

STOCHASTIC MULTIDISCIPLINARY IMPROVEMENT: BEYOND OPTIMIZATION¹

Jacek Marczyk, Ph.D., Senior AIAA Member
 EASi Engineering, Madison Heights, MI 48071, USA

ABSTRACT

Contrary to popular belief, contemporary optimization practice in CAE is an outdated and limiting paradigm. Given the performance of today's computers, the practice of optimization is difficult to justify, not only from a philosophical, but also practical standpoint. It is argued that optimization should be substituted by the more complete and simpler stochastic simulation, which apart from enabling a rigorous treatment of uncertainty and complexity, establishes a firm conceptual platform for the unification and simplification of CAE. The complexity of today's optimization tools, together with the multitude of available algorithms, very often diverts the engineer's attention from physics to purely numerical aspects. It is shown, how in the presence of uncertainty, robustness is a much more valuable feature than optimality and how stochastic simulation, based on Monte Carlo techniques, can replace all forms of optimization.

Introduction and Background

Contemporary CAE is stagnant and fragmented. If one takes a step back and looks at what the CAE market offers, one notices excessive articulation in almost every direction. There are tens and tens of tools that replicate each other's capabilities and features. As Kuhn points out in his influential book, *The Structure of Scientific Revolutions*, such articulations are a characteristic proper of a state of crisis. Whenever a discipline of science runs out of ideas, the first symptom is fragmentation and fractalization of the same concepts. Small incremental improvements and refinements, but no major and significant innovation. One such overly articulated concept is the popular and trendy Multi-Disciplinary Optimization, MDO. What comes after crisis is revolution. Crises precede revolutions, and CAE is precisely in this state now. According to Kuhn, every time a revolution in science takes place, a change of paradigms is the result.

Contemporary CAE is medieval in its structure and orthodoxy. Almost "geocentric". The fundamentalism in today's CAE hinges on a few dogmatic (untouchable) pillars: determinism, reductionism, and optimality. Although these

concepts are highly related, the analysis of the significance of optimization and optimality shall be the focus of this paper. A little bit of history is necessary here.

When something is optimal, it is essentially perfect, impossible to improve, non-plus-ultra. But why do humans pursue optimality, this state of grace, so fiercely? The roots of this are to be found, not surprisingly, in religions. In medieval Europe, governed by the Inquisition, and based on the pseudo-science established by Aristoteles, perfection was the name of the game. Since God was perfect, humans were obligated to seek perfection everywhere, in every angle of the universe. Geometry, the motions of planets, architecture, were all based on perfect geometrical figures and constructions (circles, triangles, pentagons, etc.). Even Copernicus, when he discovered that our solar system was in reality heliocentric, did not dare to propose elliptical orbits, since an ellipse was not a perfect geometrical figure. It was not divine enough, politically incorrect. It was then Kepler who had the courage to replace circular orbits by the more vulgar ellipses. Science has suffered greatly due to the search of perfection. Even Darwin had difficulties in having his theory of evolution of the species accepted. Surprisingly, it was not because of the fact that man, according to Darwin, descended from the ape, although

¹ Copyright 2000 by J. Marczyk. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

Vice President Advanced Technologies

already this assumption was on a frontal collision course with the dogmas of his days, it was because Darwin's theory, that evolved the simplest organisms to the human being, was an open-ended process. It did not stop at the final and perfect (optimal) result, man. Now, since man is an image of God, and therefore perfect, any theory, whether based on evolutionary or single-shot creation, has to lead to this optimal design. Since, according to Darwin, evolution never stops, it does not guarantee a perfect and happy ending, and is therefore unacceptable. What Darwin's contemporaries did not accept was the fact that in his theory there was no Master Choreographer who would decide what the final result was to be.

