Problems of computational mechanics related to finite-element analysis of structural constructions

Anatoly V. Perelmuter and Sergiy Yu. Fialko *

Software company SCAD Soft
13, Chokolovsky bld., room 508
Kiev, 252680 GSP, Ukraine
e-mail: fialko@erriu.ukrtel.net

Abstract

The problems of computational mechanics, concerning with application of finite element analysis to structural constructions, are discussed. Our attention is addressed to medium-class software for personal computers with which structural constructions are usually analyzed. The complexity of a system and simplicity of its components, the large-scale of finite element problem, the heterogeneity of finite elements and its coupling, the estimation of correctness of finite element model, the problems of seismic analysis, the problem of indeterminacy and so on are the objective of this work.

Keywords: finite element analysis method, structural engineering, seismic analysis, fast solvers

1. Introduction

The contemporary market of industry-oriented software for structural strength analysis impresses very much by its versatility and widest functionality. There are real giants on this market such as ANSYS, ADINA, COSMOS, MSC NASTRAN and others, not restrained to any particular field of application but oriented at large-scale problems. A special place in this sphere is occupied by software intended for analyzing and designing structural constructions — SAP 2000, SCAD, GTSTRUDL, Robot-Millennium etc. These we will call medium-class software. Programs like these succeed in providing features especially appreciated by structural engineers — such as graphical preprocessors and postprocessors, catalogues of profiles, materials, regional climatic regulations. They include specific analysis options (construction of influence lines, seismic analysis etc.) Other special-purpose software can be mentioned, too, particularly programs oriented at narrow classes of problems or tutorial purposes.

Our attention will be addressed to medium-class software for personal computers with which structural constructions are usually analyzed. Problems related to the analysis of this kind have their peculiar flair that affects the structure and functionality of a computer program. There are certain requirements to analytic methods employed by the software, too.

Another important circumstance is that the software of this type is oriented commonly at the level of expertise possessed by a design engineer rather than a scholar researcher. Therefore the software should have an intuitive interface and highly automated functions. These programs should also account for specifics of the management of structural design activities. In particular, a typical form of organization used in this industry is a design team that includes a lead analyst who solves complicated problems of general nature and a few engineers who prepare data and solve series of more specific problems. The latter use simpler satellite programs interfaced with the main application administered by the analyst.

Our discussion is based on the experience of development of the SCAD Office [6] and Robot Millennium [16] software because the authors of this report are members of their development teams, participate in the support of the software, and are familiar with both the architecture and functionality of these programs.

2. Peculiarities of computational analysis in structural engineering

2.1. Complexity of a system and simplicity of its components

Objects of structural engineering are residential and public buildings, bridges, tanks, television towers, industrial buildings, and a great variety of other types of structures (Fig. 1). Civil buildings belong to most widely spread objects of construction.

All these objects, though much different, have common peculiarities in their design models:

- Bar elements are used extensively in structural models, unlike most objects of mechanical engineering. Even if the shape of a structure seems sophisticated, its load-bearing framework may consist of elements of relatively simple geometrical configurations. A characteristic example of this is one of most whimsical buildings, the Gugenheim Museum in Bilbao (Fig. 2). All the more so, this statement relates to most of the objects shown in Fig. 1. Design models of industrial, residential and public buildings, in their vast majority, consist of sets of rectilinear bars, plates, and flat shell elements. The latter have rectangular configurations, as a rule, or contain a number of rectangular sub-areas. Therefore the civil-engineering-oriented software systems deal very little with spatial finite elements which are often used to analyze FEM models of mechanical engineering objects (an example is a popular program SolidWorks [2].) These peculiarities of structural construction models beg for special software to be developed.
- High dimensionality of models required by complicated geometrical shapes of walls and floors, the use of automatic mesh generators and an object approach which treats a structure as a set of story, wall, floor etc. objects.
- A noticeable spreading of stiffness properties which causes an ill conditionality of the respective mathematical model.

Joints between elements of different types with different numbers of degrees of freedom in a node. Often enough, this circumstance brings the necessity to regularize the model's equation system, and this causes the ill conditionality again.

a)

b)

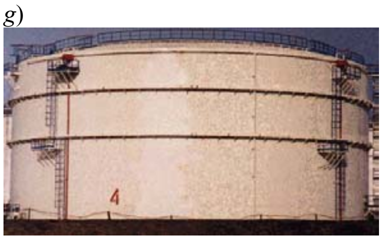
c)

d)

e)

f)

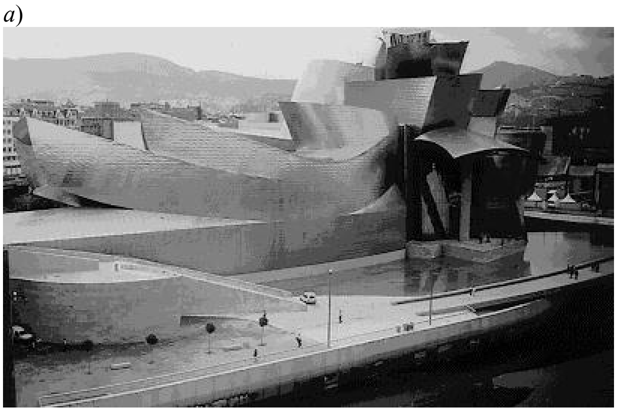
g)



h)

Figure 1: Examples of structural objects: (*a*) a residential house; (*b*) a bridge; (*c*) a skyscraper; (*d*) a television tower; (*e*) an industrial building; (f) an office building; (g) a tank; (h) an arc of a hangar

a)



b)

Figure 2: The Gugenheim museum: (*a*) its appearance; (*b*) its load-bearing framework

2.2. Model dimensionality

The structural analysis may involve models and schemes which are quite typical and by no means break records by containing 20 to 30 thousand nodes, 30 to 50 thousand elements of various types (bars, plates, shells, elastic links), and possessing over one hundred of stiffness property sets. 15 to 30 different loading patterns are usually under consideration, each one including hundreds of components of nodal or distributed loads. The dimensionality of the model grows drastically if one has to analyze load-bearing constructions of a structure jointly with its soil bed. An example of this kind is shown in Fig. 3,a where the model of a structure includes 27,138 equations while the "soil–structure" model consists of 319,133 equations.

