atmospheric conditions had to be captured. This today is not only possible but generally done on the sites.

The obtainable accuracy, last but not least, is of economic and not of technical interest. The first prediction of the material properties and the construction times is always a first approximation only. A higher degree of accuracy in the calculation than that in the first adoption is only justified when a new run can be made with known new correction values. If such a new run is not economically acceptable, the accuracy required in the first run was of small utility. In other words the limits of obtainable accuracy are determined more by the calculation costs than by outside circumstances. This, however, means that as the hardware is becoming cheaper and cheaper, a temptation is created for a higher accuracy in the results by repeating runs, especially when no or only small programme costs are involved.

5. CONCLUSIONS

I have tried to demonstrate the problems of a coupled calculation of complicated, technical and physical processes by means of an actual example of the calculation of the "Computer Stage Analysis" used in bridge design, taking into account the elastoplastic properties of materials.

Of greatest importance, it seems to me, are the following facts:

5.1. The progress in practical construction, as well as applied research, forces the engineer to make more and more complicated calculations.

5.2. These calculations are made practically possible only by using computer programmes.

5.3. Due to the enormous progress in computer manufacturing and the consequent important price reductions of the calculators, such calculations today are possible at low costs and therefore the demand for them will certainly increase.

5.4. The possibilities of controlling the results, however, are very limited.

The real problems of such a development therefore are not in the technical mastering but in the general consequences, especially with regard to the responsability for the product and the allowance for an action without sufficient possibilities of control.

This, however, is the theme of another session.

Leere Seite
Blank page
Page vide

IABSE
A I P C
I V B H

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

Computer Calculation of Prestressed Concrete Bridges

Projet de ponts en béton précontraint à l'aide de l'ordinateur

Computergestützte berechnung von Spannbetonbrücken

W. OBERNDORFER

Dipl. - Eng. Dr., University Lect.
STUAG
Vienna, Austria

Summary

The paper deals with the software for a computer calculation of prestressed concrete bridges. First the main features, programming details and file organization are described briefly. After sketching the users' profile and the practical experiences that were found working with the system, conclusions are drawn about the communication between man and machine and about the professional responsibility for computer calculations.

Résumé

Le rapport traite d' un système très complet de programmes pour le projet de ponts en béton précontraint à l'aide de l'ordinateur. Les caractéristiques principales, des détails de programmation et l' organisation des archives est présentée. On essaie d'établir le profil de l' utilisateur et on décrit les expériences acquises lors de l'utilisation de ce système. Les conclusions traitent de la communication homme-machine et de la responsabilité de l' homme pour l'exactitude des calculs exécutés par l' ordinateur.

Zusammenfassung

Der Bericht behandelt ein umfangreiches Programmsystem für die elektronische Berechnung von Spannbetonbrücken. Es wird kurz der Leistungsumfang, Programmierdetails und die Dateiorganisation beschrieben. An eine Skizzierung des Anwenderprofils und der praktischen Erfahrungen, die mit dem System gewonnen wurden, schliessen sich Schlussfolgerungen zur Kommunikation Mensch-Maschine und zur Verantwortung für die Richtigkeit von Computerberechnungen.

1. INTRODUCTION

The requirements by the engineers concerning the accuracy and extent of computer calculations for prestressed concrete bridges increase ever more and computer programs appear to be an indispensable means.

The following paper deals with a software having been used since the year 1968 and being improved time by time. The programs were used many times for many different structures by many different users, concerning their ability to set up the input data and to understand the results. These experiences give way to make some generalizing statements on the application and use of such systems under real life conditions.

2. SYSTEM PERFORMANCE

2.1. General Remarks

As above said the computer can be only an aid for the engineer when designing a bridge. Therefore the general layout of the system was not done in a way that the computer grinds out a fully calculated and optimized bridge after feeding in input datas and parameters. An automatic and wholly integrated system surely would confine the freedom and the imagination for the design and the construction through the engineer and hinder the development of new ideas due to the necessary assumptions and rules of design that would have to be built into such a system. However, referring to the system described in this paper, the engineer can interrupt the computations at many states, repeat certain steps, maybe with altered input data, and drive the calculation of his bridge through the system like a car's driver on a crooked road. By doing so it is made sure that the engineer does not lose the immediate contact with the calculations and with the methods and algorithms being used in the programs.

The stock datas are the dimensions of the crossections, the magnitude and the coordinates of the prestressing force and the topological description of the structural overall system. They are fed in at the beginning, stored on a disc, and can be altered by ordinary procedures at any state of the work.

2.2. Short Description

Using the softwere dealt with in this paper the calculation of a complete prestressed concrete bridge can be carried through, beginning with the basic values of the crossection (area, static moments, moments of inertia etc.)and coming up with the behaviour of the structure under service and ultimate load conditions at the very end.

2.2.1. Structural System

It can be a plane frame or a plane girder grid. 3-dimensional frames are bent into a plane for finding the forces and reactions in the substructure. In dealing with the superstructure the columns are substituted by torsion bars with

$$I_x = I_y , \quad I_d = \beta(1+\nu) I_x .$$

(Checks have shown the validity of this assumption for practical purposes many times.)

2.2.2. Shape of Crossection

T-shaped beam (load distribution to be known)
Single hollow box
Double hollow box

2.2.3. Input

Dimensions of crossection
Topological description of the structural system
Magnitude and coordinates of the prestressing force
Loads
Materials' properties

2.2.4. Results

Values of crossection (F, y_s, W, I, I_d)
Formwork levels
Dead load and prestressing forces and moments
Internal forces and deflections under dead load, wind, lowering of supports, temperature, earthquake (quasistatic method)
Lines of influence and their evaluation for bending, torsion and shear, considering corresponding single loads on grids, selecting the governing live load mix according to the Austrian building code B 4002

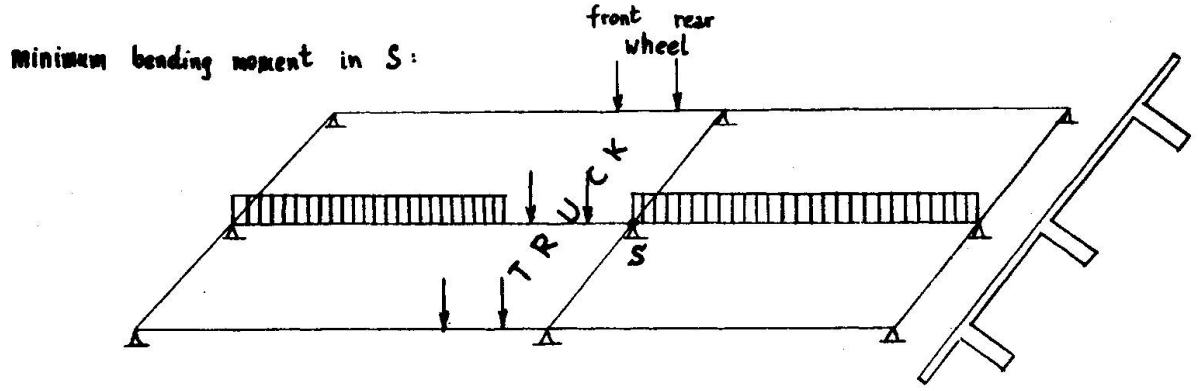


Fig.1: T-shaped beam: taking into account corresponding single loads by evaluating the lines of influence

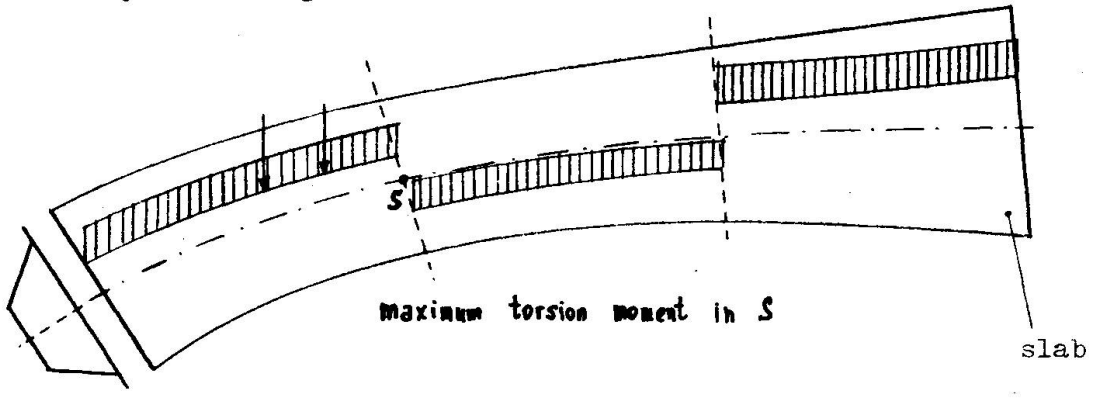


Fig.2: Box: selecting the governing live load mix (f.i. for MT)

Superposition and storage of the internal forces governing the states of construction and the final state under service load
Free cantilever states of construction
Timedependent development of prestressing forces, prestressing paths
Service load behaviour (bending stresses, shear stresses)
Ultimate load behaviour (ultimate moment, shear)
Stability of columns and the frame, 2nd order theory

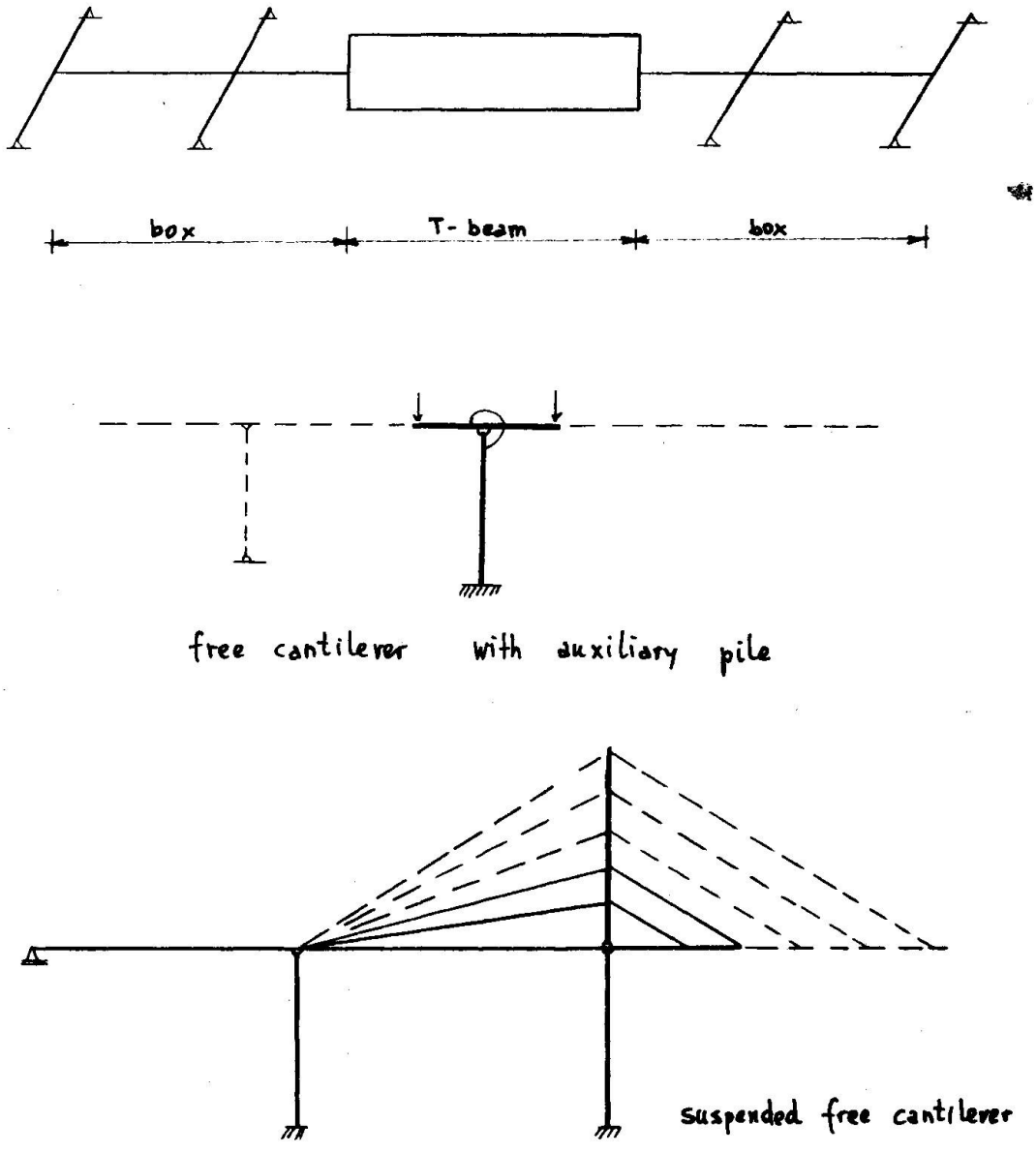


Fig.3: Examples of structural systems having been computed by the system

3. PROGRAMMING DETAILS

3.1. Programming language

The entire system was written in ALGOL. It consists of 36 programs with about 11.000 statements altogether. The choice of ALGOL was given by coincidence and this was the biggest obstacle for a greater improvement and proliferation. The mainframer do not comply with the ALGOL specifications in many ways, as you know, and machine dependent implementations can be found many times. The specific language used for the system was ALGOL (SIEMENS) 300. For the below mentioned machines a conversion was planned but not done due to the enormous work that would have to be put in:

IBM /370
CDC 3000,6000
UNIVAC 1100
SIEMENS 7000

The greatest incompatibilities were found in the file handling and in the paper peripheries, due to the weakness of ALGOL concerning the input/output procedures. As is well known the basic deficiency of ALGOL is the lack of well defined a n d h a n d s o m e input/output statements.