Today in CAE, but also in other branches of science and life, the situation is strikingly similar. Western culture, in particular, still relies on dogmatic and authoritative forms of education that make people want certainty instead of knowledge. Uncertainty has been axiomatized out of CAE, because it is politically incorrect and because it inhibits the use of many elaborate, and often elegant theories and tools that abound, to the delight of the Establishment. Uncertainty is incompatible with optimality, and is therefore an uncomfortable thing to deal with. Optimality and uncertainty can't coexist. Uncertainty eliminates the possibility of imposing top-down choreographies that produce pleasing and a-priori known results. What is argued, and argued in this paper, is that the basic idea behind optimization is wrong and that the optimization paradigm is responsible for the fragmentation in CAE. However, as the history of science teaches us, major advances in any discipline have taken place when generalizations and simplifications of paradigms were made. This paper is about the next paradigm: simulation.

What Is Wrong With Optimization?

There is something terribly wrong with optimization. Optimality and optimization don't exist in nature. Nature only produces designs that are fit for the function and the mechanism it chooses is not optimization, it is spontaneous self-emergence, or self-organization. Let us consider, as an example of a fantastically complex and efficient system, the system of veins and capillaries in a human body. Is the topology of the vein network the result of an optimization problem with millions of variables that nature has solved? Clearly the answer is no. In fact, every individual has a different one, although dominant features and patterns may

easily be identified. The details are not repeatable. How could a similar problem even be formulated in terms of optimality? What is an optimal vein network? Clearly, on paper everything is possible. Is the vein system perfect? Who cares? As long as it works. This is the main point. Nature settles for design that are good enough. Formulating problems in an optimization context is equivalent to making them extremely complex, much more than necessary. There exist very simple tools that can "solve" extremely complex problems if we are willing to accept a paradigmatic change. Nature bases its most powerful mechanisms and machinery on the recently discovered laws of complexity and spontaneous self-organization. Complexity is a phenomenon that arises on the frontier between chaos and order, in a sort of phase transition. It is on this frontier that Nature is most prolific and creative. Life, in fact, has emerged spontaneously from an auto-catalytic chemical environment obeying a small set of very simple laws. Complexity did the rest. Complexity, the study of which is being pioneered by the Santa Fe Institute, has been claimed to be the next major revolution in science. In CAE, we could see Monte Carlo simulation as being the "equivalent" to self-organization.

There are always many ways to formulate the same problem. The best one is the simplest that works. However, human nature finds complex theories more credible, and certainly more appealing. Why has it taken nearly twenty centuries to abolish Ptolemy's model of the solar system based on epicycles? Because it predicted the motions of the planets correctly! In fact, philosophers have shown that many theories can fit the same set of observed data. Most of them are wrong. For this reason, it sometimes occurs that baroque and excessively ornamented numerical procedures, composed of Design Of Experiments, response surfaces, genetic algorithms and so forth, yield results that can be contrasted with measurements. Let us look at another example.

In the brain there are 10 to the power 12 neurons. Imagine that each neuron is a "design variable". Does it make sense to talk of sensitivities of each single neuron with respect to some performance descriptor of the brain? Does there exist one single most important neuron in the brain? Clearly the answer is no. In complex systems, it is the connectivity of the components and their basic properties that establish the global behavior, not the details of the

components. Therefore, if we want to change, or improve, the global behavior of a complex system, what is required is a collective change of the fundamental properties of the single components, or a change of topology. Approaching complex problems on the basis of sensitivity, or optimization, is a cause that is lost from the very beginning, although from the point of view of computer algebra everything is possible (people call these problems Grand Challenges, such as the Traveling Salesman problem with millions of cities). In complex systems, even minute changes in the properties of the operational environment can often have macroscopic implications. The point is therefore how to steer and manage the intrinsic uncertainty that all complex systems possess, so that they can co-exist with their changing environments in a robust and controllable manner.