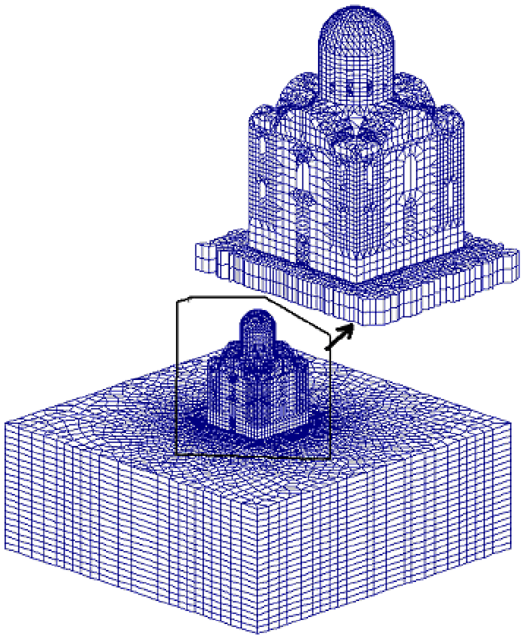


Figure 3: A design model of a structure together with its foundation

Usually, this class of problems can hardly be solved by direct methods because the structure of the adjacency graph with its root in a pseudo-peripheral node is not extended. In the example given above, we did not manage to factor the stiffness matrix using a PC Pentium III (CPU Intel 1000 MHz, RAM 512 MB) because the multi-front method with nested section ordering required 1292 MB of RAM to store the maximum front, while the skyline method (RCM ordering) demanded over 20 GB of disk storage. So, this problem was solved by iterative methods.

2.3. Heterogeneity of finite elements, problems with matching those

A plethora of complexities with the creation and verification of design models are related to a typical heterogeneity of finite elements often encountered in this class of computational analyses. It is only a rare case that the whole structure is represented by elements of the same type (such as plates). Most often, a single design model includes bars, plates and other finite elements at the same time.

It is a must for an advanced computational software system to allow nearly every possible combination of finite elements of most various types, dimensionalities, sizes and shapes, different stiffness properties. There are a lot of dangers here, sometimes revealed and sometimes concealed. The latter are especially hazardous.

A typical example can be an analysis of a spatial bar framework together with its slab foundation. This kind of a design model includes plate finite elements and bars attached rigidly to the slab. The axes of the column bars should cross the median surface of the slab in nodes of the finite element mesh on the slab. If no additional measures are taken, the design model described above will provide for a perfect match both between vertical displacements of the slab and the columns (perpendicular to the plane of the slab) and between respective slopes in nodes where the plate and bar elements join one another. Though, bending moments in sections of the columns near the slab calculated by this model have nothing in common with the true distribution of internal stresses.

To see this, imagine how the mesh is getting denser and the user expects the computational results to become more and more accurate. Though, starting from a certain scale of the mesh, further densification will be lessening the absolute values of the bending moments in the bars at points of their attachment to the slab.

In the limit, as the maximum size of the mesh cell tends to zero, these bending moments will tend to zero, too. This means that the design model in question provides a hinged rather than rigid connection between the framework elements and the slab. The fact that the user does obtain some formal nonzero bending moments with a particular finite element mesh of his choice evidences just an error of discretization and nothing more. But there is no reason at all to take the discretization error for an intended credible result!

In the design model presented above, the bars transfer concentrated bending moments to the slab. As is known, the solution of this elasticity problem has a logarithmic singularity in the slopes. Therefore the slope at the concentrated bending moment application point tends to infinity as the mesh becomes denser. Consequently, in order for the work of the concentrated moment at the slope to be finite, the bending moment itself must be zero. Thus, making the mesh denser will force the numerical solution to tend to the hinged column-to-slab attachment case.

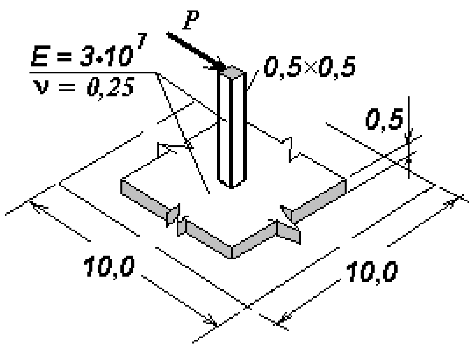
Results presented below make a confirmation of that. Let’s consider a square slab clamped along its sides and having a single column standing in the middle of the slab and fixed to it rigidly. The free top end of the column is subjected to an external concentrated force P directed along the global axis X (Fig. 4,a). The bending moment in the bottom section of the column will be a constant magnitude not depending on the size of the finite element mesh because the system is statically determinate with respect to the column.

The calculation of the displacement wn of the column's free end in the direction of the load using different finite element meshes (2n by 2n) shows that the deflection of the column grows almost linearly as n is increasing above n > 32. This results in an unlimited growth of the slope in the root section of the column as the finite element mesh becomes finer (see Fig. 4,b).

Of course, the fact is that it is the user who must be responsible for correctness of his design model from the viewpoint of mechanics and adequacy between the finite-element model and the real structure. Though, there have been made numerous attempts to prevent the said dangers by means of software implementation. In particular, flat shell elements resistant to drilling rotation have been introduced. This has been claimed to solve the problem with a torsion in an attached bar. Though, a detailed analysis shows that this way may lead to serious errors.