3.2. File layout

Fig.4 shows a makro data flow chart. There are 3 stock files:
- the GTV (geometry-, topology-, prestressing-) file
- the ESD (elastomechanic system data-) file and
- the MQT (bending-, shear- and torsion-) file.

The input for modeling the structure representing a specific state of construction (object structure), the load datas, the superposition commands and the commands for the evaluation of the lines of influence are to be considered as object datas. By doing so it is possible to build up the MQT-file stepwise by adding up the internal forces given in the specific states of construction and by choosing the live load mix leading to the extreme live load stresses.

The MQT-file consists of the following elements:

- dead load
- permanent load
- dead load creep redistribution
- prestressing
- prestressing creep redistribution
- max. live load bending
- min. live load bending
- max. additional load bending (wind, lowering of supports, temperature, earthquake)
- min. additional load bending
- max. live load shear
- min. live load shear
- max. live load torsion
- min. live load torsion

Each element consists of the 3 values for bending moment, shear force and torsion moment.

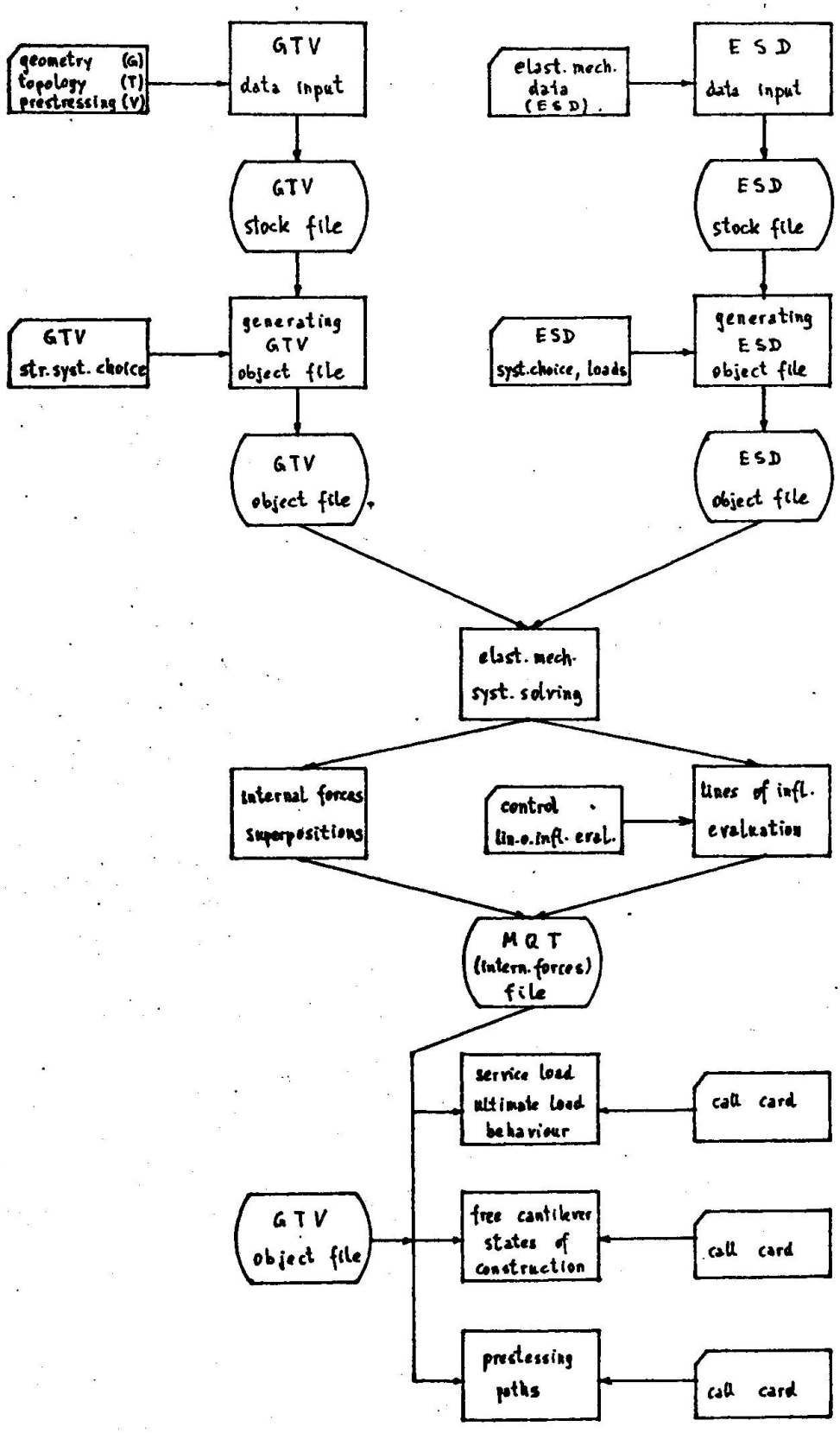


Fig.4: Makro data flow chart

The service load behaviour and ultimate load behaviour can be found easily by tapping the GTV-object file and the MQT-file taking into account the respective state of construction. In order to find out the maximum stresses under service load (for bending and shear), the calculations are made using the respective internal forces of the MQT-file. Fig.5 shows the makro flow chart and may give an idea about the flexibility of the system.

4. CONCLUSIONS ABOUT THE RELATION MAN-MACHINE IN DESIGN AND CALCULATION

4.1. Users' profile

From the beginning the system was developed for the use by 3 design departments at different cities of a large Austrian construction company. The 3 departments were staffed as follows:

Dept.1	Dept.2	Dept.3
7 university engineers	5 university engineers	6 university engineers
1o high school engineers	6 high school engineers	6 high school engineers
SIEMENS 3o4	SIEMENS 3o5	DIEHL ALPHATRONIC
closed shop operation	open shop operation	open shop operation

The SIEMENS 3o4 and 3o5 are middle computers, DIEHL ALPHATRONIC is a desk calculator (not running the system described here of course).

Dept.1 used the system seldom arguing that the closed shop operation does not allow a quasi-interactive handling of the machine. Furthermore was said the system does not allow for finding a quick and less accurate solution when working on a tender. That led to the development of a new small system on a NOVA 122o, a small computer, by dept.1. The NOVA was standing 2 floors above the SIEMENS.

Things were completely different in dept.2. Due to the open shop operation it was possible to modify the system in short time for specific purposes and it was used for every tender and for each structure that was under construction. The pressure to deal with the programs and to understand their programming motivated the engineers in dept.2 enormously.

Dep.3 was equipped with a desk calculator that could be used for rather small and subordinate calculations. Quick calculations for tenders were done at a near computer center in the same city. But some scruples were left that the preliminary calculations for tenders were not carried through confidentially. Therefore later on, the calculations were mostly done on the desk calculator, even if necessary with great inconveniences. Just difficult structures were sent to dept.1 to be run through the system.

Because the costs of developing such a software were high and the profit from the application could not be quantified it was decided to offer the use of the system to third persons. After some advertising there were some 3o customers, especially consulting engineers and design departments of other construction companies. They used the system many times more than the 3 departments altogether. The only restriction was that a customer working on a tender and simultaneously competing with the own company could not be served.. a matter of course.

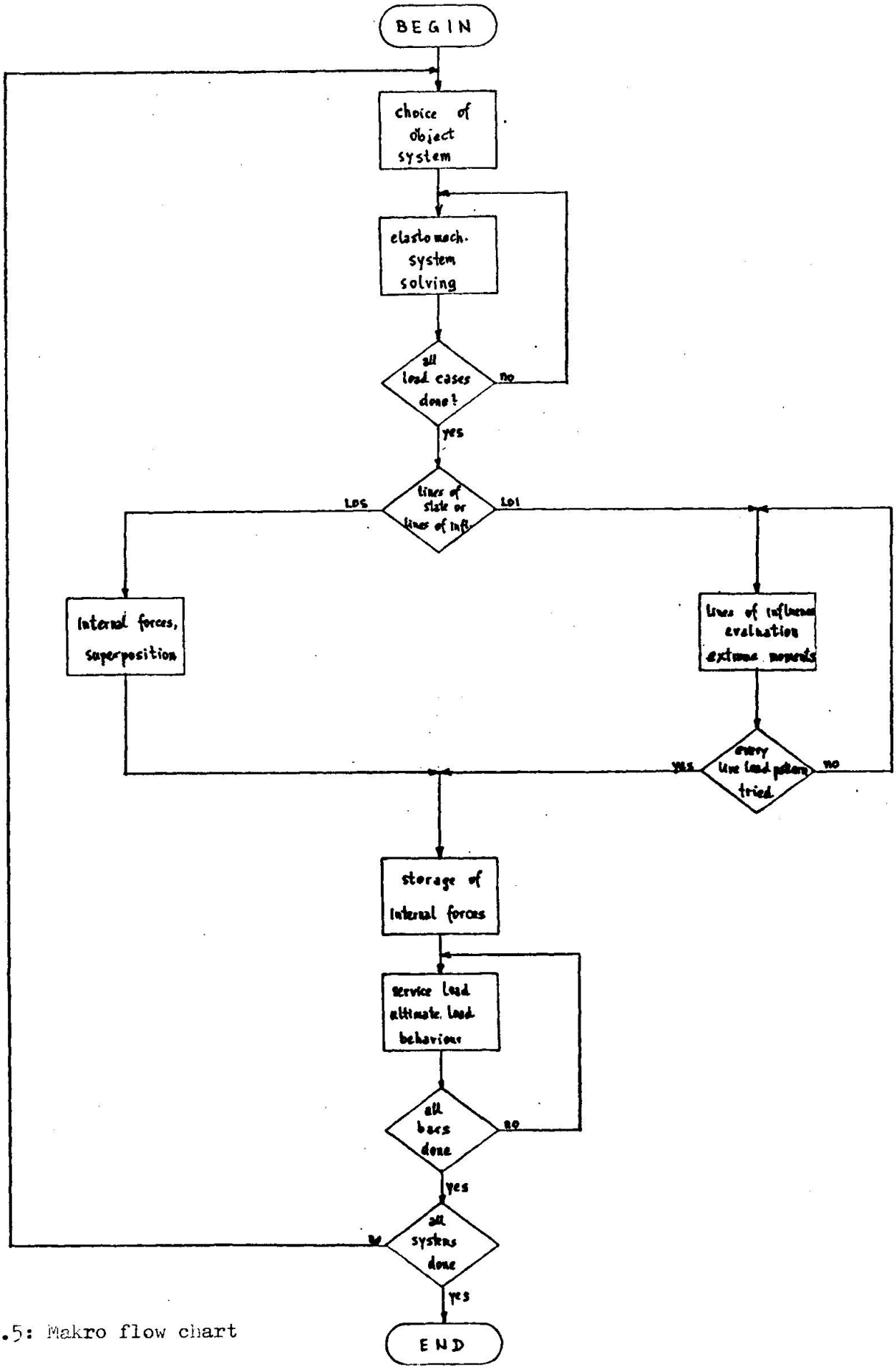


Fig.5: Makro flow chart

The users could be divided rather easily into 2 kinds:

The first kind of users was sending structural sketches, sketches of loads and the choice of materials (sometimes not even that) and left the choosing of the structural system and the setting up of the input up to the computer center. That required a deep understanding of structural behaviour from the computer man concerning the choice of the structural system, support conditions, rotational restraints and arrangement of auxiliary bars. This kind of user also did not bother to read the manuals and often times it was necessary to explain the results personally.

