2013 Questions of the Month

****

The January Question of the Month

****

Suppose we want to analyze the fluid flow around an airfoil (or another solid object). For simplicity, let's assume that the problem is linear; for example we are solving the potential irrotational flow equations or the linearized Euler equations. In the physical problem the domain is the entire space outside the airfoil, which is practically an infinite domain. If we want to use a standard method based on finite differences or finite volumes or finite elements, we must have a finite computational domain. So we must "truncate" the infinite domain somewhere, by introducing a "computational box" enclosing the airfoil. The question is: how large should this computational box be? I do not expect an accurate, quantitative answer, but a (short) qualitative discussion on this issue.

****

Answer

****

Let's assume first that the problem is a steady-state problem. The original domain is infinite, which means that the flows comes in (say) from x = minus infinity and goes out toward x = plus infinity. Thus, at -/+ infinity the flow is "undisturbed". The airfoil creates a disturbance in the flow. In theory, this disturbance extends to the entire space around the airfoil, but of course the intensity of the disturbance decays fast as one goes away from the airfoil. The computational box should be big enough so that on its external boundaries the flow disturbance caused by the airfoil in negligible and does not affect the accuracy of the results. On the external boundaries one then imposes the "non-disturbed flow" boundary conditions.

The above is the qualitative view. But what is the quantitative answer? How big should the box be so that the disturbance caused by the airfoil be negligible on the external boundaries? This of course depends on the level of accuracy sought, and it is not easy to give a rule that quantifies this in general. Some old papers stated that for "engineering accuracy" one should take the box to be about 5 airfoil-cords in size. A possible option is to determine this size in an adaptive manner. As AH proposes, one may "examine gradients of the solution within the finite domain till they become negligible".

In any case, the computational box determined in this way would be quite large. A large domain causes difficulties because there is a need to cover it with a grid/mesh, resulting in many degrees of freedom and a large computational effort. However, one may devise discretization methods to solve the problem efficiently in this large domain. One idea was proposed by OL: "One should decrease the resolution away from the airfoil. This can naturally be implemented with a multigrid-FAS composite grid. The decay rate of the resolution can probably be derived from a variational principle applied to the system in question, or from work-error exchange optimality criteria."

If the problem is time-dependent, describing a "signal" that starts to propagate from time t=0, then in principle one can construct the box as indicated by ET: "The domain can be chosen large

enough so that no information reaches the outer artificial surface and reflects back into the domain of interest within the time of integration being considered." But in practice, such a domain would often be very large and render the computation inefficient. So here too, one may be content with a smaller domain, but still large enough to generate only a small error.
But as OABY, AH, ES and ET remarked, there is a different approach that is regarded as much more prudent. This approach is based on taking a _small_ computational domain - typically just the minimal area of interest - by truncating the unbounded domain via an artificial boundary B, and on using an appropriate boundary condition on B. Sophisticated "non-reflecting boundary conditions" or "absorbing boundary conditions" (and also absorbing layers such as PML, as ES remarked) have been devised by various people, most notably Eli Turkel who has been one of the pioneers in this area. The problem in the small domain can then be discretized by Finite Elements or Finite Differences or whatever other discretization method one wishes to use. Research in this area is still ongoing quite actively today.
Correct answers were received from:

Orna Agmon Ben-Yehuda, Amiel Herszage, Oren Livne, Evgeny Shavelzon, Eli Turkel, Zvi Zaphir.

****
The February Question of the Month
****

The Question this time if for the more solid people among us. Consider the problem of linear elastic plate bending. There are two well-known theories of plate bending. The more basic theory is the Kirchhoff theory. (I have heard someone claiming that the only tricky thing in this theory is the spelling of the name Kirchhoff.) It neglects effects of shear deformation and therefore is suitable for thin plates. The more sophisticated theory is the Reissner-Mindlin (RM) plate theory. It does include the effect of shear deformation (by letting the normal displacement and the rotations be independent unknowns), and is suitable for thick (and thin) plates. Now, we could expect that in commercial Finite Element (FE) software one would be able to choose to use either a Kirchhoff-based element or a RM-based element. But it is a fact that all (or almost all) commercial FE codes include only RM-based elements in their menus. What do you think is the reason for this?

****
Answer
****

First of all, it seems that my statement that "all (or almost all) commercial FE codes include only RM-based elements in their menus" is not really correct. DB and RBZ remark that several commercial FE codes (e.g., NASTRAN and MARC) include a variety of shell elements, that can also be used as plate bending elements (if they are flat), and some of them are based on the Kirchhoff-Love theory. I will write more about this later.

Still, the Kirchhoff-plate elements are problematic, and the reasons can be categorized into physical reasons and numerical reasons. The physical reasons were outlined nicely by RH and ZZ (and also pointed to by RBZ). Let me quote RH on this:

"I can speculate on several factors for preferring to work with RM elements:

1. Even thin plates need shear deformation when the shear modulus is small, as is usually the case for composite materials.

2. Furthermore, interlaminar shear, that is important for failure of composites is poorly calculated even for fairly thin plates.

3. The thinness of the plate is relative to the wave length of the deformation. So when vibration modes and buckling modes are calculated, the plate may have a large length to thickness ratio, but the wavelength to thickness ratio may still be small for higher frequency modes and for many buckling modes.

4. Finally, since many people who use this FE packages do not know much plate theory, it is safer not to give them the chance of choosing wrong."

In fact, one may wonder: if we have two theories, one being more general from the other (because it includes shear deformation effects), why should we _ever_ use the less general theory? The main answer is that for problems where shear deformation is really negligible (e.g., very thin homogeneous plates), the standard RM finite element formulation runs into trouble; it gives rise to so-called "shear locking", which is a serious numerical difficulty that ruins the FE accuracy. There are special techniques to overcome shear locking, and indeed they are used in commercial codes. But this is already a complication, that may suggest that maybe it is best to use the Kirchhoff theory when shear deformation is negligible.