Looking for a global optimum in a complex system is like trying to hit a bullet with a bullet. Although in theory any problem may be formulated in this fashion, complex optimization problems are essentially ill-formulated and exceptionally fragile. This fragility is apparent when one changes algorithm, or even starting point. An important source of debility of the optimization paradigm is the fact that while engineers concentrate on modifying the system parameters, so that some "optimal" performance is found, they overlook the importance of boundary conditions and their unexpected changes and variations. This point is of fundamental importance. What sense does optimality make when the boundary conditions are beyond our control? In Nature fitness comes from the most advanced form of "robustness", namely from adaptivity. This is what characterizes the most resilient designs, like humans and complex mammals. These systems can react and adapt to very large changes in their environments (boundary conditions). Other systems only have built-in (static) or pre-confectioned robustness. This is a less intelligent form of fitness/survivability, also a kind of overkill. The most brutal form of overkill is certainly deterministic optimization based on with safety factors. What this guarantees is a high degree of fragility and built-in excessive conservatism. Optimal designs are in fact overly optimistic and fragile. Changes in boundary conditions, due to the fact that they are seldom contemplated in the process of optimization, result critical in terms of robustness and survivability. One way, though, to overcome this problem would be to formulate many optimization problems, each one with a different

set of boundary conditions, and solve them all, in an attempt to reach some reasonable compromise. But then, there exist more elegant techniques to reach this compromise (read robustness, or fitness) and one such technique is stochastic simulation. With simulation we can move to higher, more advanced forms of design, namely direct uncertainty management and design improvement, the latter being a sort of evolution. Simulation, in fact, enables to obtain designs that are not optimal, but fit. Robust designs versus fragile designs. Optimization is actually just the opposite of robustness.

In contemporary deterministic design, what the engineer is after are repeatable details. This necessity is justified by the need to reach predictability (again, something impossible in the face of uncertainty). Nature looks at its problems a bit differently. What is of concern are patterns and structures, and not numbers. Examples of patterns are illustrated in Figures 1 and 2, where the axes represent performance quantities (displacements, frequencies). These entities, called also response clouds (in opposition to response surfaces) or fitness landscapes, may be obtained via Monte Carlo simulation even for applications as expensive to run as automotive crash. Abstract creations like smooth and differentiable surfaces and curves don't exist in reality, and serve the purpose of keeping most of our modern numerical methods and computer algebra busy. While the study of curves and surfaces leads to data and information, the understanding of response clouds, their patterns and complex structures, is a source of knowledge and insight. Examining Figures 1 and 2, one realizes that being able to explain such properties as clustering (bifurcations), non-symmetry, anisotropy, attractor sets, outliers, etc. is equivalent to the understanding of the underlying physics. Clearly, it is more important to understand the structure and dominant features of response clouds than imposing some abstract model on top of them, like a second order polynomial surface, and, at the same time, destroying the information they contain.

This obsession of top-down choreography pervades many branches of science. A good example is Artificial Intelligence. In the past, people tried to build-in artificial intelligence into computer programs in a top-down fashion, thinking that it was only a matter of getting enough flops. We all know how this approached has failed. Intelligence cannot be imposed, it has to be acquired, it has to emerge naturally. This

logic applies also to CAE. If we impose on a group of points a second order response surface, we are already defining what type of physics will the model be able to yield (unwrap). A simple example of unwrapping is the following. Imagine we write a computer program that simulates the solar system. If all we code is Newton's laws of dynamics and gravity, we will find Kepler's laws from using the model. We will have learnt something new. If, on the other hand, we pre-confection elliptical orbits into the code, Kepler's laws will not stem from our computer program. The important conclusion that must be drawn from this simple example is that computer codes should be based on first principles, and not on complex built-in models or behavioral mechanisms. Just recall how much has been found from the innocent set of Lorenz equations.

It is often argued that when many parameters of a system are changed in a random and independent fashion, it is impossible to separate any parameter influence and impact on the performance descriptors. The fact is that this is the true face of reality. Complex systems should be designed considering collective changes of all of the variables because this is what happens in reality. In systems of hundreds and thousands of variables it is very difficult, not to say practically impossible, to see single dominant design variables. Only in the abstract world of response surfaces is this possible, where even sensitivities can be computed. The whole point is not to look at design variables, but at design features. I have often come across publications (from the DOE community) in which it is stated that Monte Carlo techniques have the scope of desperately scrambling variables in an attempt to find these sensitivities. This is not the case. Here lies the whole point. We are not after one-to-one relationships, we're looking for global patterns, structure and dominant features (clusters, bifurcations, etc.). Actually, once a stochastic look at reality is taken, design improvement rather than optimization becomes the obvious design paradigm. Improvement has the target of positioning oneself on the fitness landscape in a point where performance is good enough or acceptable. In complex fitness landscapes, with hundreds of local minima, there is no such thing as an optimal location. Essentially, instead of trying to hit a bullet with a bullet, we prefer to hit a truck with a shotgun. One way to accomplish design improvement (see Figure 3) is to change collectively the design variables, in particular their mean values, so that the system globally behaves in a different and acceptable