The second kind of users, the smaller amount, was sending input sheets filled up correctly and completely as was outlined in the manual. For this sort of people the input sheets were quasi the keys of the operator's console or the screen. They gave no problems at all.

The use of such a large and compound system required a high professional responsibility. The programs were tested before use widely that one can say almost no serious mistake occurred during the years. The mistakes were either wrong input or wrong file handling. In a covering letter (slip) to the results the customers were asked to check the input data and to control the results for plausibility. Errors of the computer center were undone by repeating the job without charge of course. A furthergoing liability or claims of compensation were excluded.

4.2. Conclusions

From the experiences that were made in 8 years of service we would derive the following rules:

On the interaction man-machine:

- The closer the computer comes to the user, the simpler the machine and the software can be handled, the more available the computer is, the better it can be used for the design and optimization of a structure. The small inhouse scientific computer is superior the terminal to a large outdoor computer.
- The larger and more compound a software system, the more reliable is a calculation under surveillance of qualified people, let us say people in a special service center. Finite element programs on small computers are wonderful but dangerous.

On the professional responsibility for calculations:

- A program without complete and detailed description by a manual should not be allowed to be used at all (lack of codes!).
- Input data have to be listed 1:1 (echo print). More complicate structural systems are to be plotted.
- Automatic plausibility checks help locating input errors or programming errors. Unfortunately they are found in few programs only.

REFERENCES

1. OBERNDORFER,W.: Brückenstatik mit EDV, MAYREDER-Zeitschrift 18 (1973), Heft Mai

Leere Seite
Blank page
Page vide

IABSE
AIPC
IVBH

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

On Dynamic Analysis of Thermally Cracked Concrete Structures

Analyse dynamique de structures en béton présentant des fissures d'origine thermique

Dynamische Berechnung von thermisch belasteten Betonbauwerken

A. CASTELLANI
Professor, Struct. Eng. Dept.
Technical University
Milan, Italy

L. FINZI
Professor, Struct. Eng. Dept.
Technical University
Milan, Italy

Summary
The safety analysis of a concrete structure requires taking into account the crak distribution, in particular when thermal and dynamic loads are present. A considerable spreading of the design parameters may occur and consequen tly an increasing of the building cost. Two broad categories of reinforced con crete structures have been investigated in this respect: shear walls of buildings and cylindrical containment structures. A mathematical model for computing their lateral rigidities is commented.

Résumé
Pour une analyse de sécurité d'une structure en béton il faut tenir compte de répartition des fissures surtout lorsqu'il y a des charges thermiques et dynamiques. Une extension considérable des paramètres du projet peut avoir lieu et en conséquence une augmentation du coût de la construction. On a examiné en ce sens deux grandes catégories de structures en béton armé: des batiments à parois verticales et des structures à réservoirs cylindriques. Un modèle mathématique permettant de calculer les rigidités latérales est illustré.

Zusammenfassung
Die Sicherheitsberechnung eines Betonbauwerkes muss die Rissverteilung berücksichtigen, besonders wenn thermische und dynamische Belastungen dabei sind. Es kann daraus eine entsprechend erhöhnte Anzahl von Entwurfsparame tern resultieren, was die kosten des Bauwerks beeinflusst. Zwei Haupttypen von Bauwerken aus Stahlbeton werden untersucht: Schubwände in Gebäuden und zylindrische Tanks. Ein mathematisches Modell, welches die later ale Steifigkeit prüft, wird kommentiert.

1. Premises

As cracks in reinforced concrete are more the rule than the exception, one has to think in terms of an anisotropic material when defining the stiffness matrix.

As to the flexural behaviour of beams and columns, such model can be supported by a large amount of both experimental and theoretical knowledge but this is not the case for a two-dimension structure, a part from the complexity of the involved calculus.

The need of reliable mathematical models for cracked concrete is typical for building shear walls, for containment structures and for chimneys. Their vertical sections carry only a limited amount of compression due to dead loads and so they are likely to be cracked by thermal or shrinkage loads. A limit case, finally is the one of precast large panel buildings, where in practice all the vertical members are shear walls and the construction joints, at the boundary of each panel, act as natural cracks.

Cracking could be avoided,but only through the adoption of a very dense mesh of reinforcing bars: for a 20 cm thick shear wall, a 12 mm bar each 10 cm both vertically and horizontally, for instances. This means an impressive consumption of steel (say abount 200 kg/m^3) for structures that, until now, have been built practically with no steel inside.

The fact is that shrinckage or thermal cracks do not mean neither the collapse of a structure nor the loss of serviceability of it. What is requested to the designer is simply to limit the width of the cracks with reference to the corrosion dangers or to the leakage specifications.

Thus when entering in the evaluation of the dynamic response of reinforced concrete structures, in most of the cases, one should think in terms of a cracked structure and should properly evaluate the loss of stiffness due to the cracks which obviously means an increased local deformability of the member under consideration.

To face the above mentioned situation one normally assumes suitable upper and lower bounds for the concrete rigidities and envelopes internal stresses related to either bound. The upper bound for rigidities is obtained by assuming uncracked concrete. The lower bound by minimizing the concrete stress transfer across the cracks. This procedure gives up to a spreading of the stress analysis results, and consequently to a possible increased building cost.

This procedure does involve not only sophisticated structures, like the containment of a nuclear reactor, but, implicitly, conventional structures too. Their design in fact is often made depending directly from the more severe situation relying on either the lower or the upper bound of concrete rigidities, which in general are far from the real rigidities. Examples of this spreading will be shown for the containment of a nuclear power plant, and for a building shear wall.

Both the ACI and CEB Codes |1,6| give suggestions on the Shear Strength of Reinforced Concrete beams taking care of the basic mechanisms of shear transfer that is:

a) Shear transfer by concrete shear stress (on the uncracked parts of the member).
b) Interface shear transfer (aggregate interlock through a crack).
c) Dowel shear (dowelling forces in the bars crossing a crack).
d) Arch action (as significantly possible in deep beams).
e) Shear reinforcement (stirrups and bent bars).

Special rules are not given instead for two dimensional members. In any case reference is alwanys made to the strength of the structural element but not to the deformability of it. That is if one has to build the stiffness matrix of a cracked two-dimensional structure is really in trouble.

On the subject an interesting document is nevertheless the ACI-ASCE Committee 426 Report n° 70-46 |16| as here many papers dealing with various aspects of shear strength and behavior are collected.

Most of the experimental work has been done on relatively small specimens and so the scale effect, going to full scale structure, must be evaluated. But, as we know, this is the only source of information on the stress-strain relation for cracked structures that we have at present.

2. Reinforced Concrete Containment Structures

Some of the leading engineering firms at the IV SMIRT |5,9,11,19| declared that to cope with the combination of thermal and earthquake loads, three separate dynamic assumptions are currently enveloped in the design of a containment structure:

1) undamaged wall;
2) vertically cracked wall;
3) both vertically and horizontally cracked wall.

Moreover, to cope with the combination of thermal load and internal pressure - generally included in the loading of a containment structure- the cracks are considered as involving the entire thickness of the wall.

The most rough way cracks can be included is by neglecting any stress transfer across the two surfaces of the crack but for the reinforcing crossing it. This means disregarding the mechanisms of "aggregate shear transfer' and that of "concrete tension stiffening". An overevaluation of the crack effects may occur, depending on the deepness of the crack within the wall. The kind of load also deserves its importance.

As an example, fig. 1 and 2 show some aspects of the dynamic analysis of the reactor building, at Caorso Power Plant. The primary containment wall has been modelled under two assumptions: undamaged wall and cracked wall. The cracks involve one half of the thickness, both vertically and horizontally. The comparison between normal modes shapes under these two assumptions is shown.

From these figures no meaningful difference is apparent between the two approaches, expecially for the first mode of vibration, consisting of a nearly rigid rotation of the structure on the soil. Significant differences on the dynamic behaviour of the structure would be apparent only if higher modes of vibration could be meaningful, but this is not the case of seismic excitation.

On the other hand, when seismic internal actions are superimposed to pressure and thermal loads, the combined effects result fairly apart each from the other within the two approaches. Let us in fact regard the M-N interaction domain of the horizontal section at the foundation level - fig. 3 - where the representative points are shown, as related to the two opposite approaches. Not only the reinforcing steel percentage but also its distribution is largely affected by the crack hypothesis and, therefore, by the way it has been dealt with.

About the distribution of the reinforcing, note that the Caorso containment used both an orthogonal (hoop, vertical) and inclined (diagonal) reinforcing system. The inclined system was designed to resist the tangential shear. Only one horizontal earthquake component, 0.24 g peak ground acceleration, was combined with a vertical 0.16 g peak acceleration. The absolute summation of internal stresses due to the two components was assumed.

This kind of reinforcing was common to all U.S. containment designed approximately 10 years ago. In recent times among the major goals of improving structural design for containment structures there is either to verify the need for inclined reinforcing or to provide information to substantiate their elimination

3. Shear walls of a Building

3.1. On the need of thinking in terms of cracked shear walls

Suppose to look at a shear wall 20 cm thick reinforced with Ø 10 mm each 20 cm vertical and horizontal corrugated bars near each face.

Let the characteristic strength f_{ck} of the concrete be 30 PMa and the one f_{yk} of the steel 440 MPa.

The steel to concrete ratio, longitudinally and transversally is:

$$\rho_l = \rho_t = \frac{A_s}{b.d} = \frac{0,78x5x2}{20x100} = 0,39\%$$

and the steel consumption (overlapping included) is about 70 kg/m^3.

If cracks must not appear the design shear stress allowed by the CEB Code is:

$$\tau_{Rd} = 0,25\ f_{ctd} = 0,25.1,36 = 0,34\ MPa$$

for plane concrete and

$$\tau^{*}_{Rd} = (1 + 50\ \rho_l)\ \tau_{Rd} = 1.195\ \tau_{Rd} = 0,406\ MPa$$

when taking advantage of the dowelling effect of the existing longitudinal reinforcement. The improvement therefore is less than 20%.

The above evaluated design shear stress in rather low as the value 0,406 MPa can very easily be overcome through shrinkage and thermal effects acting together with modest transversal loads

Infact if the floors can in some way avoid the design thermal elongations, the decrease of temperature needed to crack the wall is only:

$$\Delta t = \frac{\sigma_{ctd}}{\alpha . E} = \frac{1,36}{10^{-5} . 32.000} = 4,25\,^{\circ}C$$

and the corresponding crack width is

$$w = \alpha\ \Delta t\ l = 4,25.10^{-5}\ l$$

that is, for a 6 m long wall

$$w = 0,255\ mm$$

Therefore both the shear weakness of the uncracked structure and the sensitivity to thermal and shrinkage effects ask the designer to think in terms of cracked shear walls when designing them.

If one thus thinks in terms of a cracked wall, the allowable design stress goes from 0,406 MPa to

$$\tau_{Rd3} = \beta_1 \cdot \rho_t \frac{f_{yk}}{s} + \beta_2\ \tau_{Rd}$$

The coefficients β_1 and β_2 are a given function of $\rho_t = 0.39\%$:

$\beta_1 = 0,57$ and $\beta_2 = 2,04$

Therefore in our case we get:

$$\tau_{Rd3} = 0,57\ \frac{0,39}{100}\ \frac{440}{100} + 2,04.0,34 = 0,85 + 0,69 = 1,54\ MPa$$

and the total design shear strength is now four times greater than the one of the uncracked wall.