The numerical reason that the use of the Kirchhoff theory in FE analysis is problematic (which is what I had in mind when asking the Question) was pointed out nicely by EH. Finite elements based on the Kirchhoff theory that are not rectangular are very difficult and cumbersome to construct. These elements must have C1 continuity, namely their shape functions (basis functions) must have continuous first derivatives. This is in contrast to standard FEs - and to RM elements - which are C0, namely their shape functions are only required to be continuous. Thus, standard shape functions cannot be used with the Kirchhoff FE formulation. For rectangles, C1 shape functions can be constructed easily (as cross products of 1D Hermite cubics), but for triangles and for general quad elements, this becomes a nightmare. For example, the polynomial shape functions of a triangular element with "no tricks" must be at least of degree 5 (quintics). On the other hand, the RM theory leads to C0 FEs that are easy to implement.

Now we go back to the observation of DB and RBZ that in several commercial codes, Kirchhoff shell elements (that can be used as plate elements as well) _are_ offered. The truth is that these elements are not entirely and genuinely based on the Kirchhoff theory. There is a trick involved. These are actually elements that are originally based on the RM theory, but then they are modified to satisfy _at a discrete number of points_ in the element the Kirchhoff-theory constraint of zero transverse shear strains. This is done while still using C0 elements.

Another comment is that shell theory is much more complicated than plate theory. As opposed to the case with plates, there is no consensus on the best shell formulation to use in all cases. Therefore commercial codes are obliged to offer a large variety of shell elements to the user. While doing that, they automatically allow the use of all the corresponding plate elements, as a special case of the shell elements, when the shell is flat.

Correct answers were received from:

David Barlam, Rami Ben-Zvi, Rafi Haftka, Ehab Hamed, Zvi Zaphir.

****
Comment on the February Question of the Month
****

The Feb. Question dealt with the discretization of plate bending problems. As the answer explained, the difficulty entailed in using the Kirchhoff theory of plates with the Finite Element Method is the 4th-order derivatives that appear in the governing equation, which are difficult to handle for non-rectangular elements. A comment that should be made is that the recent methodology of Isogeometric Analysis (devised by Tom Hughes; see http://en.wikipedia.org/wiki/Isogeometric_analysis) enables the easy treatment of high-order derivatives. Within the Isogeometric framework, the Kirchhoff-Love theory of plates and shells can easily be implemented with no "tricks".

****
The March Question of the Month
****
To solve a time-dependent problem, one has to discretize the problem in time, and advance the solution in time by "time-stepping" (or time-marching, or time-integration). Usually the time-step size Delta_t is chosen to be a global constant. Are there cases when it is better to take a step size Delta_t which is _not_ a global constant? If so - what cases? How?

****
Answer
****

There are certainly situations where it is advantageous to take a non-constant time-step. One such application is the use of adaptivity in time. The solution may be slowly varying is some intervals of time, and rapidly varying in other intervals of time. In those intervals of time when the solution is rapidly varying, a small time-step may be necessary to resolve the details of the solution. But in those intervals of time when the solution is slowly varying, it would be a waste of computing time if the same small time-step is used. An adaptive scheme may estimate the local rate in which the solution changes in time, and determine the necessary (and ever changing) time-step needed as time proceeds.

In certain situations, the time-step may vary in space and not (only) in time. This may seem strange, but here is one relevant scenario. Suppose that we are interested to find the steady-state solution of a certain problem (for example, Navier-Stokes equations), and we do this by solving the time-dependent equations, and marching in time using an explicit scheme, until the solution reaches a practical steady state. This is very commonly done in CFD. In such cases, there is no interest in obtaining an accurate time evolution of the solution, since all we are interested in is the steady state. In fact, the time evolution that we obtain should not be regarded as "physical"; only the steady-state is. Now, during the time marching, it is possible to use different time-steps in different regions of the spatial domain (or even a different time-step for each element or each cell or each grid-point). The local time-step size is determined by the local material properties and size of the cell/element/grid-spacing so as to guarantee a numerically-stable solution. (We shall not elaborate on this point for the sake of brevity.)

Another situation where different time-step sizes are used in different regions is called sub-cycling. Here we _are_ interested in the time evolution, but different regions of the domain are associated with different physical phenomena, and hence require a different time resolution. For example, we may decide to use a time-step in region A which is smaller by a factor of 2 than the time-step in region B. In this case, every two time-steps the two regions which reach a common level, but in region A there will be an intermediate step for each global step.

Correct answers were received from:

Orna Agmon Ben-Yehuda, Ronen Payevski, Anne Weill.

****
The April Question of the Month
****

Consider the solution of eigenvalue (EV) problems in continuum mechanics. The most common example is structural vibration. Also, a problem related to stability, of a structure or of a flow, also usually leads to an EV problem. After discretization with finite differences, finite volumes or finite elements, one typically obtains the following algebraic EV problem:

K v = s M v .

Here K and M are square matrices, the v is the eigenvector and the s is the eigenvalue. The goal is to find all the values of s and the corresponding vectors v (aside from v=0 which is trivial) that satisfy this equation. Typically the matrix M is non-singular.

Now, this EV problem is _not_ the standard EV problem that every first-year engineering student learns about. The problem above is called the generalized EV problem (GEV), and it involves _two_ matrices: M and K. On the other hand, the _standard_ EV problem is:

A v = s v ,

and it involves a single matrix A.

Let us go back to the GEV problem. It looks tempting, since M is non-singular, to multiply the equation on the left by the inverse of M, namely M^(-1). This would yield

M^(-1) K v = s v ,

and thus would turn the GEV problem into a standard EV problem, with the single matrix A=M^(-1) K . However, in reality this is rarely (if ever) done. There are indeed techniques that convert the GEV into a standard EV problem, but not in _this_ way. There are also algorithms that solve directly the GEV (and they are implemented in various linear algebra packages, including MATLAB). Why isn't multiplying on the left by the inverse of M a good idea?

****

Answer

****

Here is the short answer. Typically, the matrices M and K - having been obtained from standard discretization methods - have nice properties, that allow the efficient computation of the eigenvalues and eigenvectors. The matrix A=M^(-1)*K usually loses these nice properties.