A more involved consideration of this problem is presented in the report [10] at this conference.

a)



b)

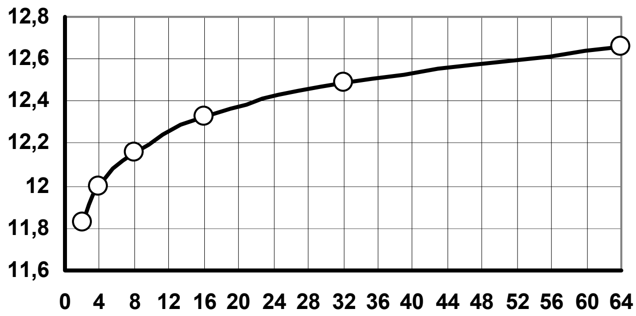


Figure4: A column fixed to a slab: (*a*) a schematic; (*b*) a displacement changing under the effect of the force

3. Estimation of correctness of input information

3.1. Trivial checks

It is known that the probability for an error in input data is much higher in large-scale problems. Engineering psychology researches state a power dependence of the human error probability on the volume of information processed by the man.

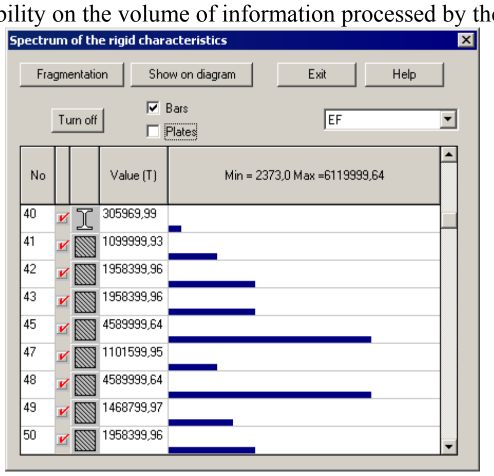


Figure 5: A spectrum of rigidities

Any contemporary computational analysis program operates fairly heterogeneous data which describe properties of finite elements, nodes, loads etc. The heterogeneity of the input information is of especial complexity for software systems oriented at performing structural design tasks. It is important for one to be able to detect and extract deviations from a common relationship, for example, analyze the spread of rigidity properties and represent those as a spectrum of rigidities (Fig. 5).

3.2. Validation of kinematical stability in the course of the matrix decomposition

Checks can be performed in the course of solving a problem, too. Errors in the model can be detected right during the solution, particularly such as a kinematical instability and the lost of positive definiteness of the matrix (in cases there must be one).

The presence of the kinematical instability is evidenced by a substantial reduction of the governing element of the matrix comparing to the respective diagonal element before the factoring. Though, the detection of a node's No. and the respective degree of freedom in the node based on this method is not always successful.

3.3. Detection of kinematical instability and other errors in a design model, visualization of kinematical mechanism schemes based on the natural oscillation analysis

To detect possible mechanism-type motions in a structure, the SCAD software implements a special analysis mode based on a Lanczos block method and involving spectral transformations. The idea of this option is to use a shift technique which enables one to analyze unconstrained systems and even mechanisms. In the latter case the system's stiffness matrix $\mathbf{K}$ is singular. Though, the $\mathbf{K}_\sigma = \mathbf{K} - \sigma\mathbf{B}$ matrix is not singular provided that the shift σ is chosen appropriately, and it can be factored. In particular, the matrix $\mathbf{B}$ can be assumed equal to the mass matrix $\mathbf{M}$. Then, the presence of zero natural frequencies in the model $\mathbf{K}_\sigma\phi - \varpi^2\mathbf{M}\phi = 0$ evidences the kinematical instability, and natural modes that conform to zero frequencies describe possible movements of the mechanism strictly. included in their original form and will be reproduced in black and white.

This approach enables us both to determine whether a particular system is kinematically unstable and to visualize modes of mechanism-type motion, thus giving the user a hint where to install additional constraints in order to eliminate the instability. Fig.6 shows a fragment of a structure that contains an unconstrained solid body, and one of its possible mechanism-type modes of motion.

A more detailed presentation of the same material can be found in the report [13].

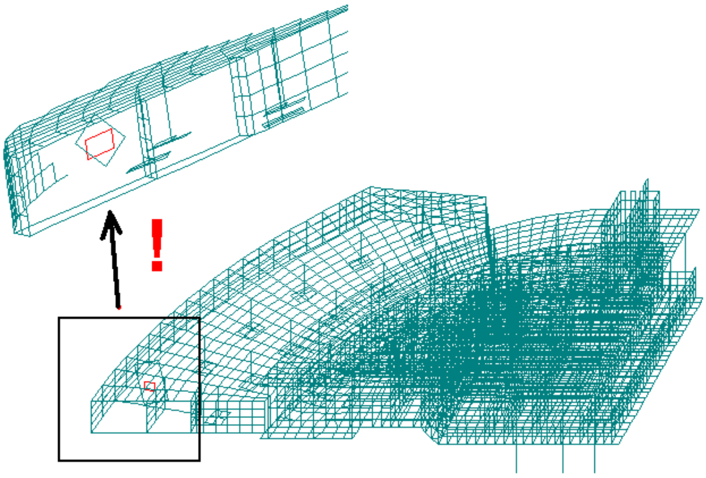


Figure 6: Detecting a rigid-body mode of motion

4. A specific type of analysis — seismic analysis

4.1. Sum of modal masses and local modes

It is known that the sum of modal masses is a criterion whether the number of natural modes taken into account is sufficient. Current seismic regulations require that the sum of modal masses along each direction be at least 90%. In many cases this requirement can hardly be met. It often appears that the lower part of the spectrum includes local oscillation modes that contribute only a little to the response of the whole system as it moves during the seismic event. Some kinds of structures do not even contain oscillation modes which contribute much to the seismic movement of the system. One finds that in those structures small pieces of its seismic response are distributed over a large number of natural modes. In this connection, there arises an enormous computational problem — how to determine 100 to 500 or even more natural frequencies and oscillation modes.

4.2. Development of specific algorithms — Ritz vectors, a residual mode, a seismic mode etc.

To escape from this situation, the following measures are usually taken. E. Wilson suggests in his papers that decompositions by natural modes should be replaced by decompositions by specially constructed Ritz vectors, more informative ones for the purpose of seismic analysis [14]. This method is implemented in the SAP2000 software. A similar technique has been used in the Robot Millennium software [15]. Another trick is to use a residual mode (a pseudo mode).