The contribution of the vertical reinforcement appears to be of the same order of the one of concrete. It must be noticed, moreover, that, according to the CEB Code, the longitudinal reinforcement does not affect the result.

The crushing of the wall due to diagonal compression is surely avoided as the corresponding design shear stress τ_{Rd2} is very high:

$$\tau_{Rd2} = 0,30\ \frac{f_{ck}}{\gamma_c} = 0,30\ \frac{30}{1,5} = 6\ \ MPa$$

3.2. Influence of cracks on the lateral stiffness

Not only lateral resistance, but also adequate lateral rigidity has to be proven for shear walls. The hypothesis of cracks weakening the wall's lateral rigidy has been not normally taken into consideration. If it were, it would significantly alter the structural organization of the building and would give rise to an overriding increase in the design stresses in the columns. In fact, when a given shear wall distribution against lateral forces is assumed in the design of the building, the dimensions of adjacent beams and columns are largely based on the presence of these shear walls. Usually these shear walls provide almost all the resistance to lateral shear; in this case for thermal load, current design practice is based on the assumption that, if present, it does not weaken the lateral rigidity, or, better, it must not damage the shear walls.

In this case thus the upper bound rigidity is assumed for the wall, leading to a conservative reinforcing steel distribution.

An opposite approach has been pursuited for precast panel buildings in presence of vertical joints. Notice that when vertically cracked, a shear wall is likely to act as a monolitic cantilever if, along the vertical section, an adequate shear transfer can be accommodated, in spite of the presence of cracks. Such transfer can be contributed, in

the case of a vertical joint, by shear friction, and by the dowel action provided by horizontal reinforcement, if present.

Nevertheless, in the practice, this shear transfer is neglected unless a special joint design is performed. So that the two adjacent panels are modeled as independent cantilevers and the stiffness matrix is obtained through a lower bound approach. Fig. 4,5,6 and 7, on the other hand show that although if the shear transfer is associated with a finite slip between the two adjacent panels, nevertheless the structure behaves as a monolithic cantilever.

A shear stiffness reduction of the joint by a factor 0,1 is allowed without altering the lateral rigidity K - fig. 5 - by more than 10%.

Some experimental tests on this subject have been reported in |2|.

4. Incorporation of the Available Experiences in Mathematical Models

4.1. Shear Transfer

Several researches have documented experimental studies on shear transfer across open cracks in precracked concrete |8,11,14,15|. These experiments confirm that combined dowel action and interface shear transfer is an efficient mechanism for shear transfer at a slightly open crack. Mean while, under cyclic loading, bond deterioration promotes relative displacement at the crack faces, causing aggregate interlock cracking and dowel cracking parallel to the longitudinal reinforcement: a loss of stiffness thus occur |15|.

According to |19| , the load - slip characteristic at the crack, including degradation during cyclic loading, are sufficiently known so that they can be embedded in a nonlinear analysis to predict the effect of deformations at the crack on the dynamic response.

As to authors' knowledge, at least one computer code, commercially available and of a general purpose, includes a "concrete model": Adina Code |3|.This is able to switch from the isotropic behaviour of uncracked concrete to the orthotropic behaviour of cracked element, and can take into account any kind of shear transfer, but its degradation during cyclic loading. Only a few special-purpose computer codes do this: for instances fig. 8 represents the shear stress vs. shear slip at crack which has been incorporated into the dynamic response analysis program |19| .

Let now remind that the main parameter of the dynamic response obviously is not the maximum shear stress locally transmitted. Also the shear strain accumulated during half a cycle is no more of interest, because, in general, the motion is reversed during the next half cycle and so part of the strain may be recovered. The total shear strain cumulated across the crack at the end of the dynamic excitation is among the important parameters.

To this purpose in the authors' opinion at least two comments may be suggested, about the shear stress - shear slip curve.

1) Any idealized curve as that of fig. 8, is necessarily symmetric around both axes, while symmetry is more or less lacking in practice. Moreover, idealized dynamic loads are often symmetrized too , see for instances artificial time histories representing seismic excitations. Under these premises the net strain cumulated during a dynamic excitation may underestimate substantially the real strain, and may be even zero.
2) As to vertical cracking due to pressure or thermal loads in a cylindrical containment, the numerical approach shows cracks as straight lines, but in practice they are fairly irregularly shaped,

so that a slip surface does not appear. Any irregularity acts as the aggregate interlock, providing a further contribution to shear transfer.

In the first case, therefore, the random nature of the crack leads to a more severe strain than expected by analysis; in the second case it leads to a less severe stiffness loss than expected. In both cases, in conclusion, however simple be the structures to which such models apply, nevertheless the sign of the involved errors is quite unpredictable.

4.2. Concrete Tension Stiffening

Consider fig. 9 where a concrete specimen is shown of length 1. The stiffness $F/\Delta l$ is not only contributed by the reinforcing bar, but also by the concrete itself, however cracked, provided that a suitable bond allows stress transfer from the steel to the surrounding concrete. Let call this contribution "concrete tension stiffening".

The existing practical proposals for dealing with this phenomenon are founded on tensile tests on bars surrounded by concrete, and checked by bending tests.

In particular, according to Beeby's tests on bars surrounded by concrete |4|, the tension stiffening can be represented by an average stress distribution in the concrete, linearly distributed from a value of zero at the neutral axis and a value of 10 kg/cm^2 at the centroid of tension steel. This distribution does not change sensibly by increasing the applied stresses on the cross section. Such a distribution is noticeably non linear versus the applied forces, and the average strain for steel surrounded by concrete can be reproduced to a given stress level by assuming a suitably reduced modulus of elasticity for concrete in tension, apart from the presence of reinforcing steel.

The tension stiffening is in general of a limited effect in the moment-rotation diagram for a beam. It is of noticeable importance in the problem here considered only when axial stiffness of a beam or a wall is concerned, mainly for underreinforced concrete.

When the crack width has to be evaluated sophisticated friction elements are required to represent the steel-concrete slip: see for instance |7|. This element connect two separate nodes occupying the same physical position: see fig. 10.

This second approach may be applied to analyse local situations of small extent: in fact the mesh size for truss elements representing steel and for plane elements representing concrete needs be very refined otherwise the constant stress truss element cannot reproduce suitably the mechanism of stress transfer from steel to concrete, and so cannot reproduce adeguately the structure stiffness. Typically the mesh dimension needs to be 1/10 ÷ 1/5 of the crack separation, i.e., of the order of 10 cm in a large number of cases.

5. References

|1| ACI Standard: Building Code Requirements for Reinforced Concrete (ACI 318-71).

|2| ALEXANDER C.M., HEIDEBECHT A.C., TSO W.C.: Cyclic Load Tests of Shear Wall Panels, V WCEE, Rome 1973.

|3| BATH J.B.: ADINA - A Finite Element Program for Automatic Dynamic Incremental Non-Linear Analysis, Mechanical Engineering Department, Report 82448-1, Massachusetts Institute of Technology, 1975.

|4| BEEBY A.W.: A Study of Cracking in Reinforced Concrete Members Subjected to Pure Tension, Technical Report CCA, 1972.

|5| BUCHERT K.P., BALLARD T.A.: Analysis of Reinforced Concrete Containment Vessels Considering Concrete Cracking, IV SMIRT, Structural Mechanics in Reactor Technology, San Francisco, 1977.

|6| CEB-FIP Model Code for Concrete Structures, Bull. n° 124/125 E, april 1978.

|7| CEDOLIN L., DEI POLI S.: Non-Linear Plane Stress Analysis of Reinforced Concrete by the Finite Element Method, Studi e Rendiconti, Costruzioni in Cemento Armato, Vol. XIII, 1976.

|8| CELEBY M., PENZIEN J.: Behaviour of Reinforced Concrete Beams under Combined Moment and Shear Reversal, Preliminary Report, IABSE Lisbon Symposium, 1973.

|9| CHANG H.S., ATKATSH R., HERTING D.N.: Nonlinear Analysis of RC Structure with Interaction of Thermal Cracking and Mechanical Load, IV SMIRT, Structural Mechanics in Reactor Technology, San Francisco, 1977.

|10| CLARKE J.L.: Thermal Stress Problems in Offshore Structures, Offshore Structures Conference, Brighton, March 1978.

|11| GREEN D.F., JOHNSON T.E.: Design of Concrete Containments for Tangential Shear Loads, IV SMIRT, Structural Mechanics in Reactor Technology, San Francisco, 1977.

|12| LAIBLE J.P., GERGELY P.: Nonlinear Dynamic Response of Cracked Reinforced Concrete Nuclear Containment Vessels, Nuclear Engineering and Design, vol. 30, 1974.

|13| LEWICKI B., PANW A., Joints, Precast Panel Buildings, Planning and Design of Tall Buildings, T.C. 21, vol. III, 1972.

|14| MATTOK A.H.: Shear Transfer in Concrete Having Reinforcement at an Angle to the Shear Plane, ACI SP 42-2 "Shear in Reinforced Concrete" Detroit.

|15| PAULAY T., LOEBER P.J.: Shear Transfer by Aggregate Interlock, ACI SP 42-1 "Shear in Reinforced Concrete" Detroit.

|16| Shear in Reinforced Concrete, vol. 1 and Vol. 2, ACI Publ. SP 42, 1974.

|17| SMITH J.K., GERGELY P., WHITE R.N.: The effects of Cracks on the Seismic Analysis of Reinforced Concrete Nuclear Containment Vessels, Department of Structural Engineering, Report N. 368, Cornell University, April 1977.

|18| WHITE R.N., GERGELY P., LAIBLE J.P., FAJARDO O.H.: Seismic Shear Transfer Across Cracks in Concrete Nuclear Reactor Containment Vessels, V World Conf. On Earthquake Eng., Rome 1974.

|19| WHITE R.N., GERGELY p.: Design Considerations for Seismic Tangential Shear in Reinforced Concrete Containment Structures, IV SMIRT, Structural Mechanics In Reactor Technology, San Francisco, 1977.

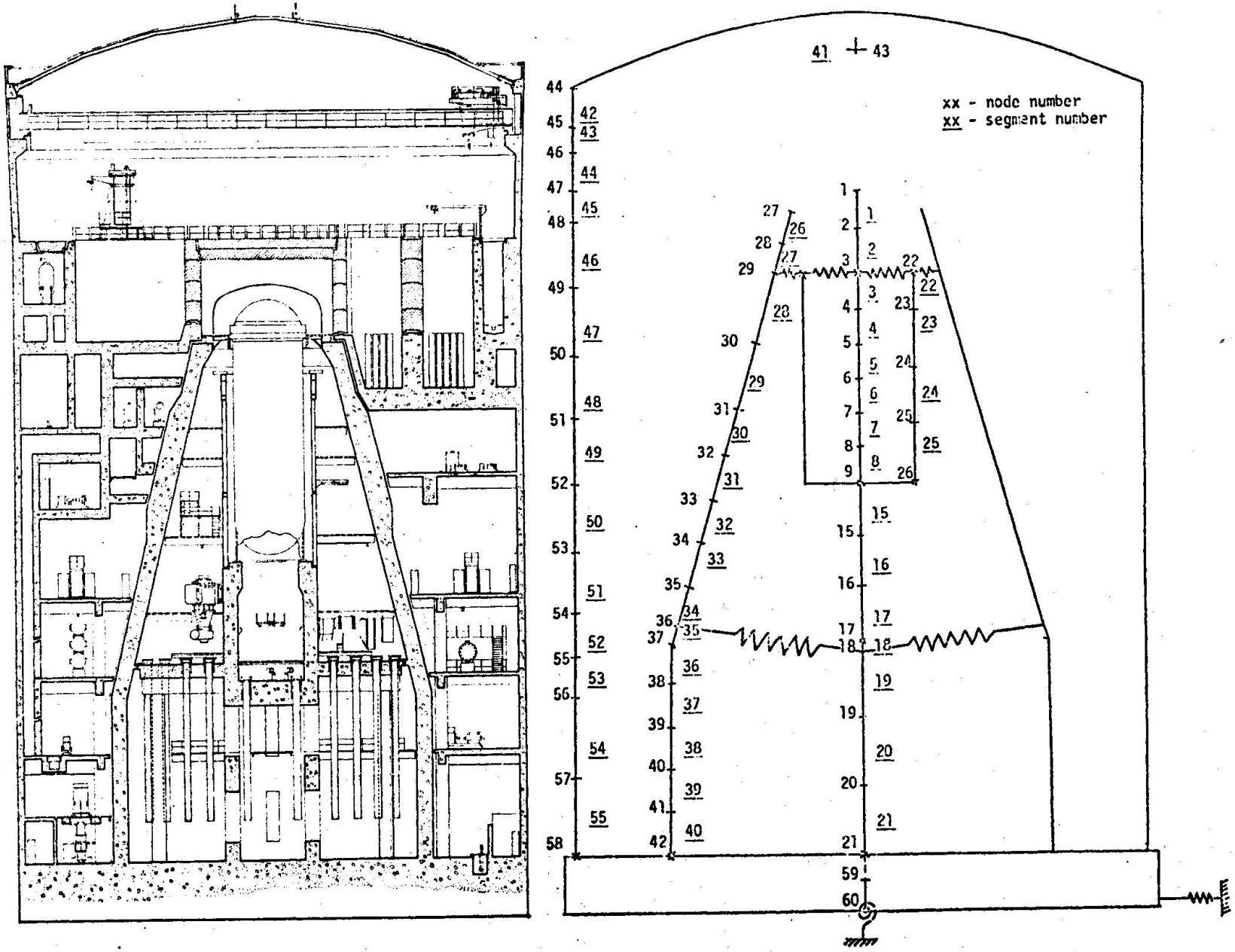


Fig. I Caorso reactor building:model for dynamic behaviour analysis.