One property that is lost is symmetry. If M and K are symmetric, A=M^(-1)*K in general will not be symmetric.

Another property (and a very important one) is sparseness. The matrices M and K are typically sparse, namely their non-zero entries are condensed in a "band" around the diagonal, while most of the off-diagonal entries are zero. The matrix A=M^(-1)*K is much less sparse, and is sometimes completely "full".

Finding the spectrum (eigenvalues and eigenvectors) of a non-sparse non-symmetric matrix A requires more sophisticated algorithms and involves much more computational effort (and is prone to more numerical difficulties) than finding the spectrum for the generalized EV problem with symmetric sparse matrices M and K. (MB provided a nice short mathematical discussion on this, which I will omit here for the sake of brevity.)

For these reasons, multiplying by M^(-1) on the left is usually done only when M is diagonal (which is called "a lumped mass matrix" in the FE jargon). Then A has the same nice properties as K.

Additional comments received from readers are as follows:

* OL: In Algebraic Multigrid, the standard EV problem will anyhow lead to a GEV problem on the coarse levels, so there is no point to transform an originally given GEV problem into a standard EV problem.

* OL: In some contexts (e.g., homogenization, and I would add optimization) it is important to keep separate the effect of the "mass" M and of the "stiffness" K, and not to "mix" them into one matrix A.

* OL, MB: It is typical that one is interested only in a few eigenvalues and eigenvectors (for example, only in the 10% of the eigen-modes with the lowest eigenvalues). In this case, the calculation of A=M^(-1)*K will probably be more expensive than solving the GEV problem for only these modes.

* RH, SF, ZZ: In case of an ill-conditioned problem (e.g., typical for some types of elastic shell problems), the operation M^(-1)*K may worsen the ill-conditioning and introduce large round-off errors.

Correct answers were received from:

Michel Bercovier, Steven Frankel, Rafi Haftka, Oren Livne, Eli Turkel, Zvi Zaphir

****
The May Question of the Month
****

We haven't heard from our hero Roshka for a while. To remind you, Roshka is "your" employee in the computational engineering company BATALA (Benchmarks And Theoretical Analysis for Low-tech Applications). Here is a recent dialog between you two.

Roshka: Boss, I found out that you had given Yael the new multiscale project. I protest; I wanted to do it myself!

You: Well, Roshka, I wanted to give her this special challenge. She has been complaining that she had lately been assigned only non-challenging projects...

Roshka: But this project is also not challenging. I mean, it's an extremely intensive computation, but other than this it is straight forward. That's why I like to do it. I just love these brute-force head-in-the-wall intensive computations. I submit the job, and then I can take three days off while the computer is working.

You: I appreciate your motives, Roshka, and thanks for being so honest about them, but this project _is_ challenging aside from the intensive computation.

Roshka: Look, it's a multiscale problem, namely we want to capture both the large-scale phenomena and the small-scale phenomena, right?

You: Not really, we are actually interested only in the large-scale behavior. But the small-scale behavior affects the large-scale behavior.

Roshka: Whatever. Anyway, we need to take the small-scale phenomena into account, and the only way to do that is to take a fine enough discretization that will capture the small-scale phenomena. There is no way around that. You can't capture a variation of temperature that occurs along a length-scale of 1mm with a grid that has spacing of 1cm. So it all boils down to taking a very fine grid with many degrees of freedom. This is what you call a challenge that requires Yael's talents?

Explain to Roshka why he is wrong.

****
Answer
****

The key sentence in what "you" say to Roshka is "we are actually interested only in the large-scale behavior, but the small-scale behavior affects the large-scale behavior." Roshka wants to go by brute force and resolve the solution in the small scale. But this is usually not the prudent thing to do if one is interested mainly in the large-scale behavior, as "you" explain to him. There are a number of techniques that allow one to compute the large-scale behavior while taking into account the small-scale behavior, without actually using small-scale (very fine) discretization throughout.

I have received an answer from five readers, and all answers are excellent. I choose to quote a few segments from some of them.

OABY writes:

"Dear Roshka,

You do not have to use so many degrees of freedom. If Yael thinks hard enough, and digs up enough literature, or calibrates a model on her own, she might be able to find a physical model that takes into consideration the effect of the small scale phenomena on the large scale one without actually computing the small scale.

For example, there are models that use an effective temperature instead of a temperature variation, and elastic models that use the number of dislocations (Neka'im in Hebrew), but not the actual dislocations and their position and development. If you have more time and computing power on your hands, you might use mezo-scale models, such as done by Dan Mordechai at the Technion, and show the movement of dislocations but not every atom in the lattice.

And next time you have a really head-in-the-wall project, I will just give you several of them at once.... BATALA CTO"

RBZ gives 4 examples for analysis that takes into account the small scale through a sub-grid model:

"1. Fluid flow and heat transfer in porous medium may be solved by brute-force using high resolution grids (much smaller elements than the pore size), thus fully simulating the flow in the pores and the conduction, convection and radiation heat transfer in both fluid and solid domains. But if we do not wish to have all the details, just the global flow pattern, pressure drop and some sort of averaged temperature distribution of the fluid and the solid in coarser resolution, we can develop models for the pressure drop, the convective heat transfer coefficient and the radiation in a composite by focusing on a REV (Representative Elementary Volume) as a building-block.

2. Composite materials mechanical analysis is another field where the direct approach requires a very high resolution (much smaller than the smallest dimension of a single element, e.g., a fiber diameter). However, it can be modeled on larger scale, in a conceptually-similar approach to the porous medium treatment.

3. A repetitive structure (e.g. a corrugated plate) - or even a structure with symmetries - can be fully resolved, but can be replaced by a lower resolution grid if a model of the basic component is available.

4. Turbulence is a 3D transient phenomenon of flow instability in small scales. Direct Numerical Solution (DNS) of solving the 3D transient Navier-Stokes equations is adequate, but it is limited

to simple geometries. It cannot be applied to a car or an airplane or even to geometrically-complex 2D problems with today computing power (and probably for a long time into the future). Turbulence models allow solving the problems (sometimes in 2D and/or steady-state if applicable) on large scales, and is routinely supplying engineering needs. LES (Large Eddy Simulation) is a “compromise”, in which only the small scales are modeled and the larger scales are resolved."

