ASSESSMENT TECHNIQUES FOR COMPUTER MODELS OF SOUND PROPAGATION

by

David H. Wood
SACLANT ASW Research Centre
La Spezia, Italy

ABSTRACT

Computer models of sound propagation have exactly three drawbacks: the model only approximates reality, the computer program only approximates the model, and the cost approximates infinity. The only way to assess these shortcomings is to compare the computer model's performance against data: experimental data, synthetic data, and financial data. Examples of these comparisons will be featured. Comparison with theoretical examples within the scope of the computer model is the best way to assess the relation between the model and the computer program. Financial data on cost of speed, accuracy, size, and special features is spotty, but a few examples will be shown.

Computer models of sound propagation have three drawbacks: the model only approximates reality, the computer program only approximates the model, and the cost and computer requirements approximate infinity. The only way to assess these shortcomings is to compare the computer program's performance against data: experimental data, synthetic data, and financial data.

Experimental Data is the final test of the model's relation to reality and an abundance of it could measure the other shortcomings by statistical methods. Unfortunately, there are too few experiments and fewer still with dependable, accurate data.

Synthetic Data poses the most rigorous test of the programming of the computer model. As in the last paper, by numerical simulation of an ocean experiment, synthetic data can suggest that the model reflects reality. Two sources of synthetic data are computer models and theoretical examples. Comparison with theoretical models throughout the area of their overlapping capabilities, whether realistic or not, promotes confidence in the programming of both models. Comparison with theoretical examples is the best way to assess the relation between the model and the computer program. This includes comparison with special models, which are simplified and accurate but otherwise impractical, these models being specifically developed for this purpose.

Financial Data on the cost of speed, accuracy, size and special features is scarce, probably because we don't yet have even one dependable model.

Let me show you (Fig. 1) some of the interrelated concepts that we have to work with. Here we have a red disc representing reality, partially overlaid with a blue disc representing the wave equation. If I had heard Di Napoli's talk before, I would have made the red disk much, much larger. Now I place a yellow disc representing the model. Notice that part of it lies outside the wave equation. Models sometimes have unrealistic features such as speeds given by complex numbers. In the orange region, we have, for example, absorption, a reality not modelled by the wave equation. In the green area, we note that many models can deal with unrealistic extreme changes in speed. In the purple area, we note that most models have layering and can only approximate the very smooth sound-speed profiles that may be found in the ocean. In the blue area, we might find zero frequency; in the red area, non-linear effects. Let me overlay on this diagram the computer program in question. I have deliberately made its representation a little spotty. Little, because it cannot fully realize the model; and spotty, because the capabilities of the

program are seldom known -- even to its author. Authors of computer models write reports about them. In the reports we see the models validated: they are compared with experimental data or with other large, general-purpose, sound-propagation computer models. Do they agree? No! Of course we would be shocked to see experimental data points fall exactly on the computed curves. On the other hand, I am shocked to see the two models' curves fail to lie exactly over each other. I think that the most we can conclude from these comparisons is that the models do not contain programming errors that have a gross effect in the test case shown. Now, I say this is not validation. I say that validation has to proceed in two separate and distinct phases: first, we have to be assured that the program truly represents the model; second, we want to see that the program approximates reality.

Reading these reports, we are expected to assume that all programming errors have been eliminated — the authors don't say. It is considered quite impolite to intimate that a particular program might contain errors. However, if the authors have gone to all the work of testing their programs as thoroughly as I am about to suggest, I am surprised that they don't say so.

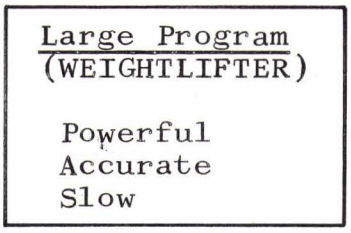
In the last paper, Di Napoli told us that in a typical realistic case the models typically differed from FFP be more than 5 dB. He concludes that the models cannot produce accurate results in this test case. I don't necessarily agree. I think that Di Napoli's comparisons are probably a statistical demonstration that these models contain programming errors. I feel that before models are compared, we have to be sure that each model is correctly programmed. Let me outline the three distinct types of computer programs that I think are required to obtain the desired modelling of reality.

