Nonlinear Programming Applied to Thermal and Fluid Design Optimization

Brent Cullimore
C&R Technologies
Littleton, Colorado
303-971-0292

Abstract

Historically, thermal/fluid modeling began as a means of validating and sometimes correcting passively cooled designs that had been proposed by nonspecialists in heat transfer and fluid flow. As dissipation fluxes have risen, and as air cooling reaches the limits of its usefulness, involvement of thermal engineers is required earlier in the design process. Thermal engineers are now commonly responsible for sizing and selecting active cooling components such as fans and heat sinks, and increasingly single and two-phase coolant loops.

Meanwhile, heat transfer and fluid flow design analysis software has matured, growing both in ease of use and in phenomenological modeling prowess. Unfortunately, most software retains a focus on point-design simulations and needs to do a better job of helping thermal engineers not only evaluate designs, but also investigate alternatives and even automate the search for optimal designs.

This paper shows how readily available nonlinear programming (NLP) techniques can be successfully applied to automating design synthesis activities, allowing the thermal engineer to approach the problem from a higher level of automation. This paper briefly introduces NLP concepts, and then demonstrates their application both to a simplified fin (extended surface) as well as a more realistic case: a finned heat sink.

Keywords: Optimization, thermal analysis, parametric modeling, design automation, design synthesis.

Beyond Point Design Simulation

Given input powers, environments, and thermal resistances, temperature responses can be calculated. Given fan curves and filter flow resistances, pressures and flowrates can be calculated.

The above solution sequences represent what is convenient to solve numerically, but rarely do those sequences directly answer the design questions of interest to thermal engineers. Unfortunately, such narrow "point design simulation" formulations are all that is available in most thermal/fluid *analysis* software. What is needed is *design* software.

For example, most available software allows an engineer to build a detailed model of a single specific design, then ask simple questions such as "How hot does this get under this steady-state condition, or during this transient event profile?" When the answer is "too hot," the engineer must try another design, often spending considerable time developing a new model before being able to reevaluate the new design.

It has been so many years since computer-aided analysis solutions have been available, and so many new engineers have joined the community during that time that many have perhaps become accustomed to what software *can* do instead of what it *should* do: help produce a design using point-design evaluations as mere subtasks of this larger purpose rather than as an end to themselves. The purpose of this paper is to show how this ideal is achievable not with artificial intelligence and not by abandoning all the tools and capabilities that exist, but by adding a higher-level design "search engine." The resulting design synthesis environment allows the engineer to ask more powerful questions such as "What is the minimum size fan I should use, and where should I locate the power conditioning unit such that temperatures of the processor do not exceed its limits under three diverse usage/environment scenarios?"

Nonlinear Programming: Generalized Tasking

Nonlinear programming (NLP) techniques attempt to find the maxima and minima of an *objective function* in N dimensions, while obeying arbitrarily complex constraints. Many algorithms exist for solving such problems (Ref 1-9), as do several off-the-shelf software packages. For example, the Solver module in Microsoft's Excel® spreadsheet software is representative of this class of algorithm.

Formal mathematical descriptions of NLP techniques are not necessary to understand their importance to automating thermal/fluid design tasks. Rather, it is important to understand the four parts of a optimization task, as listed below (and as depicted in Figure 1):

1. O*bjective function:* an arbitrarily complex figure of merit to be maximized or minimized.
2. *Design variables:* the parameters whose values at the optimum point need to be determined: the degrees of freedom that can be adjusted to achieve the objective.
3. *Constraints:* arbitrarily complex relationships that distinguish feasible design points (sets of values of design variables) from useless ones.
4. *Evaluation procedure:* the generation of current values of the objective function and the constraint functions given a current set of design values.

While this paper concentrates on the application of NLP technology to design optimization, the math underlying NLP can be applied to a wide variety of tasks: it is a generalized means of defining a complex task or search.

For example, References 10 and 11 document how NLP technology (and other statistical design techniques) can be applied to automatically calibrating thermal/fluid models to test data. Reference 12 describes how NLP techniques were applied to the generation of compact models.