OL writes: "One can completely eliminate the use of fine grids by first running the simulation on a small patch ("window") with different boundary conditions, and finding the dependence of the fine-to-coarse tau correction in MultiGrid on the coarse solution. Then one can work only with the coarse grid, adding the tau term to the right-hand-side of the coarse equation to boost its accuracy."

Additional answers mentioned (1) local refinement and adaptivity in space and/or time, (2) combination of analytical and numerical methods to solve problems with boundary layers and other singular perturbation problems, and (3) use of substructures and model reduction.

Correct answers were obtained from:

Orna Agmon Ben-Yehuda, Rami Ben-Zvi, Amiel Herszage, Oren Livne, Zvi Zaphir.

****
The June Question of the Month
****

Suppose we would like to simulate some mechanical process (fluid flow, deformation and stresses in a structure, etc.), but there is some uncertainty in the values of some of the parameters defining the model. For example, suppose the values of some material properties are not certain. We know the average values of these properties, but there may be some deviation. We have some statistical information on the deviation, but no more than this. A simple approach is to use the average values of the parameters in our simulation, and take some "safety factor" in the end to account for the uncertainty. Can we do better than this?

****
Answer
****

I received many answers, and together they seem to span all the possibilities.

RH, who happens to be an expert in this area, writes:

"Given statistical information about variability and errors, we can calculate the probability of exceeding a critical value (which itself is often random). So instead of designing to a safety factor, we can design to a probability of failure. However, statistical information is hard to come by and inaccurate. On the other hand, safety factors have been developed by historical experience, and so they incorporate implicitlhy good statistical information. One way of getting out of this dilemma is to use the limited statistical information, but not believe the probability of failure that you calculate, but instead just demand that the probabilistic design has the same probability of failure as the design based on safety factor.

If you have a single failure mode, the safety factor and probabilistic approach will give you the same design. However, if you have multiple failure modes (including failure due to the same phenomenon being possible at multiple locations) the probabilistic approach may give you a lighter design. A paper on this subject is: Acar, E., and Haftka, R.T., (2007) "Reliability-Based Aircraft Structural Design Pays, Even with Limited Statistical Data," J.Aircraft, 44 (3): 812-823."

OL, JT and RBZ suggest a sensitivity analysis as a pre-process. In OL's words: "run test simulations to figure out the dependence/sensitivity of the result on a unit change in the parameter values. This will yield a more accurate safety factor." OABY, OL, AY, JT and ZZ suggest ways to make many runs with different values of the parameters so as to gather a distribution of results from which the statistical information can be extracted about the solution.

AW, OABY and RBZ mention the possibility of working with a Stochastic (or Probabilistic) numerical method, e.g., a Probabilistic Finite Element Method, which takes into account uncertainties in the model parameters. There have been a number of such FE formulations, and in fact this is still an active area of research. In such formulations, the unknowns are not only the values of the sought field (pressure or displacements at the nodes, etc.) but also some probabilistic information about these values. RBZ offers the following reference for interested readers: LIU, Wing Kam; BELYTSCHKO, Ted; MANI, A. Random field finite elements. International Journal for Numerical Methods in Engineering, 1986, 23.10: 1831-1845.

Correct answers were received from:

Orna Agmon Ben-Yehuda, Rami Ben-Zvi, Rafi Haftka, Oren Livne, Jonathan Tal, Anne Weill, Asher Yahalom, Zvi Zaphir.

****
The July Question of the Month
****

Many numerical methods, like Finite Elements, Finite Differences and Finite Volumes, lead to an algebraic matrix problem with a matrix which is _banded_. A banded matrix is a matrix which contains only a small number of nonzero entries that are clustered around the main diagonal. The _bandwidth_ of the matrix is defined as the number of diagonals (around and including the main diagonal) which are not zero. (Note: a diagonal is a zero diagonal only if _all_ the entries on this diagonal are zero.)

Just as a simple example, the 1D problem given by u''=f(x) and boundary conditions u(0)=u(1)=0, gives, when discretized by the simplest Finite Element or Finite Difference method, the matrix problem Kd=F, where K has a bandwidth of 3. (This is called a tridiagonal matrix). Namely, K has 3 nonzero diagonals: the main diagonal and the two diagonals adjacent to it. The matrix itself may be very large, say 1000x1000. In 2D and 3D problems the bandwidth is typically larger than 3, but still much smaller than the size of the matrix. This property of the matrix is the key for a fast solution of the algebraic problem. [Remark: There are other ways besides bandwidth to characterize and exploit the structure of a sparse matrix, but we shall not go into this here.]

Now, it is important to realize that the numbering that we assign to the unknowns affects the bandwidth of the resulting matrix. For example, in the Finite Element and Finite Difference methods, the unknowns are values of the solution at the nodes or grid points; one has first to number the nodes, and the method of numbering affects the bandwidth. To see this, consider a 1D problem, defined in the interval 0<x<L (say, a rod of length L), and construct a mesh with 100 nodes along this interval. If you number the nodes 1,...,100 from left to right you will receive a very tightly banded matrix. But if you number the nodes randomly, say 32, 1, 78, 12, 33, 36, 90, ... there would hardly be any "band", and the bandwidth of the matrix would be very large (close or equal to the size of the matrix). The same issue arises with Finite Volumes, but there we have to number the cells, not the nodes / grid points.

Comment: When we use software to construct the model, create the mesh, etc., this business of node / cell numbering is done within the code, and we (as users) are not aware of it. Still, this process has to be done as part of defining the discrete model.

Because getting a matrix with a small bandwidth is very important for an efficient solution, the following question arises: How should we number the nodes (or cells) so that the bandwidth that we get is minimal? In other words: What is the optimal numbering of the nodes? In 1D the answer is trivial: number sequentially from one end to the other. But in 2D and 3D the answer is not obvious. Even for meshes with a simple shape, like a rectangular mesh, it is not obvious that numbering row-by-row or column-by-column (as is usually done in many in-house codes) would result in the lowest possible bandwidth. Maybe one can do better?