First, we need a type of program, I call an Archer. Typically, this

Small Program (ARCHER)
Accurate
Simple
Unrealistic
Slow

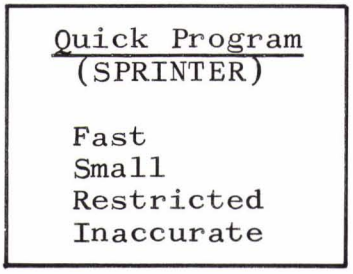
is based on a class of theoretical examples. Its main attribute is accuracy, but simplicity of programming is important, too. It will usually not be restricted to realistic conditions, but that doesn't matter. It is not to be used to model reality, it is to be used to detect programming error in the next class of computer model. Therefore, it is more important that it has capabilities in common with the next model than that it has features in common with reality. Its least important feature is speed. It is to be tested against theory and against other Archer-type programs, when they have an application in common.

The second type of program, I call a Weightlifter. Its main



attribute is power. It is to cover as much of the abstract model as possible, including unrealistic cases, which often exaggerate the effect of previously undetected errors. It is accurate because this helps debug the program by comparing it to Archer-type models; and because it will be used to check that the third class of model retains sufficient accuracy for applications (which is not much). Speed is not expected because it is incompatible with the other requirements. This is a program that accurately mimics the abstract model. As such, it is suitable for comparison with any synthetic data or experimental data.

The third class of program, I call a Sprinter. Speed and perhaps



size are all-important. If restricting the program to realistic problems will make it faster or smaller, fine. We are happy to sacrifice quite a lot of accuracy, too, if that will help. This program is to be evolved from a weightlifter. It will be free of the errors that were eliminated earlier. It is to be compared to its weightlifter version to verify that it has not lost too much accuracy in realistic cases. And now, finally, it is tested against experimental data.

My thesis, then, is that we need three distinct types of programs. Certainly, if we want fast, realistic programs, we will have to have powerful programs and accurate programs. All three types should be equally documented and distributed. One model developer I know publishes weightlifter programs, but never has anything to say about his archer programs. As a matter of fact, I have one that I cherish that I got out of his wastepaper basket. At the very least, any organization that is interested in evaluating sound propagation models ought to have a collection of all three types. At SACLANTCEN we are creating a super computer model, SOLMAR, that includes many models of all three types under a common user-oriented input-output language. I believe that is an important step towards the day when we can have meaningful comparisons between models.

I want to show you a few examples, but let me apologize in advance. I show the results of these three models not because they are good examples of the three types of programs I spoke of. They are only rough approximations of an Archer model, a Weightlifter model, and a Sprinter model. However, they are models that are running at SACLANTCEN, and illustrate typical results in comparison of models.

Figure 2 shows the geometry of the example: a uniform layer of water over a plane, rigid bottom. I will show four comparisons, using four frequencies. Everything else is held constant: only the frequency changes.

1) At a radian frequency of 160 (Fig. 3), an Archer-type model, based on the exact solution of the wave equation given by Brekhovskikh, generates the monotonic curve for random phase addition and the humped curve for coherent phase addition. A Weightlifter-type model, a normal mode program by F. Jensen of SACLANTCEN, gives a curve for the coherent field indistinguishable from Brekhovskikh. The FACT model, a Sprinter-type model, gives the same result for both coherent and incoherent phase additions, some 3 or 4 dB below Brekhovskikh's random phase curve.

2) At a radian frequency of 800 (Fig. 4), we obtain the same type of results.

3) At a radian frequency of 2130 (Fig. 5), the coherent field is much more intense than the random phase approximation. Jensen's mode program no longer agrees exactly with Brekhovskikh. The FACT model again gives the same result for the coherent and random phase fields.

4) At a radian frequency of 2131 (Fig. 6) -- recall the last example was at 2130 -- the coherent field has dropped down much closer to the random phase approximation. Jensen's normal mode model again agrees with Brekhovskikh. The FACT prediction is unchanged.

So even in these simplest examples, we see that the comparisons are puzzling enough to demonstrate the need for maintaining three distinct types of computer models: accurate Archer models, powerful Weightlifter models, and fast Sprinter models.