Apparently, it should be admitted that the problem has no easy solution for today. The matter is that the solution of a dynamics problem in terms of Ritz vectors is strict only if motion equations are integrated directly. At the same time, most dynamical problems in the structural engineering practice are solved by the spectral technique. The latter has some immanent contradictions when using a basis different from the set of natural modes, so one may obtain noticeable discrepancies between the calculated displacements/stresses and those observed in reality.

4.3. Problem of data amount, filtering

In many cases one can reduce the computational effort essentially by using only some chosen natural modes that contribute much to the seismic response of the system. This kind of filtering is based on an analysis of modal mass values for each natural mode. This approach is implemented in the Robot Millennium software [15, 16].

5. Requirements to the software's response speed

5.1. Sources of requirements

Actually, the reduction of computation time has ceased to be a critical issue in common structural design activities. The usual relation of the effort and time is such that the most part of business time (at least 80%) is spend for preparations and verifications of input data, and then for reviewing and analyzing results. Under these circumstances one may think that trying to reduce the governing equation system solution time from twenty minutes to two makes no real sense. Developers often express this point of view, but we believe it is not true.

We deem it useful to indicate a number of practically important problems in which the speed of solution of linear/linearized governing equation systems is a really critical point:

- the design in an interactive dialog mode seems impossible if the software responds too slow; a long and painful waiting for an answer makes the interactive design procedure almost fruitless;
- in spite of the authors' steady belief that an extensively detailed design model is just a serious mistake of an analyst, large-scale problems do arise in real design

practice — for example, as long as a complex "structure–soil" system is under consideration where three-dimensional finite elements are used to simulate the soil bed's behavior;
- the solution of nonlinear problems involves multiple solutions of linearized systems of governing equations at each step of a step-by-step procedure or at each iteration;
- optimization problems and related multi-variant searches also lead to the need for multiple solutions;
- problems with indeterminate parameters are posed and solved more and more often recently; for these, one of most universal techniques is an imitative modeling (including the dynamical behavior modeling) which requires the same multiple repetition of linear solution.

These and other related problem formulations can be feasible only if the software used to perform the task is provided with efficient algorithms for solving systems of algebraic equations and eigenvalue calculation.

5.2. Construction of quick solvers

The quick solvers in the FEM analysis available for today include direct methods for sparse matrices and highly efficient iterative methods.

The advantage of direct methods is their low sensitivity to ill-conditioned matrices and to the number of right parts (if there are not too many), and the possibility to detect the kinematical instability of the design model. The efficient direct methods are based on a reduction of filling. The filling means nonzero elements of the factored matrix standing in positions where zeros used to be in the original matrix. The less filling is reached, the higher the efficiency of a direct method. Therefore the most responsible step for a direct method is the matrix reordering. Until recently, the most popular reordering algorithm was an inverse Cuthill – McKie method employed to decrease the profile width [17]. In recent years the most widely spread methods have become the minimum degree algorithm, the nested section method, and the multi-section method based on domain decomposition [18]. Commercial FEM programs implement direct methods for sparse systems, most often, on the basis of a multi-front approach [18-22].

Iterative solvers are preferable for large-scale problems in which the number of equations can be 100,000 to 700,000 or more. Their drawbacks include a slower convergence in the case of ill conditionality and a high sensitivity of the computation time to the big number of right parts. The effective technique to suppress the ill conditionality effect is a preconditioning. Suppose a linear static problem $\mathbf{Kx} = \mathbf{b}$ needs to be solved. The preconditioning consists of a transition to another problem $\mathbf{B}^{-1}\mathbf{Kx} = \mathbf{B}^{-1}\mathbf{b}$ where $\mathbf{B}$ is a preconditioning operator. If $\mathbf{B}$ is positive definite, the system of equations $\mathbf{Bz}_k = \mathbf{r}_k$, where $\mathbf{r}_k = \mathbf{b} - \mathbf{Kx}_k$ is a residual vector and k is No. of iteration, can be solved much faster than the original system, and its conditionality number $C(\mathbf{B}^{-1}\mathbf{K}) < C(\mathbf{K})$, then the preconditioned problem will have a faster convergence than the original one. In a limit case, when $\mathbf{B} = \mathbf{K}$, the preconditioned problem converges to its exact solution in one iteration. So, the trick of quick iterative methods is how to construct such preconditioning which would not require much time and resources and at the same time provide that $C(\mathbf{B}^{-1}\mathbf{K}) \to 1$.

The theory of iterative methods states that lower modes are the slowest in convergence. The worse the conditionality of a problem, the slower their convergence [23]. Hence the idea of multilevel methods [29, 30]. The key point is to construct a rough-level model intended for predicting low-mode components of the solution. The convergence in high-mode components is ensured by smoothing. The maximum effect is usually achieved by combining the preconditioned conjugate gradient method and the multilevel method's idea. This builds up a family of conjugate gradient methods with multilevel preconditioning.

Commercial FEM programs employ most often the conjugate gradient method with preconditioning of an incomplete Choletsky factorization type, multi-mesh methods, algebraic multi-level techniques [24, 25], and aggregative multi-level method [26], methods of space decomposition and subspace correction [27]. A review of iterative methods is presented in [28].

Talking about the eigenvalue problem, we should note that currently most popular methods based on the stiffness matrix factorization include the block subspace iteration method [14] and the Lanczos block method [30, 31]. The implementation of the Lanczos block method with shifts in the SCAD software will be discussed in [13].

In cases when we cannot reorder the stiffness matrix efficiently, it is reasonable to use methods not requiring the matrix to be factored. The most efficient technique is the conjugate gradient method with preconditioning [28]. Though, the traditional algorithm of this method suffers from a convergence suffocation in some cases [33]. This is overcome by introducing shifts into the preconditioning [34, 35]. A more detailed discussion of this problem will be given in the report [36]. The report [37] presents a quick approximate method for determining natural oscillation frequencies and modes. That one is a Ritz method which makes use of a gradient procedure with aggregate multilevel preconditioning to construct an orthogonal system of basis vectors.