***** The optimization problem can be posed as follows:
Number the nodes is such a manner that the maximum difference (in absolute value) between neighboring node numbers is minimized.

The question is: What is the algorithm that gives the optimal numbering of the nodes / cells? I do not expect you to write down the algorithm itself, but just to give a very rough description of it, in words, and/or to indicate the name of the algorithm (if it has a name) and/or the names of the inventor/s of the algorithm, and/or to indicate commercial codes that are using this optimal algorithm.

****
Answer
****

First, I repeat my remark that there are better ways than "bandwidth" to characterize and exploit the structure of a sparse matrix (e.g., the use of the "skyline" or "profile" or "front" of the matrix), but we shall not go into this here.

The question was about the algorithm that gives _the_ optimal numbering of the nodes / cells. The straight answer to this question is that _such an algorithm does not exist!_

"Wait a minute", would protest Roshka, "what do you mean by that? I can give you such an algorithm myself! Simply, try all possible node-number orderings (there is a finite number of those) and pick the ordering that gives the smallest bandwidth!"

Ok, Roshka's "algorithm" indeed gives the optimal node numbering. But it is not regarded as a legitimate algorithm because it is totally impractical. For a problem with N nodes, the number of node orderings is N factorial (= N Atzeret, denoted by N!). N is typically a large number, and so N factorial is such a huge number that no computer - even the strongest supercomputer available today or in the foreseeable future - can find the optimal solution in a reasonable time.

As OL indicated, in the language of complexity theory (a topic in computer science), the optimization problem above is NP-hard, where NP stands for Non Polynomial. This means that the running time (or number of operations) of any algorithm that finds an exact solution to this problem would be larger than any polynomial function of N: typically exponential with N or even larger. (Remark: For large N, the value of N factorial is proportional to N^N sqrt(N), which is even larger than an exponential function. See Stirling's formula, http://en.wikipedia.org/wiki/Stirling%27s_approximation .)

This means that any algorithm that tackles the node-ordering optimization problem, should attempt to find a "quasi-optimal" solution (namely, a solution close in some sense to the optimal one), since it cannot find _the_ optimal solution.

Indeed, algorithms for finding quasi-optimal solutions have been proposed in the 1970's and early 1980's and some of them are used to this day in all the commercial FE codes. There is no more effort to develop better algorithms than these - they are good enough, and one cannot hope to improve the results significantly anyway. Some names (mentioned by the readers) of the inventors of these quasi-optimal algorithms are Cuthill-McKee, King, Collins, Akin-Pardue, Gibbs, Razzaque, Pina, Sloan-Randolph, Fenves-Law, and Sloan.

Correct answers were received from:

Rami Ben-Zvi, Nicolae Gluck, Rafi Haftka, Amiel Herszage, Oren Livne, Sreekanth Ravindran, Jonathan Tal, Zvi Zaphir.

****
The August Question of the Month
****

This time the Question is a bit more challenging than usual. If you do not have a clue, please do not worry!

First I would like to define the notion of "a linear method". We all know what a linear problem is. One can define a linear problem as a problem whose solution is linear with the "input". For example, if a problem in continuum mechanics is linear then

We solve the problem with load f1 and get the solution u1.
We solve the problem with load f2 and get the solution u2.
Then, the solution of the problem with load f1+f2 is u1+u2.

Now, suppose you take a linear problem and solve it approximately using some numerical method. We call the method "linear" if the numerical solution that it gives is also linear with the "input", as in the original problem. Denoting the numerical solution by u^h, we have the following:

We solve the problem numerically with load f1 and get the solution u1^h.
We solve the problem numerically with load f2 and get the solution u2^h.
We solve the problem numerically with load f1+f2 and get the solution u1^h+u2^h.

Using this definition, a "nonlinear method" is a method which, when applied to a _linear_ problem, does not have this property. Namely, if we use a "nonlinear method", the numerical solution of the problem with the load f1+f2 will _not_ be equal to the sum of the numerical solutions for loads f1 and f2 separately.

It seems logical to always use a linear method for solving a linear problem. However, there are some situations where people have used nonlinear methods to solve linear problems, with some advantages. Give an example for such a situation.

****
Answer
****

This was a tough one, and indeed I have received only two correct answers.

As RH observed, "If the approximate solution method is iterative, the convergence criterion is likely to induce non-linearity, albeit only a small amount of non-linearity." More specifically, I had in mind methods based on _mesh/grid adaptivity_. For example, suppose we start with a certain crude mesh, and our code obtains a solution for this mesh. Then our code checks the gradient of the solution at various places (say, in each element), and refines the mesh automatically at places where the gradient is larger than a certain threshold. Then the code solves the problem again with the new mesh, and again check the gradient. This process continues, until a certain stopping criterion is met.

It is not hard to see that, because of the mesh adaptivity involved, we lose the exact property of linearity, namely the method is nonlinear in the sense defined in the question. For example, if we consider two problems - one with load f1 which is distributed uniformly, and one with load f2 which is concentrated at a small region, then obviously the final meshes for the two problems would be completely different, and probably still different than the mesh that would be obtained for a problem with the load f1+f2.

Another example for a nonlinear method is the method of "Moving Finite Elements" of Miller and Miller (see http://www.jstor.org/stable/2157254). The idea in this method is to find _simultaneously_ the solution of the problem and the optimal location of the finite element nodes. Instead of fixing the mesh, and looking for the solution that gives the minimal energy, this method looks for the solution _and_ the node locations that yield the minimum energy.