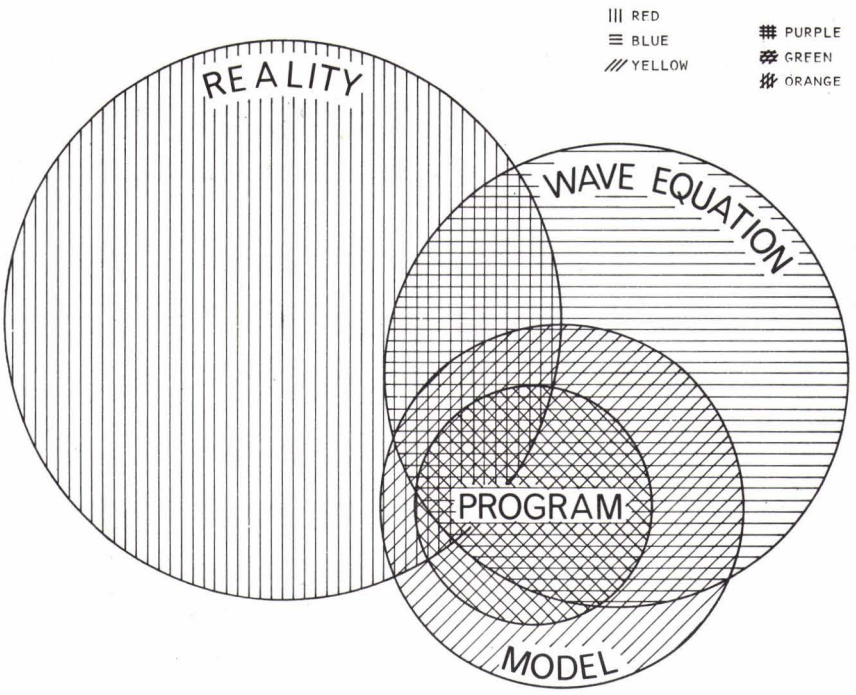


FIG. 1

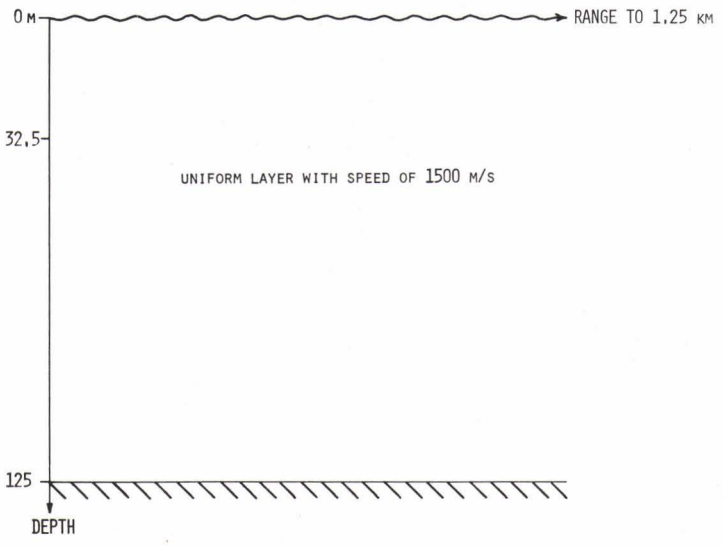


FIG. 2

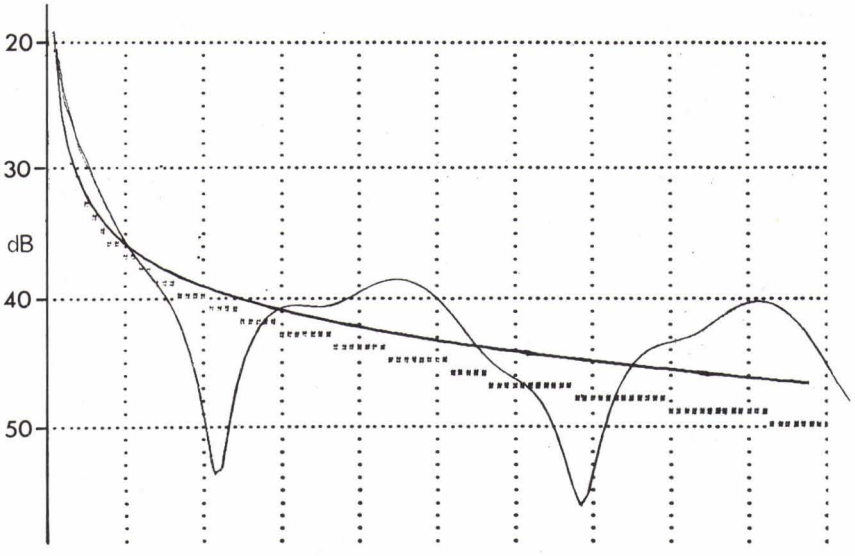


FIG. 3