A modern FEM analysis software must implement both quick direct methods and iterative ones because nobody can say beforehand what method is going to be most efficient in a particular case. For example, prominent programs like MSC NASTRAN, ANSYS, ADINA include both direct sparse matrix solvers and efficient iterative solver tools at their disposal. Among civil-engineering-oriented software, we should notice Robot Millennium which also implements both direct sparse solvers and an aggregate multilevel iterative solver [15]. The SCAD software that implements a multi-front solver is worth mentioning, too.

6. Choosing a most disadvantageous combination of loads

The peculiarity of construction objects is that one has to deal with a plethora of variations of loads applied to a structure. This is an essential distinction from "common" engineering where this problem is less sharp.

The fact is that even simplest buildings have tens of rooms, and in each of those the useful load can be present or absent in a particular moment. It is by no means obvious that the critical case will be the fully loaded structure. Moreover, we can say for sure that it is not the case for a good deal of structures. Then add the necessity to account for a few possible directions of wind or seismic loads, and numerous possible positions of movable loads such as those caused by bridge cranes. All this makes it quite clear how much effort the problem of choosing a design load combination may take. One should note also that a direct exhaustion of possible variations is difficult even when there are only twenty or thirty independently acting loads.

In essence, one needs to solve some optimization problem where one has to find an extremum of the structure's response in the set of possible loaded states of the system. This set may be of a pretty complex build because some of the applied loads can be related to one another via logical relationships of the following types:

incompatible — some loads cannot act together for purely physical reasons, for example, a south wind cannot be accompanied by a north wind, nor snow can be combined with the maximum summer heat;

bound — certain loads can be treated only as acting together; this is often the case when different loads are of the same physical origin and are presented separately only for convenience;

accompanying — one of loads cannot exist without the presence of another, while the other way round is quite feasible: for example, the bridge crane's braking force cannot exist without the pressure of the crane's wheel, while the pressure can exist without braking;

limited — some of jointly acting loads cannot exceed an established limit in total, for example, loads from bridge cranes are limited to two cranes on a single pathway or in the same section.

One of feasible approaches to the solution of the problem is to represent the logic of interaction between different loads as an oriented graph [9]. Then the problem can be formulated as a known problem of searching a network for the biggest flow [8].

Let's give an example of such graph for the situation when the following elementary loads can be applied:

- 1 — dead weight;
- 2 — snow;
- 3 — wind from the left;
- 4 — wind from the right;
- 5 — maximum pressure of the crane onto the left column;
- 6 — maximum pressure of the crane onto the right column;
- 7 — braking of the cranes to the left transferred to the left column;
- 8 — braking of the cranes to the right transferred to the left column;
- 9 — braking of the cranes to the left transferred to the right column;
- 10 — braking of the cranes to the right transferred to the right column.

Fig. 7 shows a schematic of the respective graph where there are arcs 1–10 conforming to the elementary loads listed above and four more arcs (dash lines in this figure) conforming to zero values of the load intensity. These additional arcs enable one to bypass those loads in the graph which must not necessarily be included in a design load combination (that is, which unload the structure).

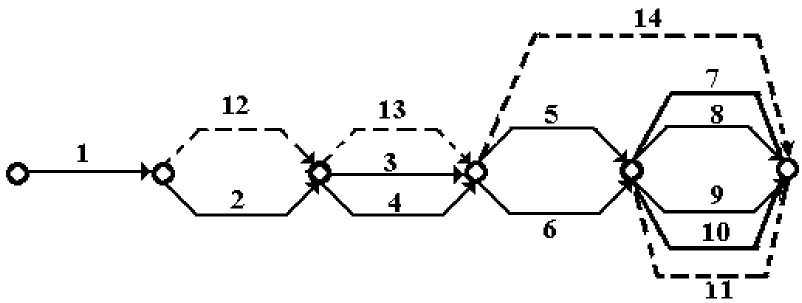


Figure 7: A graph of the logical structure of loads

7. Problem of indeterminacy

Modern buildings or other structures are, most often, complex structural multi-element aggregates created to perform a plenty of different functions. During their lives the structures go through a long sequence of various working states. The specifics of structural engineering is such that its final product (a building, a structure) must combine three features often contradicting one another: functionality, aesthetics, designability.

An idealization of the design model and impossibility to make it a perfect reflection of the real structure create a situation of some indeterminacy. It is these conditions of indeterminacy under which design decisions have to be made.

The indeterminacy is caused by either unavailability of required information (for example, we are not able to know all future regimes of the structure's operation) or its incompleteness (we can hardly imagine knowing mechanical constants precisely in any point of the structure). The unavailability of some types of information and its incompleteness are key points — they cannot be overcome altogether, and deeply as we could study the problem of our interest, we may never say we have taken absolutely all into account in our model.

Though, it is not only the unavailability and incompleteness of data that causes the indeterminacy to appear. There is also an ambiguity of the data, that is, a possibility to interpret the same factors differently. This circumstance requires us to estimate all possible alternatives. There are known classical approaches to the indeterminacy which can be classified into the following decision-making methods:

- making use of the probability theory, the decision being based on the objective earlier experience;
- making use of expert estimations, the decision being based on the subjective experience of an expert (or a panel of experts);
- a minimax estimation, when the best of achievable solutions is adopted with the assumption of the worst possible course of events, i.e. the decision is made by the possible result.

All these options can be used together or separately. They are intended to estimate the credibility of a design model. There are other factors, too, which determine how approximate the design model is and what errors, distortions, contradictions may appear in it.