A third example for a nonlinear problem was offered by OL: "To capture a shock profile in a linear time-dependent problem, one can sometimes avoid non-physical oscillations and maintain the monotonicity of the numerical profile by adding a nonlinear viscosity term or using a limit on the absolute value of the derivative (abs value is a nonlinear operation)." Indeed, various shock capturing methods are nonlinear in nature, even when the original problem to be solved is linear. If we try to solve a problem involving a shock (say, a sudden step in space in the solution) while using a standard method that is not capable of representing such shocks, we would get a very bad solution, with wide and strong spurious oscillations. There are linear methods to handle this situation, for example by using linear artificial viscosity, but they tend to "smear" the shock in an exaggerated way, so we get a very nice and smooth solution, but to the wrong problem... This is why some nonlinear shock-capturing methods have been proposed, which on one hand get rid of the spurious oscillations, but on the other hand do not smear the shock significantly.

Correct answers were received from:

Rafi Haftka, Oren Livne.

****
Comments on the August Question of the Month
****

The August Quest of the Month asked about "nonlinear methods" for linear problems. I received the following two nice comments, that relate to important nonlinear methods in contexts that are tangent to CM.

Achi Brandt writes: "Perhaps the most famous nonlinear approximation to a linear system is the density functional theory (DFT) approximation to the (3N+1)-dimensional Schrodinger equation for N particles."

Eli Turkel writes: "As an aside on using nonlinear shock ideas for linear discontinuities, there is an important application in image analysis. It is important to use nonlinear filtering and other nonlinear methods for image restoration, since with linear methods anything that reduces high frequencies smears discontinuities, i.e., edges of objects."

****
The September Question of the Month
****

This question has a historical flavor. The Finite Element (FE) method started to be used in engineering from the mid 1950s, together with the availability of computers. Back then it was used only for problems of structural analysis and solid mechanics. In fact, it was perceived as a method suitable only for _solid_ mechanics. One can find quite a few old books on FEs which take this view. Only about 15 years later (in the early 1970's), the method started to be used also for _fluid_ dynamics problems. (On the other hand, Finite Differences were already used for fluid dynamics problems much earlier.) Today, it is claimed that about 50% of industrial fluid-dynamics simulations are done with FEs. Why did it take 15 years to make use of FEs for fluid problems?

****

Answer

****

Before relating to the answer, I will mention that some readers expressed their surprise regarding my statement that "today, it is claimed that about 50% of industrial fluid-dynamics simulations are done with FEs." This is a quote heard at lectures of one of the world leaders in FEs today. ET remarks: "For incompressible flows I could believe that 50% are done by FEM. For compressible flows I highly doubt that... I strongly suspect that most commercial or industrial codes are still using finite volumes." This makes sense to me. Perhaps the speaker was talking only about incompressible flows.

And now to the question itself. What I had in mind was the following answer. When the FE method was introduced by engineers at the mid 1950s, it was presented and was thought of as a method to obtain an approximate solution for the problem of minimizing the total potential energy of the system. All the old books on FEs present the method in this context of potential energy minimization and the Rayleigh-Ritz technique. But using this approach, only problems which _have_ a potential energy can be solved. Problems of elastic solids and structures (even nonlinear problems!) have a potential energy, and hence could be attacked by FEs. Most problems in fluid dynamics (potential flow is an exception, of course) do not possess a potential energy, and hence - by the view of that time - could not be solved by the FE method. Even the simple scalar linear problem of advection (convection) - diffusion could not be solved by this approach. Only in the early 1970's people realized that the FE method can be used even when there is no potential energy, by using the weighted residual approach (or more specifically the Galerkin approach). Once this was realized, the door to solving fluid problems with FEs was opened. This answer was offered by RH, and was also partly mentioned by AH and OL.

Of course, there is another, more prosaic, reason that may also be true. The inventors of FEs in the 50's (Clough, etc.) were structures people, and we know how long it has taken for ideas to pass from one field of application to another, and from one community to another, especially in those days without internet, paper archives, etc. So sometimes, just because of an accidental historical reason, a certain method could have reached one community much later than it reached another. Today this happens much less, since there is much better flow of information between communities owing to the electronic media.

I received additional very good insights. RBZ describes the difficulties characterized by fluid problems, and concludes: "All this required a lot of generalization and adaptation of the 'old' FEM to be able to successfully cope with real-life fluid-dynamics problems, and this process takes time." AH's answer is similar, and he also mentions that various problems in fluid mechanics require stabilization, which the original FE method did not possess. He sent me a copy of pp. 1 and 2 from Antonio Huerta's book, which exactly discuss this issue in a very nice way.

ET writes: "The answer to your question depends strongly on whether we are talking about compressible ot incompressible fluid dynamics, although both are nonlinear. For incompressible problems the difficulty is that the pressure constraint is indirect and one needs special FEs to handle the difference between the velocity and pressure. For compressible fluids the equations are completely nonlinear and so much easier to treat with finite differences/finite volumes. Thus when using spectral methods one usually uses pseudo-spectral rather than a Galerkin approach. In fact even for linear hyperbolic wave equations standard FEs can be unstable unless one introduces some upwinding which is less 'natural' for FEM."

Correct answers were received by:

Rami Ben-Zvi, Rafi Haftka, Amiel Herszage, Oren Livne, Eli Turkel.

****
The October Question of the Month
****

The following 'question' is more of a 'discussion'. Perhaps we should call it "The Discussion of the Month".

About two weeks ago, an experienced researcher in numerical methods talked to me about a method that he is promoting. He said: "A version of this method was proposed already 25 years ago, but despite its relative advantages it was not accepted well, since people didn't like the fact that it led to twice as many unknowns than the standard method. But this was relevant when computers were slow. Today we have very fast computers, so this issue is not relevant anymore. And I think that the advantages of the method make it really attractive."

The 'question' is to discuss the argument above. Do you think such an argument may be in/correct? Examples are welcome, although a general discussion is also fine.

****
Answer
****

First I will write down my own thoughts about the argument quoted in the Question. (This is the privilege of the editor.) One reason that makes this argument invalid, in my opinion is the following. If at one time the advantages of the numerical method were regarded not important enough to compensate for the fact that it is less efficient than its competitors (e.g., requires twice as many unknowns), then this would remain the case even when computers become much faster. That is because the size of the problems we wish to solve also increases all the time. With computer power comes the appetite for ever larger and harder problems, which means that efficiency will always remain important.