manner. It is then the control over the standard deviations of the components that establishes the quality and robustness of our design. This is exactly what stochastic simulation-based improvement is about: steering collectively design variables so as to achieve acceptable, not optimal, performance.

How does improvement work in practice? The method is very simple. A set of N random samples is generated around the nominal design. A target location in the performance space is defined and the Euclidean distance of each sample to the target is computed. The best one is chosen as the starting point for the next scramble of N points. It may be shown that with $N=16$, the probability of improving the design (i.e. moving towards the target performance) is approximately 0.999986. Figure 3 illustrates the concept. Improvement of this type may be seen as sort of evolutionary process, where the best design survives and is chosen as the seed for a new generation.

One important problem that is overlooked in optimization practice is that of dependent variables. It happens very often that the objective function to be minimized is a function of performance variables, such as mass, frequency, displacement, stress, etc. Clearly, in many cases, these variables are correlated, i.e. are dependent. Since these dependencies are impossible to formulate in a closed form, they cannot be eliminated. One consequence of this is the over-specification (a form of over-constraining) of the problem. This makes the optimization process a lot more difficult and awkward since it involves more variables than are really necessary. In fact, in stochastic improvement, since the dependencies are naturally computed as a by-product of Monte Carlo simulation, only the independent variables are steered. This greatly simplifies the problem and reduces its dimension to the natural minimum.

Some will surely argue that optimization does work. Well, due to the complexity of multi-dimensional systems and their fantastically intricate fitness landscapes (objective functions) there is a very high chance of finding accidentally a better design. Complexity can be very generous. Then, once such a design is found, no one ever cares to prove that the solution is optimal. In fact, in many cases sub-optimal solutions are found and are erroneously accepted as optimal.

Optimization is the wrong thing to do. Let us see another example of a problem that has a relatively simple solution if formulated in an appropriate manner, and which assumes insane proportions if seen as an optimization of something. There are approximately 100000 gene types in the human chromosomes. This yields 2 to the power 100000 possible combinations (10 to the power 30000). Neglecting some basic differences between races, only one combination leads to the human being! Imagine now how difficult it must be to find this combination in such a huge search domain. Not even cosmological time scales would be sufficient to solve the problem. Nature, however, did not see homo-sapiens as anything exceptionally perfect and approached the problem from the point of view of spontaneous self-organization, or self-emergence. It is in fact known from complexity theory that a similar system will spontaneously settle in 317 attractor states, where 317 is the square root of 100000. The fantastic coincidence is that this is the approximate number of cell types in the human body. Actually, this rule has been confirmed also in other organisms; the number of cell types is the square root of the number of genes. Complexity and self-organization have offered an explanation to ontogeny. Cell types are attractors in an incredibly large design space. If Nature had solved the problem through the optimization of some objective function, and indeed found a solution, life would surely be a miracle.

Simulation (in the Monte Carlo sense) is the last (known) frontier for computers. If we can do simulation, why settle for anything less? Once a simulation has been performed, we have all the information about a system that the underlying model can deliver. The resulting response clouds will, because of physics, and not due to some top-down choreography, arrange into patterns (attractors) and form structures, which can convey impressive amounts of information and knowledge. As we have already seen, the number of these patterns/attractors is generally a minute fraction of the apparent dimension of the problem. For this reason, basic macroscopic information about a system can be obtained with a very small number of Monte Carlo samples, generally in the range of tens. Why then go for Baroque procedures, like the response surface method, or DOE, which add approximations to other approximations and which add complexity to complexity? How many tools for optimization exist? Tens, if not hundreds. So which one do we choose? Normally the one that gives the most pleasing results. DOE, on the other hand,