First, there are design modeling errors (approximation errors) that appear due to either our knowledge’s approximate nature itself or an intentional rough approximation of it. These “errors” include using simplified mathematical representations such as polynomials of low degrees for describing displacement fields in the finite element analysis, truncation of series in the Galyorkin method etc. The same category includes errors caused by discrepancies between scientific theories and assumptions that are used to simulate different parts of the same design mode. A typical example is a discrepancy between concentrated forces as popular models of loads, on one hand, and plate finite elements, on the other hand. The latter cannot balance the concentrated actions by finite values of their shear forces. It is natural that totally mythical values of the shear force in elements obtained by such calculation result directly from the said discrepancy between the models.

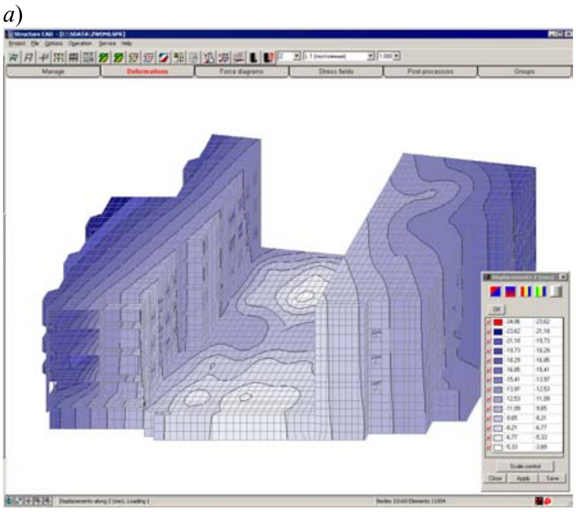
Second, we should note the approximate character of nearly all specifiable properties of a model. This is related to tolerances for sizes, weights and other measurable magnitudes existing in practice. From the practical viewpoint, both inaccuracies stated above differ little. Though, in the first case we deal with a limited accuracy of our simulation (either intentional or unconscious) while in the second case it is the original object’s properties which cause the limited accuracy.

8. Postprocessing

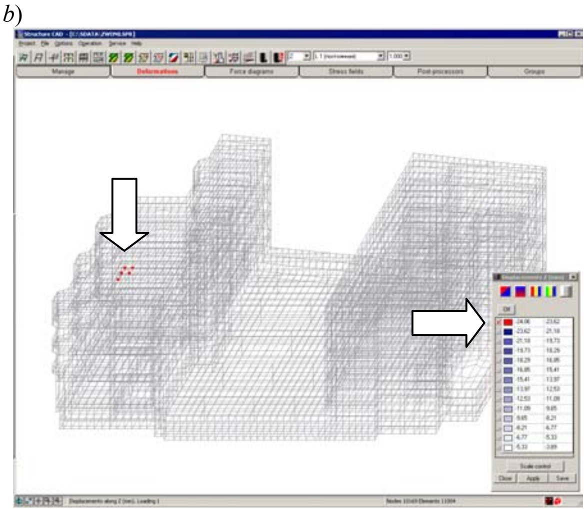
8.1. Problem with understanding

Results of static or dynamical analysis of a complex system represented as numbers contain vast arrays of data perceiving and reviewing which is practically unfeasible. A selective result printout option available in most programs is of little help, too, because the analyst does not necessarily know which of the values he expects to be critical.

a)



b)



c)

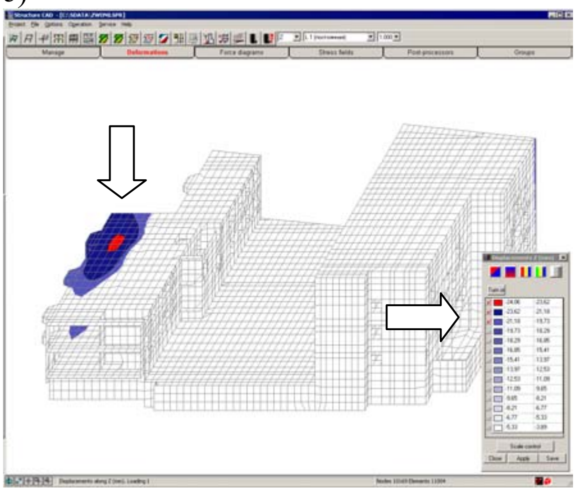


Figure 8: An example of localized ultimate values of a factor of interest: (*a*) isofields in the whole model; (*b*) displaying only maximum values of the factor in the ”transparent” model using color markers; (*c*) isofields of maximum values limited to a specific range.

A much better demonstrativeness can be achieved by using graphical representations of results as curves, color maps, isofields. These methods compress the data to a great extent, and the information thus becomes more or less apparent.

Though, even this technique is not always enough to make a proper analysis, because the graphical information can be still hardly accessible for the system as a whole (Fig. 8,a). A fragmentation of it will restore the demonstrativeness but cause another problem — how to find a particular fragment at which specific results of the user's interest have been obtained [12]. Solution of this problem is not trivial at all for a complex model consisting of tens or hundreds thousands of nodes and elements. For example, Fig. 8,a shows a design model with isofields of vertical displacements drawn on it. Note that this figure does not show an area of maximum values.

The way out of the situation can be a technique suggested in the SCAD software. It is based on a control of color indication and described in detail in the report [11] at this conference. The point of it is that one uses the color map to find the factor in question in the "transparent" model, first, and then detects the location where the needed values appear (Fig. 8,b). Next, the color indication is used only for a part of the isofield that belongs to the range of interest, and all the other levels are turned off (Fig. 8, c). In this way "critical" results of the analysis are localized.

The general depiction given by the graphical representation of the analysis results accords best with the well-known statement that the goal of a calculation is an understanding rather than a raw number. Having analyzed the general picture, one should always turn to numerical results that now can be selected from the common data flow consciously.

When dealing with problems of buckling/stability, one should be aware of a universal tool for visualizing the stress and strain distribution in a system. This tool is a picture of the deformation energy field. If the energy distribution has been constructed with taking the geometrical stiffness matrix into account, in this way one acquires the capability of classifying particular fragments of the system (down to its separate elements) into one of two following categories: either restraining or pushing elements (parts) of the system [2]. The restraining elements facilitate the stable equilibrium of the system, while the pushing elements play a negative role because they force (push) the mechanical system to buckle.