I have received three very nice inputs from readers.

OL expressed the same idea as above: "This argument can be used in reverse: as faster computers are now available, people will want to solve bigger problems on them, not the problems of the past. So twice as many unknowns should still bother them today as it has back then. So I'd say it is not a good argument."

OL also writes: "Twice as many unknowns shouldn't be an issue if the complexity doesn't change. It's just a constant factor in runtime/storage." [DG: This is a typical view of CS people, namely that there is no difference between 100,000 and 200,000 DOFs. Engineers in industry who have to actually run codes for huge problems do not always agree to this.] An exception might be when the variable doubling is due to complex arithmetic vs. real (real and imaginary part instead of just real), and during the course of time special-purposes architectures that can treat complex arithmetic at the hardware level have evolved. Then it does make a difference, perhaps of at most one order of magnitude, though, not more. Examples include linear algebra routines like matrix multiplication and inversion."

RG writes:
"I will say that the argument is simplistic. I doubt that size is the only drawback of the method mentioned by the experienced researcher. Indeed, people have been using the stabilization method of L. Franca and T. Hughes for incompressible viscous flow despite the fact that we know how to obtain the same accuracy with 4 times less pressure unknowns in 2-D and and 8 times less in 3-D. The main reason of the success of the Franca-Hughes method is that it is simpler to program than its competitors, despite the fact that it involves stabilization parameters."

AW writes:
"I think moving mesh methods are a prime example of a class of methods where such a comment is relevant. One basic question I would ask is: for a given required accuracy for a particular problem, how much computational effort does it take to compute the solution with traditional methods vs. the method with twice as many unknowns under question? I also wonder if the method has an advantage of being able to calculate problems that traditional methods cannot otherwise resolve, problems that we are interested in solving? I would also want to know if for a storage intensive problem where the number of unknowns is astronomical and barely feasible with a traditional method, is it also feasible to use the method under question given the computational storage requirements available?"

Good answers were received from:

Roland Glowinski, Oren Livne Abigail Wacher.

****

Comments on the Oct. Question of the Month

****

The October Question of the Month dealt with an argument saying that a certain numerical method which was regarded as inefficient in the past, should become much more attractive now because "computers are now much faster." Here are two late comments in this regard.

1. Micha Wolfshtein comments, very correctly, that an increase in the number of unknowns does not necessary mean a reduction in efficiency. The whole idea of high-order methods is to get higher accuracy per a computing-time unit (or get the same accuracy with less computing time). In Micha's words: "This is typically the rationale behind high order methods. You increase the number of unknowns but reduce the computer time. There are many examples to such methods. Even the 4th order Runga Kutta method is an example, but there are examples of this kind in 2D and 3D problems. I believe that many DNS runs do this."

2. Among the answers to the question, the "complexity argument" has been mentioned. (Complexity theory is an established theory in computer science.) This argument says that increasing the number of unknowns by a factor of 2 does not change the complexity of the computation. As an example, suppose a certain method requires N^2 computer operations, where N is the number of unknowns. If another method requires 2N unknowns, the number of operations would increase by a factor of 4. By definition of the term "complexity", N^2 and 4N^2 have the same complexity. (In contrast, N^3 has a higher complexity than N^2, and also higher complexity than 500 N^2. This is because when talking about complexity we always look at N arbitrarily large.) So, adopting the complexity argument, a method with N unknowns has exactly the same complexity as a method with 2N unknowns, and hence both methods have the same efficiency (in the sense of complexity theory).

This argument is theoretically correct (by definition), but in many cases it may be totally impractical. For example, suppose that the limit of your computer is such that it can solve problems with N unknowns at most. (Say, N=500,000.) This means that it can solve problems of size N but cannot solve problems of size 2N. So even though the two problems have the same complexity, there is a difference here: you can solve the first one but you cannot solve the second one on your computer! It is a matter of "to be or not to be", and no complexity arguments would change this fact...

****
The November Question of the Month
****

We sometimes solve one-dimensional (1D) problems, two-dimensional (2D) problems or three-dimensional (3D) problems. More precisely, we use a 1D mathematical/numerical model, or a 2D model or a 3D model. But sometimes there is a need to used mixed-dimensional models. This means, for example, that a part of the model is 2D and another part is 3D. (One challenge in this context is the correct coupling of the two parts of the model, namely how to "connect" the 2D and the 3D parts, but we are not asking about this here.) The question is: under what circumstances such a need arises? Give at least one example for an application that involves a mixed-dimensional model.

****
Answer
****

Here are some situations where the need to couple a low-dimensional model (say 1D) and a high-dimensional model (say 3D) arise:

1. Blood flow simulation. Suppose we would like to model the blood flow in a certain artery in the human body (say, when the artery is partly clogged and we wish to see how this affects the blood flow). We model the artery using a very detailed 3D model. Now, in order to obtain accurate results we wish to model the rest of the blood system along with this artery. But we cannot afford to model the entire blood system in the same detailed 3D way. So we model the rest of the blood system as a "tree" of 1D-like pipes. Then we need to couple the 3D model with this 1D "tree" model. The group of the Swiss/Italian researcher Alfio Quarteroni has done exactly this type of modeling. RG proposed a similar example: "In the cardio-vascular system, heart valves are essentially elastic shells or membranes and as such are approximated by bi-dimensional surfaces when neglecting their thickness; they are coupled to a blood flow modeled by the 3-D Navier-Stokes equations, typically."

2. Hydrological and geophysical flow models. Here the low-dimensional region represents a collection of channel-like entities (like rivers)
and the high-dimensional region is that of a large water body (like a lake).