happily samples multi-dimensional and intricate fitness landscapes without prior knowledge of their characteristics and independently of the underlying physics. Can this possibly capture the dominant features of a problem? Then, in order to complete the final picture, and to insulate the problem from physics, a smooth second order surface is forced to pass through the arbitrarily selected points. Clearly, a second-order multi-dimensional surface can only unwrap very simplistic and naïve types of behavior, and is incapable, for example to reflect bifurcations, clustering, or other interesting phenomena that abound in physics. At the end of this top-down choreography, one has mapped physics onto a numerical flatland where search for extremal points is, in effect, the only logical thing to do. Numerical alchemy.

Simulation is a monument to simplicity, but at the same time it is tremendously powerful. In essence, simulation provides a means to generate and explore fitness landscapes. Once we can appreciate how much subtle complexity can even small systems hide, one is inevitably led to ask the following question: do we always need a model? Why not leave the fitness landscape as it is and explore its complexity, instead of destroying it with some numerical and physics-less recipe? Knowledge is hidden in the structure, the patterns and the complexity of multi-dimensional landscapes and therefore these features must be preserved. From this perspective, the Response Surface method is clearly a destructive technique, which imposes in a top-down fashion what the model will be able to deliver (unwrap). The funny thing is that response surfaces are often used to conduct Monte Carlo studies where thousands of samples are computed (in virtue of the fast turn-around times) in order to enable "accurate" evaluations of durability or probability of failure! These studies are either accurately approximate or approximately accurate.

Science does not have the objective of being predictive, but to help explain and comprehend. We think that we are already beyond that stage and now it's only a matter of optimizing. As already mentioned, it is possible to prove that different theories can fit the same data points. For this reason, different models can be made to appear as correct (e.g. geocentric and heliocentric models can predict the same positions of the planets). This is why it is possible for so many variants of the same theory to flourish and make them appear as correct.

Many of us recall the Guyan reduction, which was used to select the important degrees of freedom prior to computing the eigen-frequencies of an elastic system. When powerful computers and techniques became available, like the Lanczos method, Guyan reduction became obsolete. Today, nobody uses the Guyan reduction technique anymore. With today's powerful computers running DOE, in combination with the response surface method and with some exotic optimization algorithm, is like returning to the "Guyan approach". Why do we need to stack approximations upon approximations, allowing the "numerics" to progressively occult reality, if we can afford to run a full-scale Monte Carlo simulation? It is surprising, that people assemble very complex procedures, never validate them versus experimental data (on stochastic grounds, of course), and then speak of accuracy and precise solutions. In similar systems, it is difficult to separate the influence and impact of the model type, the optimization technique, or the starting point, on the final result. Computers should be used as a means of peeling away these layers of approximation, and digging into the physics, not paying tribute to numerical analysis, domain decomposition or message passing. We should seek not numerically optimal but physically significant solutions.

Conclusions

In opposition to techniques like the response surface, DOE, and myriads of optimization algorithms, Monte Carlo simulation engulfs complexity, uncertainty, and robustness in a unified and simple manner. The major steps in science have been possible when generalizations and unifications took place. We now have enough CPU to enable one such step in CAE and HPC. With the possibility of running coarse grain applications like Monte Carlo simulation, CAE can transition from the overly complicated "geocentric" to the simple and elegant "heliocentric" paradigm. CAE and HPC have today enough ammunition to help study the real complexity stemming from physics, not to complicate things by the use of increasingly complex and trendy tools. CAE *can* today reach "escape-velocity".

Systems that are optimal can be terribly sensitive to small changes, especially those in the operational environment. In the classical design practice, not enough importance is attributed to the fact that survivability stems from robustness, not from optimality. Optimality is actually the

opposite of robustness. It is fragility versus resilience. It is much more sensible to settle for solutions that are "only good enough", but robust. Trading optimality for robustness is the mechanism by means of which Nature has been so successful in creating fit designs with a high degree of survivability. The trick is to settle for a design that is good enough because the number of good-enough solutions is incredibly larger than the number of optimal solutions for a given problem.