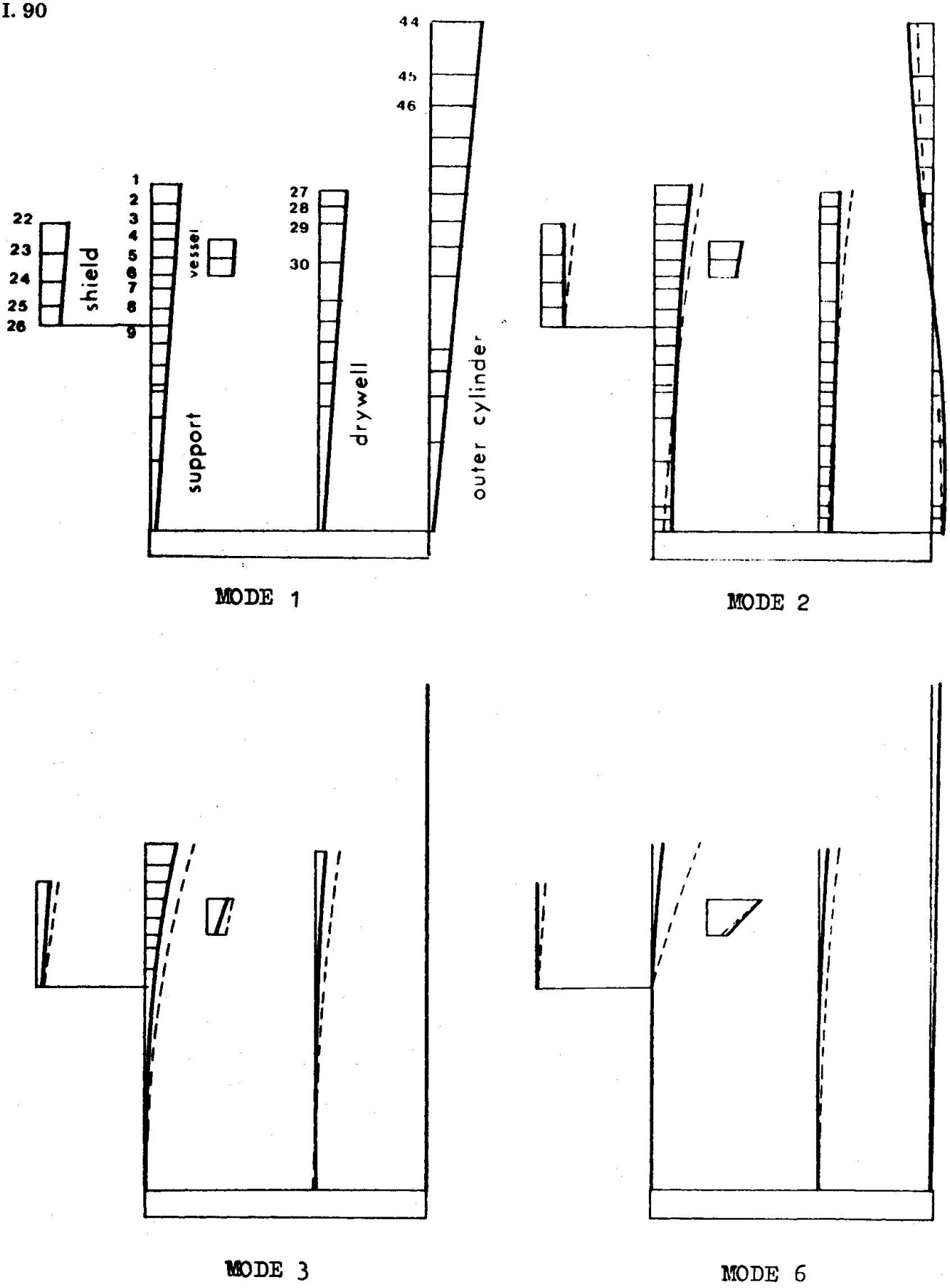
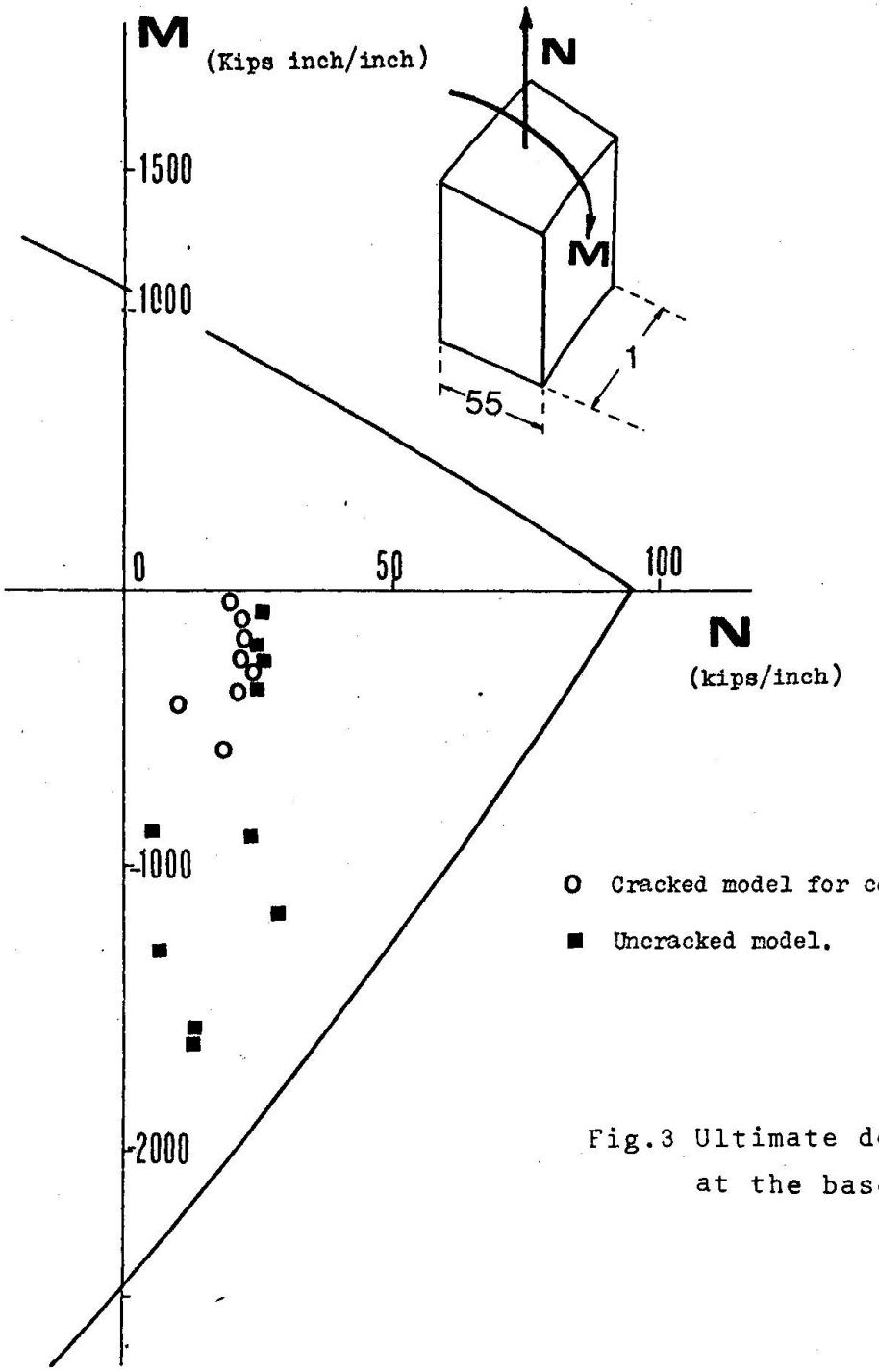
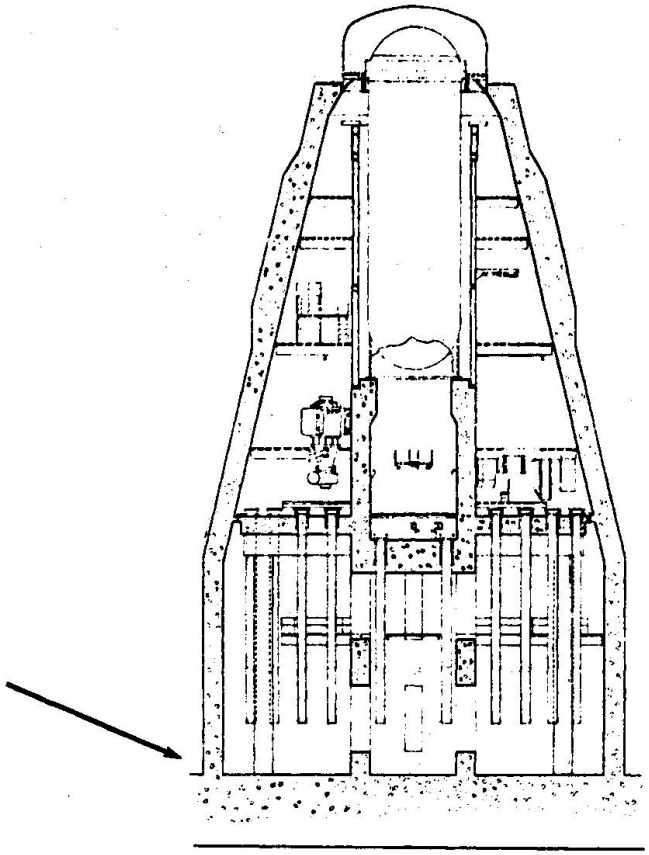


Fig. 2 Normal modes shapes of Caorso reactor building. Continuous lines refer to uncracked primary containment. Dashed lines to cracked primary containment.





o Cracked model for concrete;

■ Uncracked model.

Fig.3 Ultimate domain M-N of the horizontal section at the base of the primary containment.