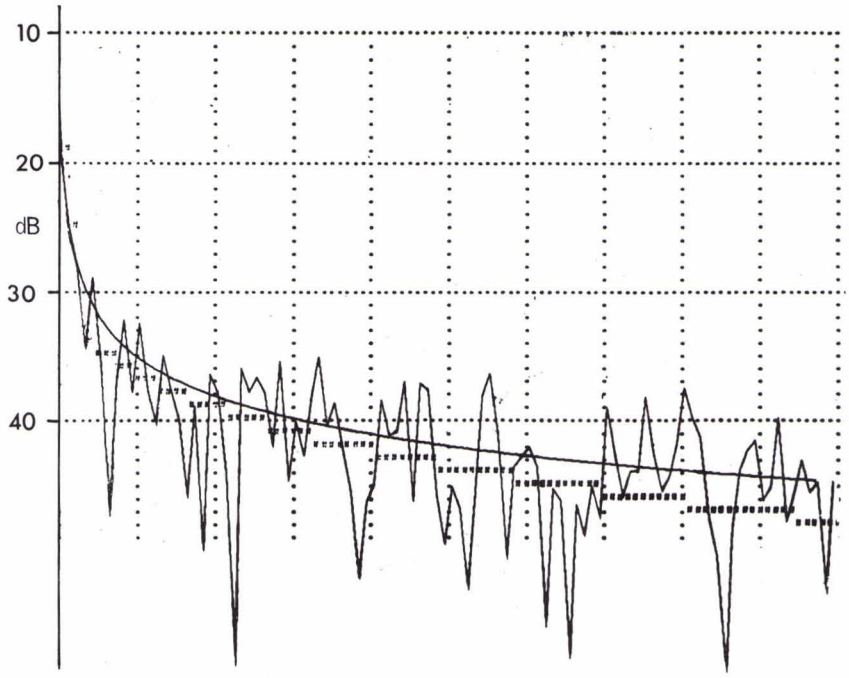


FIG. 4

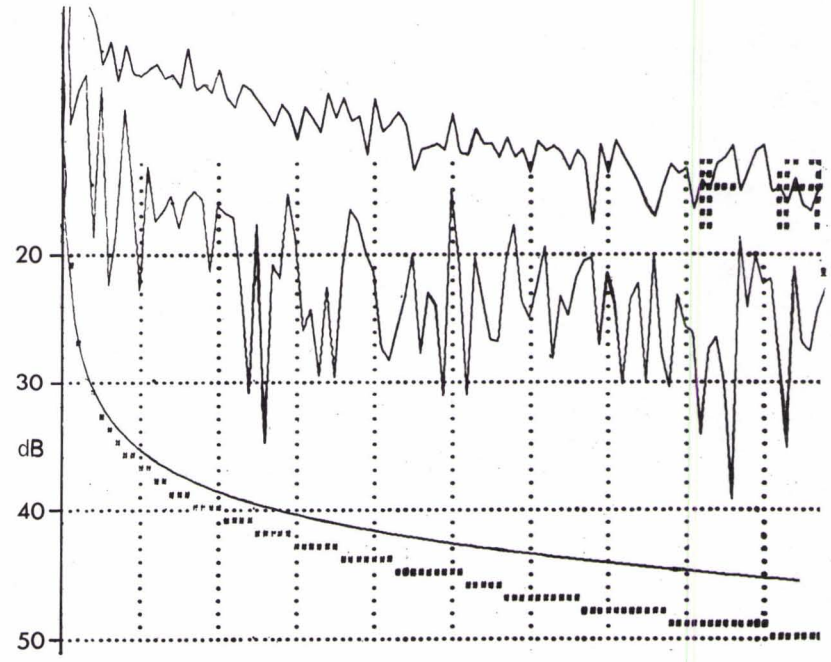


FIG. 5

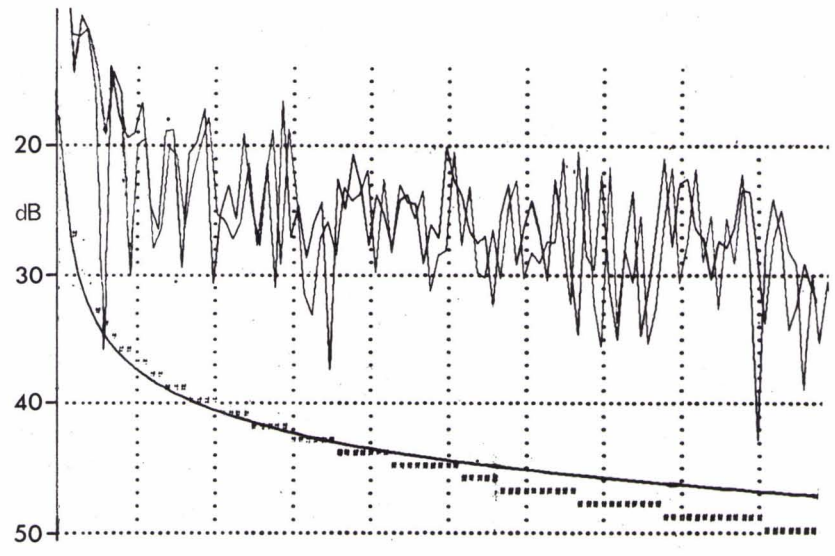


FIG. 6