3. Modeling of elastic structures. Many types of structures (aircraft, car, ship,...) contain thin-walled components (beams, plates, shells) which can be modeled using 1D or 2D models. However, these thin-walled components are sometimes connected to bulky parts which must be modeled as 3D bodies. Also, the thin-walled components are connected to one another through joints, and these joints (if we wish to model them accurately) behave like 3D bodies. Hence the need arises to connect low-dimensional structural components to high-dimensional ones. OL

suggested this possibility. RG gave a very simple example: "Consider the antenna of your car:the antenna is essentially an elastic beam (1D) and your car an elastic shell (2D)." RH proposed another example: "A bridge may have cables that are best modeled with 1D elements and plates that are best modeled with 2D elements (which are really 2.5D elements, because the thickness is also modeled approximately)." RBZ proposed an example of a membrane-rod system. ZZ discussed a crack model where the crack region is modeled as a 3D body, and the remote region is modeled as a lower-dimensional part.

Other examples, that we shall mention only briefly, were proposed by AH (a long pipeline connected to a heat exchanger), RBZ (a heat radiation system), RS (simulation of turbulent combustion), and LP (a biological model involving a cavity-membrane system).

Correct answers were received from:

Rami Ben-Zvi, Roland Glowinski, Rafi Haftka, Amiel Herszage, Oren Livne, Lydia Peres, Ravindran Sreekanth, Zvi Zaphir.

****
The December Question of the Month
****

Some problems in CM involve _a moving interface_. An example is the problem of melting or solidification. Consider a lake that freezes in the winter and melts in the summer; let us concentrate on the freezing process. (This Question is very appropriate for the weather today in parts of Israel...) The freezing does not occur uniformly in the entire lake, but different regions freeze at different times, depending on the distance from the shore, the varying depth of the water, and other factors. The problem is to simulate the freezing process, and to predict when each point in the lake will freeze. (A similar problem, the freezing of the oceans near the poles, is one which navies around the world are concerned with, so that ships and submarines will know when they can cross.)

An important feature involved in such problems is the motion and shape of the water-ice interface. This interface is unknown, and is the main thing that we need to find when solving the problem. In addition, we need to find the temperature distribution in the lake, because it will determine the interface motion.

One type of approach to solve such moving-interface problems is what is called front tracking. In this approach we track the interface location at each instant in a direct manner, and we keep changing the discretization (finite element / finite difference / finite volume mesh / grid) to fit the varying fluid-solid configuration. Namely, in each time step we find not only the temperature distribution but we find directly the new location of the points of the interface (which are determined by a balance-of-energy equation). Then we reconstruct the discretization for the new configuration, and we proceed to the next time step.

This approach works well in some situations, mainly when the interface motion is relatively simple or slight. However, there are situations when it is very difficult to use this approach. Consider for example two separate frozen regions that merge into a single frozen region. Most front tracking algorithms are not capable of handling such a situation.

What alternative approaches are possible for the moving interface problem that are more robust in complicated cases?

****
Answer
****

As RBZ indicated, this sort of problem is called a Stefan problem, named after the Solvene physicist Jozef Stefan (see http://en.wikipedia.org/wiki/Stefan_problem).

It turns out that among the readers there is a big expert in such methods, GL, recently retired from Rafael, although his expertise comes from problems a bit different than the freezing problem I described. I will quote a part from his answer.

"When the shape of the interface becomes strongly distorted, it is better to perform a multi-material calculations. That is, several materials may be present in each computational cell and the interface is cutting through the mesh lines. To follow such an interface there are several methods I will mention:

1) The direct front-tracking you mentioned is potentially the more precise but by far the most complicated as the front topology may change with the time. James Glimm at Stoney Brook univ, claims that they successfully use it for instability growth calculations.

2) Multi-dimensional VOF (volume of fluid) interface reconstruction is the most commonly used method. Essentially, if you have two fluids and you assume that the part of the interface cutting a computational cell is planar, than once you know the normal to the interface, its position is uniquely determined. I might have been one of the firsts to use this method. David Young did describe it more in detail. There are many improvements as how to better define the interface direction and how to proceed with cells having more that one materials.

3) The Level Set method (James Sethian, Berkely Univ.) is a simpler way to follow a moving interface. It is the somewhat similar to the way you plot Level Set Contour lines. If you define a function giving the distance to the interface at each mesh vertex, the interface position, which is the Level Set with value zero, can be obtained by simple interpolation. The value of the function

can be advanced in time by simple convection with the local fluid velocity. The drawback of this method that it is not conservative. Thus the interface position is not exactly consistent with the volume fractions." OABY also suggested this method.

Additional ideas are as follows.

4. OL: One may use "moving finite elements" to follow the evolution of the interface location. This may still encounter difficulties when two regions merge into one, but this sometimes can be amended by local corrections to the mesh.

5. RG: There are phase field methods for front capturing, for example those based on Cahn-Hilliard type equations. I will explain the basic idea. We add to the usual field (temperature) T(x,y,z) a binary field, called the "phase variable" and denoted phi(x,y,z), that tells us if at point (x,y,z) there is water or ice. For example, phi=0 may indicate "water" and phi=1 may indicate "ice". The Cahn-Hilliard equations are differential equations that govern the behavior of the phase-field variable phi(x,y,z). We solve together the heat equation and the Cahn-Hilliard equations and obtain simultaneously the temperature T(x,y,z) and the phase phi(x,y,z). Of course, the solution of the Cahn-Hilliard equations will not give us phi that is strictly 0 or 1; we will always get a continuous field that varies between 0 and 1. But if the modeling is good, then in most of the domain phi will be close either to 0 or to 1, and there will be a thin layer in which phi will change rapidly from almost 0 to almost 1. The center of this thin layer is regarded as the current location of the interface. RBZ also suggested a version of this method.

6. AW: "The enthalpy formulation can be used when front tracking methods are not applicable – discretization variable is no longer temperature but energy , without mentioning the actual position of the interface." A good reference for this formulation in the context of the FE method is the book of White, An introduction to the FEM with Applications to Nonlinear Methods.

7. ZZ and RG both mentioned that a similar situation - of the need to track an interface - occurs in contact problems. Thus, the computational methods developed for contact problems may be helpful. RG: "Variational inequalities models provide also interface capturing for contact, friction and some plasticity problems."

Correct answers were received from:

Orna Agmon Ben-Yehuda, Rami Ben-Zvi, Roland Glowinski, Oren Livne, Gabi Luttwak, Anne Weill, Zvi Zaphir.