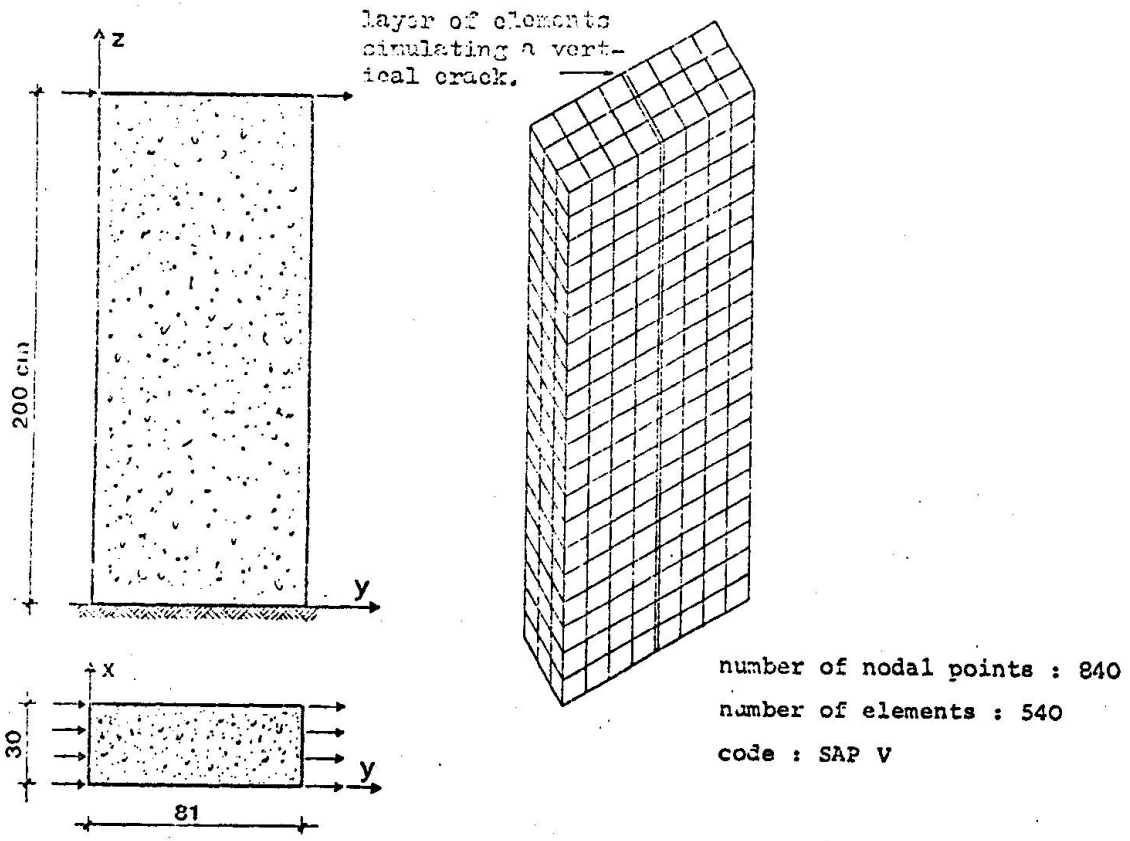


Fig. 4 Numerical model for the shear wall

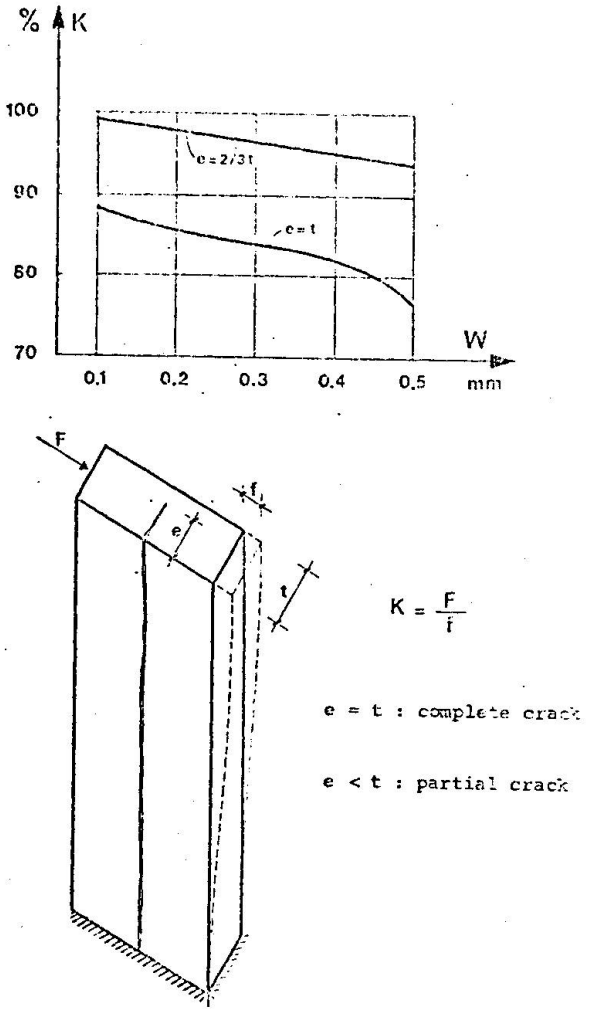


Fig. 5 Shear wall rigidity as a function of the crack width (aggregate interlock is considered).

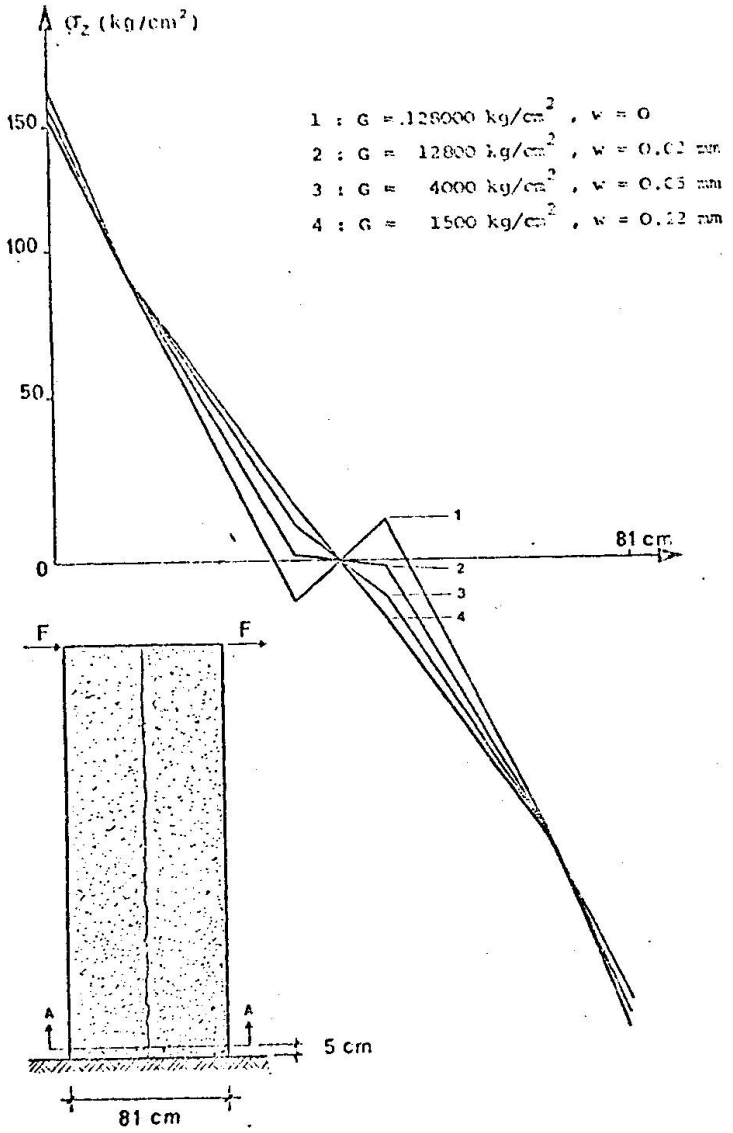


Fig. 6 Vertical normal stresses on section AA: for comparison , if no shear transfer is permitted through the crack the maximum σ_z is twice the peack stress of the uncracked wall.

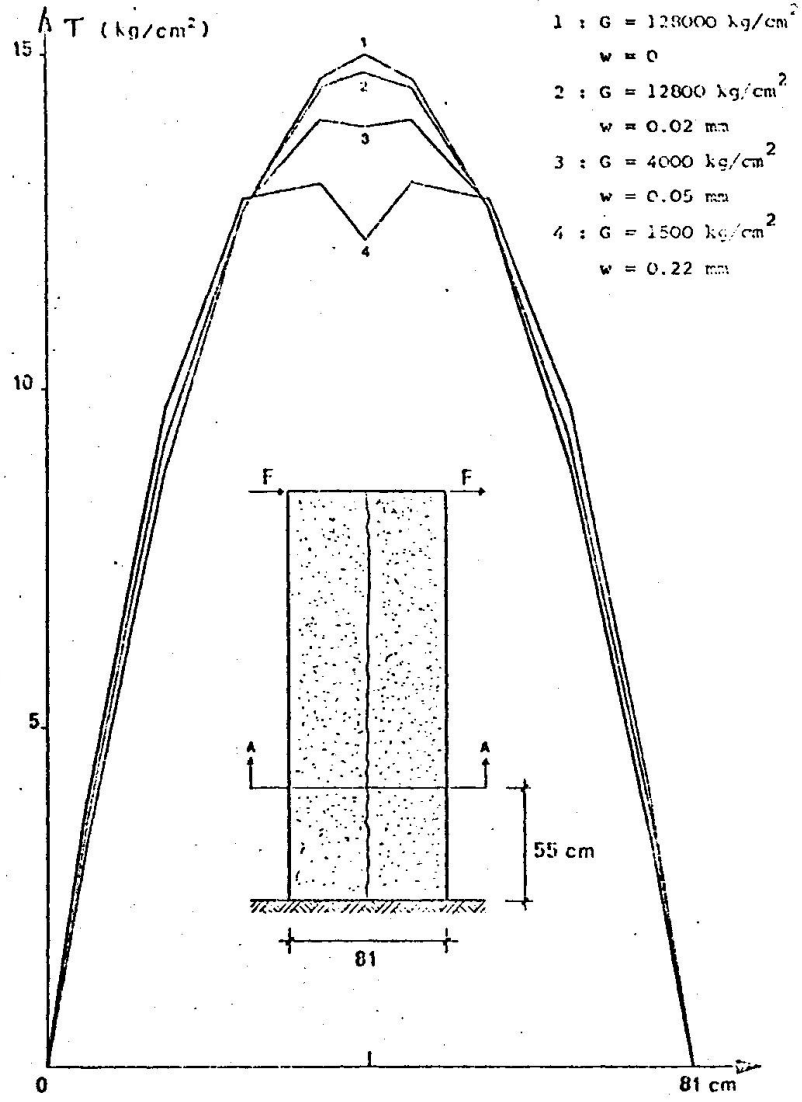


Fig. 7 Tangential stresses on section AA as function of the tangential modulus of elasticity G assigned to the layer of crack elements.

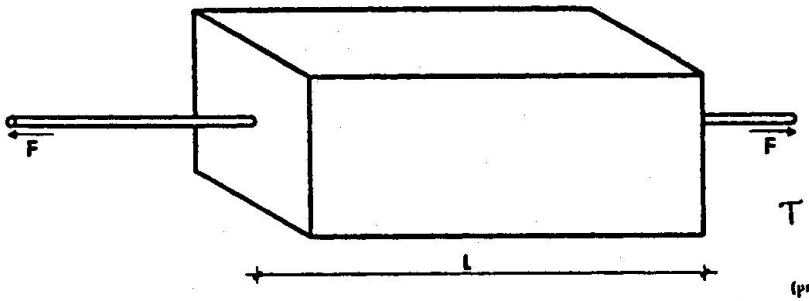


Fig.9

Fig.8 Idealized shear stress-shear strain relationship.