The role played by a particular subsystem is checked by calculating the energy accumulated by this part as it deforms by a buckling mode. For the system as a whole this energy is zero. Parts where it is non-positive are pushing ones, while those with a positive deformation energy can be classified as restraining subsystems.

Based on numerical values of the energy, pushing elements of the system can be ordered by the degree of their "blame" for the critical state of the system. A contribution of each of the system's elements to its total energy balance can serve as a convenient quantitative measure of its responsibility for the equilibrium stability.

8.2. Meeting requirements of design codes

In the course of the structural design procedure, results of static and dynamic analyses of constructions are used to estimate their strength and stability. This process is regulated by design codes. Unfortunately, design codes are by no means as strict and non-contradictory as computer mechanics methods. These documents were initially created when manual calculations reigned supreme, they absorbed informal practical experience, and they are based on a great deal of compromises.

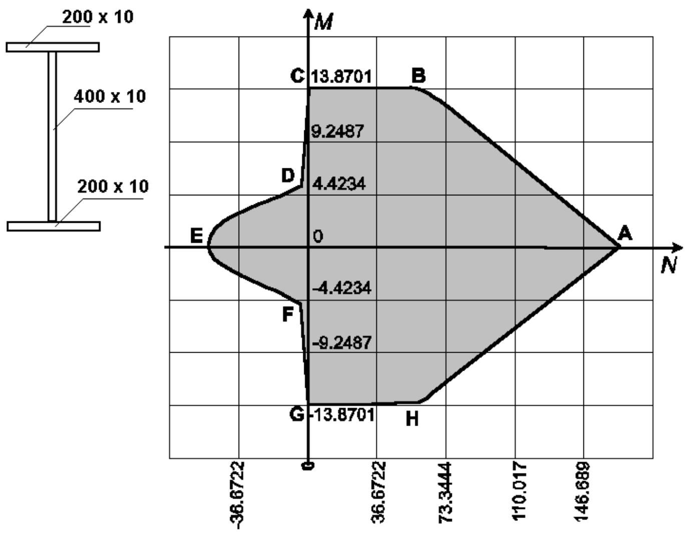


Fig.9. A load-bearing ability area

The design regulations contain numerous empirical relationships, correction coefficients, and "simplifying assumption" which very often can be "complicating assumptions" on the way of computer analysis. All these together create certain problems for software developers, namely:

- how to match the level of accuracy of the analysis with the accuracy of formulations found in the codes, because a paradoxical situation often arises — a system calculated with a higher accuracy is inferior to a system calculated approximately when compared in compliance with the code requirements;

- results of static and dynamic analyses need to be made rougher and interpreted specifically to comply with terms of the design codes;
- concepts operated by the design codes need to be introduced into finite-element analysis software. For example, a beam or a column comprising certain groups of finite elements must possess certain properties;
- analysis results need to be specially processed to obtain some properties of a structural object that cannot be described in FEM terms, such as the tilt of a building or the axis of elastic centers.

It is not only software developers who face the said problems. Computer mechanics experts have to deal with them, too, because the problems require specific methods to be developed for their solution. We are presenting only one example here for the purpose of illustration — the issue of a potential non-convexity of the load-bearing ability area of a structure's elements, in the case this ability must comply with all design regulations (strength, stability, rigidity). Let's consider a compressed and bent element of a steel structure and determine its load-bearing ability area. This area is shown in Fig. 9 for a steel bar with its design strength Ry = 2050 kg/cm2 and its effective length 600 cm in both principal planes.

The boundary of the load-bearing ability at the segments AB and AH is defined by the condition of sufficient strength under the combined effect of the tension and bending, at the BC and GH segments it is defined by the stability of the plane bending, and at the CD and GF segments (as well as at DEF) by the stability out of the moment's plane.

By itself, the non-convexity of the area in question may lead to quite a few unpleasant effects. The most apparent of the effects is related to the fact that traditional disadvantageous stress combinations estimated by engineers either do not include some actions or include them incompletely. In a non-convex area, though, it is quite possible that the disadvantageous combination occurs at some intermediate point. For example, if one variation of loads conform to the C point while another to the E point (in both cases the load-bearing ability is ensured), then we can take halves of the limit moment and force and find ourselves in the K point beyond the admissible area.

There arises a problem of this type — find conditions under which the convex surface of points depicting all possible stressed states belongs to the load-bearing ability area defined by structural design code requirements. As far as we know, there has been found no satisfactory general solution of this problem fitting for a practical software implementation.

9. Conclusion

The development of finite element software for structural design creates lot of complex specific problems of computational mechanics.

The experience of development and practical usage of software for structural design requires the permanent replenishment and analysis. The corresponding efforts should be supporting by scientific community.

References

[1] Vorovich, I.I., Lebedev, L.P. Some questions of continuum mechanics and mathematical problems in the theory of thin-wall structures, *International Applied Mechanics*, Vol. 38, No 4, 2002.— pp. 2-21.

[2] Avedian, A.B., Danilin, A.N. Strength for strength dummies, *CAD and graphics*,. No1, pp. 75-83; No2, pp. 63-68; No3, pp. 39-46, 2000 (In Russian).

[3] Oden, J.T., Belytschko, T., Babuska, I., Hudhes, J.R. Research Directions in Computational Mechanics, *Computer Methods in Applied Mechanics*, Vol. 192, No. 7-8, pp. 913-922, 2001

[4] Perelmuter, A.V., Slivker, V.I. *Numerical Structural Analysis: Models, Methods and Pitfalls,* Springer Verlag, Berlin-Heidelberg-New York-Hong Kong-London-Milan-Paris-Tokyo, 2003.

[5] Vasiliev, V.V. To the discussion on the classical plate theory, *Mechanics of solids*, N4, pp. 140-149, 1995 (In Russian).