In the future we should steer computer power towards first investigating fitness landscapes and understanding them, and to subsequently seeking the good-enough solutions. We still have very little knowledge and yet we pretend to jump directly into optimization. It is far more important to get a crude picture of the whole, rather than indulge in sophisticated multi-CPU numerical hairsplitting of some details. Holism instead of reductionism. Computers have fantastic potential to help us learn, not to enforce.

Illustrations

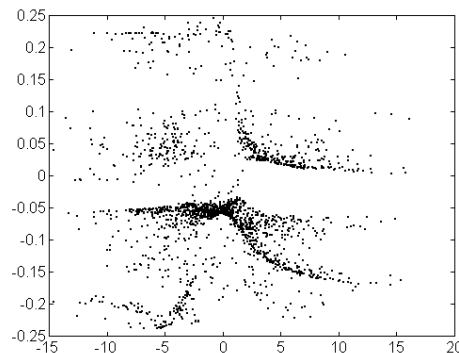


Figure 1. Example of response clouds generated by Monte Carlo Simulation. The axes represent performance quantities, such as frequencies, displacements, or design variables, such as thickness, or material property, or combinations of both. Biologists call such patterns fitness landscapes.

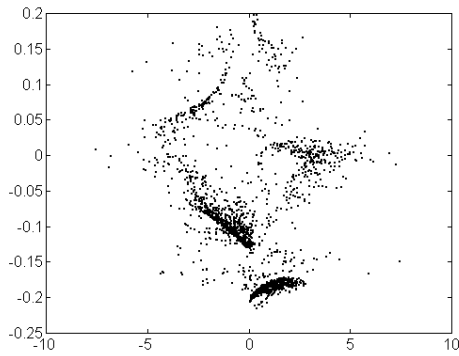


Figure 2. Fitness landscape showing clustering phenomena. It is clearly visible that small variations of the variable on the horizontal axis around zero, can be associated with large changes in the other variable, where the system can jump from one cluster to another with considerable probability. Response surface techniques are unable to even approximate similar patterns.

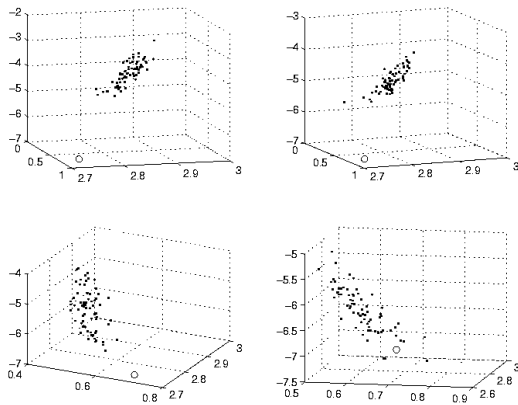


Figure 3. Stochastic design improvement. The initial model (response cloud) is shifted, using a simple return mapping technique, towards the desired target performance represented by the circle. The response clouds shown here are local sub-sets of the landscapes shown in Figures 1 and 2. Improvement has the objective of positioning the response cloud in a desired, or good-enough, portion of the fitness landscape. The process takes place in the N-dimensional performance space of the problem. Its cost is independent of N.

Bibliography

1. Kuhn, T.S., The Structure Of Scientific Revolutions, The University of Chicago Press, 1996.

2. Marczyk, J., ed., Computational Stochastic Mechanics in a Meta-Computing Perspective, International Center for Numerical Methods in Engineering (CIMNE), Barcelona, December, 1997.

3. Marczyk, J., Principles of Simulation-Based CAE, FIM Publications, Madrid, 1999.

4. Waldrop, M., Complexity, Simon and Schuster, Inc., 1992.

5. Briggs, J. and Peat, F.D., Seven Life Lessons Of Chaos, Harper Collins Publishers, New York, 1999.

6. Holland, J.H., Hidden Order, Perseus Books, 1995.