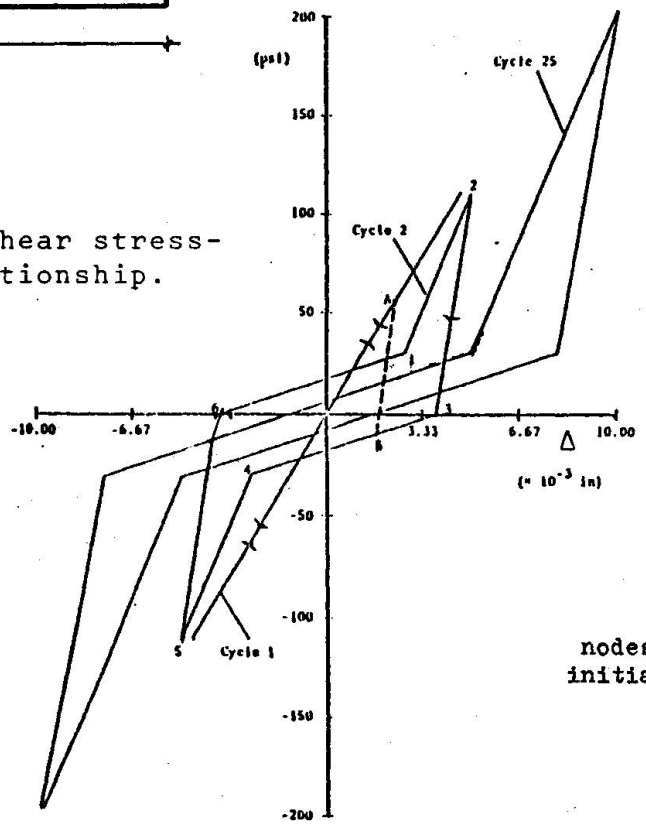
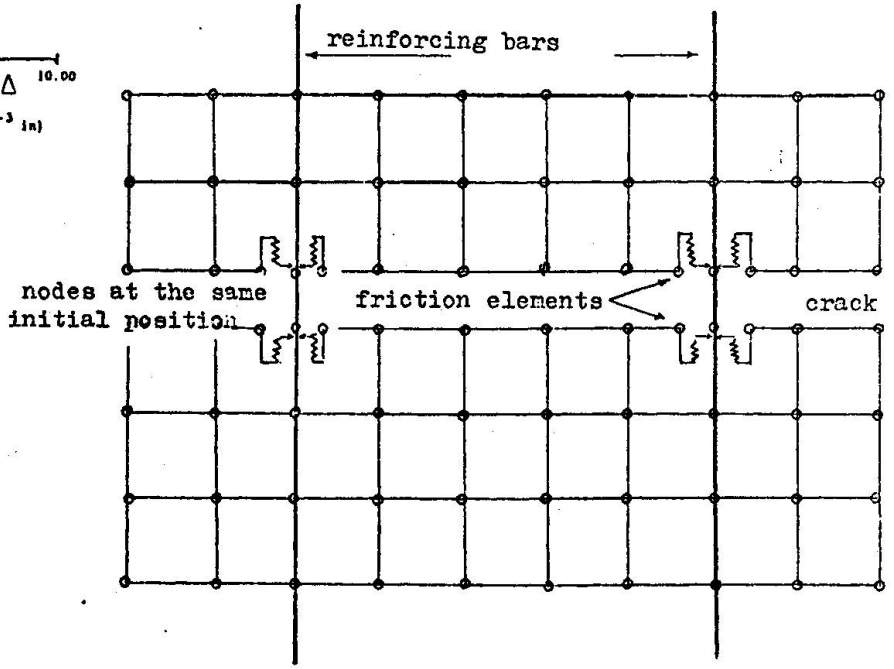


Fig.10 Simulation of a crack and of the bond slip between steel and concrete.



I SESSION

DISCUSSION

August 30, 1978. Morning.

Chairman: BOKELER (German Federal Republic).

BOKELER - Now we start the discussion. I hope we have a lot of sentences and statements in the last few papers, that we have a very interesting discussion. Please, if you start your contribution to the discussion, say your name first, ladies and gentlemen, please.

FANELLI - I was very interested by the contributions by Dr Pfaffinger and Dr Anderheggen and I have to ask both of them a question which is related to what each of them has said. Dr Pfaffinger showed us a very complex example of a complete analysis of a structure with a discretization having many thousands of loads and so on. Now, this kind of analysis is conceivable only for final design. In the stages of preliminary design, we should rely on a more simple idealization of the structure in order to be able to do many runs at a reasonable cost and introduce alterations in each reasonable final design. This is, other types of programs come in, related to much more simple structures, of such, for instance, the in-analysis program that was illustrated by Dr Anderheggen. So I asked, in fact, to start the discussion about how should we view these two kinds of approaches, one for the preliminary stages of design and the other for the final stages. Clearly, we cannot use the same approach in the two cases; we could not use a very complex analysis involving many thousands of equations in a preliminary stage, for we are bound to make many alterations and many runs of the same analysis over and over. So, it seems to me that one of the points we must focus on is a distinction between methods for preliminary design, which must be cheap, easy to run, fast and very simple for the user, and methods for final analysis, which can be as complex as needed.

PFAFFINGER - I perfectly agree with your statement and, of course, one did not start with this big analysis from the very beginning. As a matter of fact, I shall see some of the preliminary analysis we did, so the other preliminary analysis done on frame structure is very simple, just to get the problem in hand and see what is going on.

The same, indeed, is true for the evaluation of the results, to check them, so that also small problems run on a computer to verify that the general flow of forces is correct. I might stress out , in my opinion, you always should start with the second analysis first. Usually it is hard to set up the conceptual model, so you could approach the problem of computer aid, starting with the simple conceptual model and then looking for the results of a more adquate model.

ALCOCK - I am delighted with Dr Anderheggen's thesis of the syntax diagrams: the concept of this describing program with short user's manuals is a subject which can be dissociated from linear analysis, and it is on that aspect that I have a contribution here. We also believe very strongly in syntax diagrams and we re_ wrote the manual of the program STRESS using them. The result, I am happy to say, is a sort of 30 world-wide systems, including one in Austrialia, and also in a similar way, we have a tendency to find another similar system in France and another in Brazil, where we took the common language and an afternoon' s work to convert it from English to Portuguese and from English to French. I had valued Dr Anderheggen's comment on that matter because it seems to suggest some difficulties with STATIK in this respect. We said we had no trouble. And inciden_ tally as far as 40/30 world-wide system and we have no trouble. The organization consists of myself and my partner and one 18 year-old boy: that is our organization and there should be no trouble in supporting this. We do have a couple of com_ puting equipment on which to do it. Now, I really must come in on the system GENESYS, which is an English system. We have syntax diagrams very similar to the GENESYS ones. So, I believe that is good thing to describe things this way. I have really got one more thing here, i. e. to understand that machine independence. When we deliver GENESYS, we deliver it simultaneously on 5 different ways, so this can be put all in FORTRAN; and it' s a pity, I think, that STATIK was not written in low level and has not be tempted by the wonderful facilities of this equipment; but perhaps it would not be too difficult to think now to turn it back into elementary FORTRAN. A final word about output which was a remark to start the output as just a list of things. The output from GENESYS is automa_ tically put into a form which conforms to the syntax diagrams and can be reinput. We did not use the term "integrated system" at all - we used the term "integra_ ting system", so that if you wish to communicate from one subsystem to another subsystem, your job has to use GENTRAN to define the syntax diagrams of the output of one subsystem. You then automatically get the input to the other. So I think integrating system is better. Welcome are the comments and the answers on what you think of this approach system.

ANDERHEGGEN - About the languages, this will not be an afternoon work, but it may be a couple of weeks work. It would not be too difficult, expecially the input would not be too difficult (the output could be more difficult).
In the syntax diagram, I always used the first letter of each work, which makes it possible to shorten each word down to the first letter. Why so? Because we run into some difficulties sometimes, but this was a good decision, so if you have to change language, it is easy to change a little bit of it. The reason why we chose, I explained a little bit better in my paper, perhaps because we do not feel to be able to support a large work. At the University, as you certainly know, people are changing all the time. We do not have a commercial organization of this kind, we have as little a commercial organization as possible. We would have liked to write the program to get the ideas and then leave this to somebody else, which might be different from what you are doing. We are not really making money . . . we leave the making of money to somebody else. Of course, our program is successful only if money can be made out of it, this is clear. So, the problem of language is a rather minor one.

In Switzerland, there are people speaking French and other people speaking German . Integrating or integrated may be not too well aware of the GENESYS. I thought , however, this was a kind of off-spring of ICES because they have still some idea which is similar, haven 't they? You have a Gentran, which is something quite strange: I am not sure how it works. What I did not like in ICES is the idea of working with the same data base for extremely different applications like structure analysis or project management: this is something which is inhuman. It would be interesting to speak about integrating system. Of course, it makes a lot of sense to have a program producing output in a form which can be read and also can be used as an input for the next program.
Another program concerns program portability. We have used a CONTROL DATA computer and it is a big temptation to pack a lot of information on one word; then when you go to a machine with smaller word length, you run into trouble: this is the main reason. Another thing is also that CONTROL DATA has quite an exotic version of FORTRAN, which is quite different from IBM. With the new FORTRAN standard coming out, some of these problems might be solved. We will try in the future to write programs which run on smaller computers, medium-size computers for this kind of purpose.

DUTERTRE - I would like to thank Dr Anderheggen for his talk about syntax diagrams; I think if we could, in this colloquium, at least agree that syntax diagrams can be used by everybody, I would ease every one and enable you to use the other' s program. Everybody is being talking about his program, and no one is being talking about someone else using his program, and that is a bit tricky, unless you have a common way to express how to get in the program, and syntax diagrams are one way to express how to get in.
I have particularly appreciated the fact that your work started from STRESS and worked up from there. Following exactly the same pattern, the new STRESS, we have used another program called TITUS in France, which is very similar, as in the input we put the two together, in order to get syntax diagrams compatible, we worked from there and used a common syntax diagram for all our programs. I insist on that: if we could agree that syntax diagrams including simple rules can be used by everybody, there would be a big step forward .

MILSTON - I have been working as design engineer for about 30 years, which means that half my career was before the computer age, and I was really interested in Dr Pfaffinger's paper, where he gives requirements of structural engineering programs and gives conceptual analysis model of the real structure, the representation of results, the interpretation of results.
They are absolutely identical to those we used in structural design before computers were even thought of, and I think it is practically the same as Prof. Hardy-Cross of Main Distribution published in his work in 1932, nearly 50 years ago. I find exactly the same terms: first of all you have a real structure, then you have a conceptual model, then you do a numerical analysis, which gives a solution; then you represent these results and then you interpret the results.
The question is, of course, if there are essential differences between his requirements of structural engineering computer programs and requirements of structural engineering design.

PFAFFINGER - Thank you very much for your comment. I first have to say: I do not know the book by Prof. Hardy-Cross; secondly, if you read the introduction of this paper, I try to stress that the situation now has basically not changed for the structure engineer, having the computer and not having the computer. The only difference is that what we did by hand in the old days, we are doing now on the computer. We do it with more precision, we are able to set up more sophisticated models and we are able to approach reality closer than we could do by hand. The difference between the old days and now really is this: we are enabled to set up more realistic models, bigger models, more sophisticated models than we used to solve, but in the general flow of the work and in the general approach of the work, there is no basic difference. If you read the introduction, I have tried to stress this point.

BOKELER - Is there any other question to the statement "using syntax diagram is a way of standardization" ? Any question to that? Are you sure that everybody knows that a syntax diagram is what we mean? Would you like to say something Dr Pfaffinger?

TOMINO - I would like to ask the real meaning of syntax diagram. Well, I do not know if it may hold some kind of ambiguity, but many people are using that in Industry. The definition is quite odd to myself too.

BOKELER - Can you give a sort of definition, if possible?

ANDERHEGGEN - There are different kinds of syntax diagrams. Very often recursion is used, i.e. there are symbols which represent another diagram and there will be other symbols representing other diagrams, and these diagrams can contain themselves ... this is complicated. Now, the one we use, I think it is rather simple: you just have to follow the arrow and from there it gives you the allowable sequence of input data. So, by following the arrows, you know what the sequence of input data is, but you also see immediately what the program can do and which data are needed for instruction input. Reading the input description of a program which is working with this kind of diagrams, you see that all the input structure of tridimensional frames, very general with curved members or straight members, can be completely described in any detail on one page, for the whole STATIK program. You describe the input which you need for the preparation of your input card, and you also see immediately what the program can do, because there are just a few words, which tells you what it is all about.

BOKELER - The syntax diagram is a sort of checking list, with some sort of definitions for the language. It defines the syntax of the language but in the same time it shows really what you can do with the program.

TOMINO - To my understanding, the syntax diagram is a kind of diagram to define the control of the program: this is really a useful working from a program to another one. Of course, I hope to see in the future that I can use some other

people ' s program very easily, but for this I need some tool which makes possible to do it. This could be the use of a common language. Another could be the creation of a program, a system or a module for consulting program, providing so a kind of communication between one user and another.
In your paper is said it has a kind of memory manager, I mean, a kind of program which manages the numbers on the basis of high memory or auxiliary memories. If we can have such a kind of tool which can manage their transfer, then we can realize such kind of communication.