[6] Karpilovsky, V.S., Kriksunov, E.Z., Perelmuter, A.V. et al. *SCAD for users*. —Kompas Publishers, Kiev, 2000 (In Russian).

[7] Grigolyuk, E.I., Shalashilin, V.I. *Problems of Nonlinear Deformation*. Dordrecht et al.; Kluwer, 1991.

[8] Ford, L.R., Fulkerson, D.R. *Flows in Networks,* Princeton University Press, Princeton, NJ, 1962

[9] Artemenko, V.V., Gordeyev, V.N. A program for calculating rated stress combinations in conditions of a complicated logical relationship between loads, *Computational and mechanization facilities in structural design,* No2, pp. 10–14, 1967 (In Russian).

[10] Perelmuter, A.V., Slivker, V.I. Problems in matching finite elements having different dimensionalities, *Proceedings CMM 2003*

[11] Kryksuniv, E.Z., Perelmuter, A.V. Slivker, V.I. Techniques to check properties of complex design models, *Proceedings CMM 2003*

[12] Elizarov, S.V., Benin, A.V., Tananaiko, O.D. *Modern methods for analysis of engineering structures used in railway transport. Finite-element method and the COSMOS/M software,* PGUPS, St.-Petersburg 2002. (In Russian).

[13] Fialko, S.Yu, Kriksunov, E.Z. and Karpilovsky, V.S. A block Lanczos method with spectral transformations for natural vibrations and seismic analysis of large structures in SCAD software, *Proceedings CMM 2003*

[14] Wilson, E.L., *Three dimensional dynamic analysis of structures*, Computers and Structures, Inc., Berkeley, California, USA, 1996.

15. Fialko, S.Yu., High-performance iterative and sparse direct solvers in Robot software for static and dynamic analysis of large-scale structures, *Proceedings of the second European conference on computational mechanics*, *Poland, June 26-29, 2001*, 18 p.

[16] Robot Millennium v. 15.0, *User's Manual*. pp. 244-278.

[17] George, A., Liu, J., *Computer solution of large sparse positive definite systems*, Prentice-Hall, Inc., 1981.

[18] Ashcraft, C. and Liu, J.. Robust ordering of sparse matrices using multisection. *Technical Report CS - 96-01 Dept. of Computer Science, York University. February 1996 to appear in SIMAX*

[19] Duff, I.S., Reid, J.K., The multifrontal solution of indefinite sparse symmetric linear equations, *ACM Trans. Math. Software*, vol. 9, pp. 302-325, 1973.

[20] Gend, P., Oden, J.T., R.A van der Geijn., A parallel multifrontal algorithm and its implementation, *Comput. Methods Appl. Mech. Engrg*., vol. 149, pp. 289-301, 1997.

[21] Fialko, S.Y. Method of nested substructures for analyzing large-scale finite-element systems, applied to calculation of

thin shells with high ribs. *Prikladnaya Mekhanika = Applied Mechanics*, vol. 39, No 3, p. 88–96. (In Russian}.

[22] Fialko, S.Y. A multi-frontal method for solving large-scale finite-element problems applied to calculation of thin shells with massive ribs. *Prikladnaya Mekhanika = Applied Mechanics*, vol. 39, No4. (In Russian).

[23] Bakhvalov, S.N., Zhidkov, N.P., Kobelkov, G.M., *Numerical methods* , Nauka, Moscow 1987 (In Russian).

[24] Axelsson, O., Vassilevski, P. Algebraic multilevel preconditioning methods, I, *Num.Math*., 1989, vol.56, pp.157-177.

[25] Axelsson, O., Vassilevski, P. Algebraic multilevel preconditioning methods, II, *Num.Math*., 1990, vol.57, pp.1569-1590.

[26] Fialko, S. Yu. High-performance aggregation element-by-element iterative solver for large-scale complex shell structure problems. *Archives of Civil Engineering*, XLV, 2, 1999. pp.193-207

[27] Xu J., Iterative methods of space decomposition and subspace correction, *SIAM Review*, vol.34: No 4, pp.581-613, 1992.

[28] Papadrakakis, M, *Solving large–scale problems in mechanics*, John Wiley & Sons Ltd., 1993.

[29] Brandt, A., Multi-level adaptive solutions to boundary-value problems, *Mathematics of Computations*, vol.31, No 138, pp. 333-390, 1977.

[30] Hackbush, W., Trottenberg, U., *Multigrid Methods*, Springer-Verlag, Berlin, 1992.

[31] Ericsson, T. Ruhe, A., The spectral transformation Lanczos method for the numerical solution of large sparse generalize symmetric eigenvalue problem, *Math. Comput*., vol.35, pp. 1251-1268, 1980.

[32] Grimes, R.G. Lewis, J.G., Simon, H.D., A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems, *SIAM J. Matrix Anal. Appl*, vol.15, No1: pp. 1-45, 1994.

[33] Young Cho, Yook-Kong Yong, A multi-mesh, preconditioned conjugate gradient solver for eigenvalue problems in finite element models, *Computers & Structures*, vol. 58, No3, pp. 575-583, 1996.

[34] Feng, Y.T., An integrated Davidson and multigrid solution approach for very large scale symmetric eigenvalue problems, *Comput. Meths. Appl. Mech. Eng*., vol.190, pp. 3543-3563, 1999.

[35] Fialko, S. Aggregation Multilevel Iterative Solver for Analysis of Large-Scale Finite Element Problems of Structural Mechanics: Linear Statics and Natural Vibrations. *LNCS* vol. 2328, p. 663 ff, http://link.springer.de/link/service/series/0558/tocs/t2328.htm

[36] Fialko, S. Yu. An aggregation multilevel iterative solver with shift acceleration for eigenvalue analysis of large-scale structures, *Proceedings CMM 2003*

[37] Fialko, S. Yu. High-performance aggregation element-by-element Ritz-gradient method for structure dynamic response analysis. *CAMES*, vol.7, pp. 537-550, 2000