ANDERHEGGEN - May be I make it somewhat clear. We are talking about users' program, from the point of view of the users, since it really does not matter in which language the program is written. There is a question of communication between a user who doesn't know anything about programming language and the program. Now, our program is not such that you can say: use half of it and then write your own program and use the data - the data are all ready substantially somewhere, - but we are not going to tell you where, so you have to use our program as it is. I think it is a different level we are talking about; I am talking about engineering, not about the question of programming.

ALCOCK - I come back on the syntax diagrams and the levels at which they are used: the basic language is reasonably able to use 25 statements. You can draw syntax diagrams to find the whole of the basic language on one sheet of paper; it's one syntax diagram. Following the arrows through print you can write the names and the variables separated by commas or semi-colons, and this is shown by diagram you have to look. So, if we do it at the language level, we can also do it at the level as illustrated in our publication, showing how to prepare the different instruction programs, the same sort of syntax diagrams of the same kind of rules, but there is no reason why one should stay out and not permit people to get down one level from preparing the data to define the storage of data in the store, by means of the use of the syntax diagrams, whereby you can get them out and transfer them.

ANDEREGGEN - We also have another program called FLASH, for general plate- bending, plate-stretching and shell analysis; where the whole input is described in just two pages with a very compact syntax diagram of this kind; it' s just an elastic shell program, very general, and the whole thing is described in two pages.

SCHWARZ - I would like to give a short comment on this. The question was that every input user program should be standardized by such a type of syntax diagram. I am not quite sure whether this should be very good or not, since syntax diagrams are excellent methods which run in defined environment . . .
but if you have more introduction between the user and the program, this type of input should not be, I think, the best one at all. You might use many techniques to get input and direct the program what it should do, and so I think standardizing of these things should not be the best way to promote program usage at all.
I am afraid that syntax diagrams are rather frustrating for people standing away from computer usage and not familiar with the methods of computer programming.

BOKELER - Mr Schwarz, could we close this first statement? It is so important because you said these syntax diagrams are not usable for an average program. Can I say this in a short abbreviation? Syntax diagram for a normal batch program is too complicated: it might be better to use manual techniques, or, say, input sheets or something like that - is it correct ?

SCHWARZ - I just want to mention alternatives to your own interpretation. Expecially in the field of interactions, there are surely other methods, better than syntax diagrams.

ANDERHEGGEN - I don' t completely agree with you, because even when you use interaction time-shearing system, either the computer asks a lot of questions, the coordinates of point number one, and you write them, the coordinates of point number two, etc. ... Either you have to answer the question of the program by one single piece of data - which might be nice but I don' t know if it is the best way of doing it - or then you have some more less complex input statement, which have to be described somehow; so I think that a reasonable way of having a time-sharing input system is to write the whole line with a certain number of different data, and the syntax of this line can be described by syntax diagram, then the program checks the whole line and if it is bad, you have another line. The alternative, if you have a standard format, is this: you fill in the sheet with the pencil where your data have to come and then your secretary punches it. First, when you use a terminal, I think it is a very bad way of having the computer know what you mean by the position and the line and I think a Secretary can even understand our syntax diagram and what you mean by this syntax diagram.

BOKELER - Is there any other question or short comments? As you say, the only alternative to syntax diagram is coming to a formula; I do not think it is correct. There are several different methods.

TOMINO - In this discussion we all agree to come to a standardization in this problem; if this standardization is only on one page or two, it must be written, of course; if we are writing programs, we have to write also the manuals, we have to declare the used algorithms only if we accept that we can use such one or two pages for everyday' s use. The main result of this discussion is understanding the standardization problem. Maybe we come to some clearer and more concrete remark in future acknowledgments.

BOKELER - Another question on this statement should become towards standardization on the input form, on how to write a user's manual, how to write those sheets - is there any contribution to that? I am reading a paper which is describing an entirely different way of describing programs. I believe in syntax diagrams, it' s a marvelous thing, but I hate to think of reducing these as the only way of doing something. There are other ways for more complicated syntaxes.

DUTERTRE - Maybe we could simply say that we need a standardization of users' manual, and so we can understand others' manual. But we said we need standardization. Are we convinced that we are able to standardize? I think that

is a question. We cannot say we need this when we are not convinced we are able to do it. I am convinced we are able to do it from the experience in this field. People who have looked at the question of input have come up with the same answers, roughly, very compatible answers so that is a good point to me. There is a way to come out of this. It must not be the best way but we can look into it and surely there is an answer to it.

UHERKOVICH - I think it is not the question whether the standardization of the diagram is the only way or not, because each one is working, or a lot of people are working with these input diagrams; it would be very nice if they all used ours: writers are using the same language, the same tipe: this is standardization as I understand it.
Second, what I would mention is that we should tell who should do this work, who should make this standardization, because we can say: "ya , ya, this is very nice", but the real problem is who should do this. Another question: we hold an International colloquium here. Is there any tool by means of which we can transfer a program from one country to another one easily? This is a statement from Prof. Anderheggen too, who has written his program first for Switzerland. Is there any use for program transferring to another country, or is this a national standardization?

OBERNDORFER - The calculation of the reinforcement in the concrete here is quite different from the other countries, but the calculation of forces or deflections in a program should be done in a way that the program can be transferred from one country to the other one: however, the design of the concreting and of the steel members is a national problem, which it is not necessary to set up in a way that the program can be transferred.

FANELLI - Well, my comment to Prof. Takino' s paper is not really a question. I only wanted to give expression to a feeling of great admiration for the very advanced system that Prof. Takino presented to us. It seems to me that this is really an instance of how the possibilities of computers can be integrated in the fabric of an advanced society, an advanced industrial society, and how some of the questions that we are striving to answer can be answered in practice.
He in particular illustrated how the programs are reviewed by a specialized body, who in some way certifies them, and how the human appraisal is still put into the analysis in the end, when he speaks of the hand-written pages of comment at the end of the analysis. These are all important points and the solution that he illustrates is an example that these problems can be solved.
May be this is not a final solution, but it is a solution that has brought the work about, and this solution has taken into consideration, it seems to me, all the aspects that are important: the interaction with authorities, the interaction with codes of buildings, and so on. It shows the way, in a sense, to what we are looking for. So, I think that in this field, our Japanese colleagues are possibly one jump ahead of us and we could take all the possible profit from this experience to see how it was, what can be improved on it, if it can be transferred to another type of structure analysis besides building analysis and all things like that, and on all these aspects, I think we could have a very lively discussion.

KLEMENT - I would ask Prof. Takino how the telephone company does, as I think it is very interesting to hear that a telephone company gives computer calculations to so many engineers in the country. I have a question: are there other fields of computer calculations, may be commercial computer calculation, which this telephone company is selling ? Then, I want to know at what prices will the computer calculation be sold. Is there a monthly amount which is fixed to be paid for being with a terminal on the computer, are there other licenses for special programs which are used? I want to know if you are able to say whether this program system is a profitable one for the company.

TAKINO - Our corporation is a telephone company and has several kind of computer services. Many and many branches in Japan are subscribers to our system: exchange service and other offices of banks, medical agencies, distribution firms, etc. Our company is a half-governmental company , so we have not much money. We are going on well because we have many services, but in Japan, there are similar companies in competition: therefore, our service charge is not so high, it's a reasonable one. Subscribers pay a fixed charge, telephone and computing charge, however there is no difference if you use one or another program, there are no different licenses.

KLEMENT - Who pays these programs? And who pay their development and assistance?

TAKINO - Our company develops them and looks to the assistance of several software firms or manufacturers. The cost is included in computing cost.

HAAS - I have two questions to Prof. Takino of more commercial kind, about the reduction of manpower, which can be achieved when we use such structure analysis program. Here are my questions: how did you get these numbers ? Because if we have these numbers, you will promote the selling of those programs briefly. A second question: if we use those structural analysis programs, could we get, for example, a reduction of the reinforcement?

TAKINO - We investigated how a program may be used among subscribers by picking up 40 engineering services: we ask a question in each case and we distribute a form to fill in. Using your example, even if the solution is a real reduction, I suppose that normally they do not reduce the reinforcing bars, because first they choose on their experience and then they use our programs. Usually they do not repeat computer calculation to minimize the cost of the building.

TOMINO - We have run in the use of the system and also we have some contact with Mr Takino' s organization for the development of that program. If we were in a country (e.g. Pakistan) different from ours and we had to investigate a building from an antiseismic point of view following fixed rules, we could investigate it more and more in detail, using a well-chosen program. We can get some information and such a kind of calculation will give us some ideas and results. About Mr Takino' s mention of the advanced stage of the use of computers in Japan, I can' t be so optimistic as he is.

BOKELER - I think we must stop now. Thank you all for this very interesting discussion. We shall meet in the afternoon.

I SESSION

DISCUSSION

August 30, 1978. Afternoon.

Chairman: BOKELER (German Federal Republic)

FANELLI - I was willing to contribute an observation on the last remark made by Prof. Castellani about the possibility to extent his conclusions concerning two-dimensional structures. This is quite a detail point, and it is not much in the theme of our colloquium, but since the point has been raised, I think it worth while mentioning it. We made some non-linear analysis on cracked buttress dams, taking them as two-dimensional structures.
We studied cracks and made several analysis, some with through cracks and some with not-through and extended to the whole height of the dam, the apparent rigidity under hydrostatic load was almost unaffected in a two-dimensional situation, so valid deviations of the apparent stiffness begin to appear only with the completely through crack and cracks that extent at the whole height of the dam.
This could be an interesting indication, even in a two-dimensional situation, that agrees with your paper too.

BZYMEK - My comment on the paper of Mr Tagnfors is that the language presented in the paper is very clear and, as a matter of fact, not knowing the manual of this language, it is easy to understand this problem of oriented language. I think this is something similar to the idea presented by Prof. Anderheggen, because the language is divided in blocks, and this could suggest us to draw a syntax diagram from this program as well. As a matter of fact, about the way of presenting the interface between computing and design, I think that perhaps we would not be able to follow syntax diagrams, but at least we could recommend some criteria. One of these should be that the manual is presented very easy and, as far as I am concerned, I like very much the presentation of the manual based on the syntax diagram presented by Prof. Anderheggen.
At this point, I see some similarity between the paper of Mr Tagnfors and Prof. Anderheggen. This is the first comment of mine.
The second comment is on the paper delivered by Mr Oberndorfer from Austria. This was a paper which was very nicely presented, and , from my point of view, has some good conclusions, that means - it gave some criteria that should be met by good software, and it gave also some other points of concern, and I think this is very valuable.

LANG-LENDORFF - I want to come back to the paper presented this morning by Mr Takino. Mr Takino told us something about an evaluation committee working in his country. The task of this evaluation committee is as follows, I understood: to examine the different programs in order to be used in different calculations for housing and so on.
My question is: how does it work? Can you give us some features about working principles of this committee and how do you examine a program? You can't, in my opinion, divide the programs into good programs and bad programs. It all depends on what kind of examples you are calculating.

TAKINO - It is very difficult to examine the programs. The committee examines the programs in three or four meetings. At first, the committee examines manuals and the possibilities of their use by users; then they examine the manuals, and if there is some possibility that the user makes a mistake, the committee recommends to write things more simply in order to avoid misuse. Then, the committee proposes its data to inspect the correctness of programs. In Japan social contingencies have to be remitted to the building inspection officers; the building inspection officers have to understand and to examine the results of the programs. If these are not truly understood by the building officials, the committee recommends to quote the output.

BOKELER - If there isn't any other question, I would like to come back to the question if it is necessary to standardize the input documentation of programs and if we are able to specify the interface between computer and design structures engineering. If it is so, how shall we do the first step to do for this standardization?
The organization we have to go to, is it IABSE, is it EAB, is it FIP or the standard international organizations, or other institutes? In my opinion, all of us, must think about the answer.

FANELLI - I Would like to say a word of thanks to Dr Boekeler for his excellent chairmanship and now those who are interested in the visit to ISMES Laboratories can proceed to do it.