As another example, NLP algorithms can be applied to generating worst-case design scenarios. In this case:

1. The objective function is the temperature of some component to be maximized ("find the hot case") or minimized ("find the cold case").
2. The design variables are the uncertainties. Especially common are environmental uncertainties such as ambient temperature, humidity, pressure (or altitude for avionics applications), and orientation (for aircraft and spacecraft applications).
3. The evaluation procedure might consist of a steady-state thermal/fluid analysis or a transient scenario that yields the temperature of critical components, given specific values of uncertainties ("design variables") as inputs.

Note that constraints are optional, and are absent in the above example.

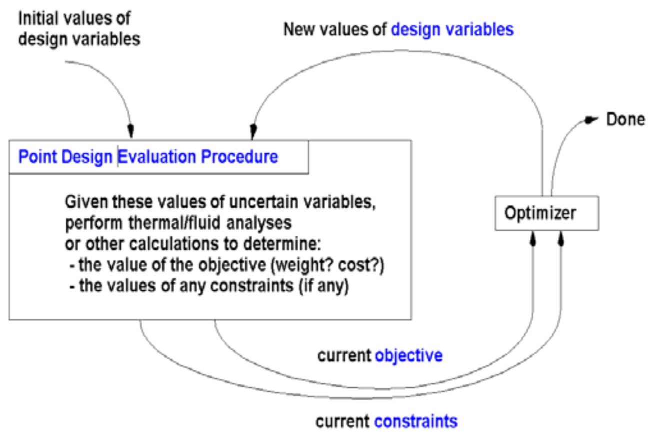

Figure 1: Four Concepts in Optimization

Applying NLP to Design Synthesis

NLP technology is easily adapted to automated design synthesis, and several packages exist that are specifically intended for application to optimization of engineering designs.

As applied to design synthesis, the four components of NLP are as follows:

1. The *objective function* is a figure of merit that makes one design better than another. It might be minimum mass or cost, or maximum performance. It might even be a weighted combination of several factors. For any particular trial design, however, it will have a singular (scalar) value.
2. The *design variables* are the degrees of freedom allocated to exploring the design space: those factors that are allowed to change to try to improve the objective function: dimensions, properties, thermo-static set points, fan speeds, PID controller settings, heater power, etc. Each trial design is identified by specific values of each design variable: a single design vector in N-dimensional design space, where N is the number of design variables.
3. *Constraints* are those rules or threshholds that distinguish a viable or feasible design from a useless one. These might be arbitrarily complex limits on either the design variables or on the performance metrics of the candidate design. For example, "reject a design that exceeds 115°C junction temperature" or "only accept a fan power input of 20W or less." As these examples show, constraint functions are often expressed by inequalities. Note that constraints are optional, but that most realistic engineering problems are heavily constrained. In fact, it is common to initially forget constraints and to have to add more constraints to yield a useful answer.
4. The *evaluation procedure* might consist of a steady-state thermal/fluid analysis or a transient scenario that yields the temperature of critical components, given specific values of design variables as inputs. Specifically, the evaluation procedure consists of any calculations that yield the current values of the objective function and any constraint functions given trial values of design variables. Sometimes these calculations are trivial, and sometimes they require several steady-state or transient solutions of a complex model using a thermal/fluid analyzer.

Two examples are provided to help illustrate these concepts.

Example: Sizing a Fin (Extended Surface)

To apply optimization theory to a specific and simple thermal problem, consider a two-sided rectangular aluminum (165 W/m-K conductivity) fin with a constant root temperature (100°C) in a combined convection and radiation environment (to a constant ambient temperature of 20°C). The emissivity of the fin is 0.2 and the convective environment is assumed constant at 10 W/m^2-K.

The design question to be asked is: "What is the minimum weight fin that can reject 25W, assuming the width (W) is equal to 10cm?"

The four parts of the problem are itemized as follows:

1. The objective function (O) is simply the mass or equivalently volume of the fin: the length (L) times the width (W) times the thickness (T) as depicted in Figure 2. In other words, O = L*W*T or even O = L*T since W is constant.
2. The design variables are the two variable dimensions of the fin: L and T.
3. A single constraint is applied: the power flowing from the root must be at least 25W.
4. The evaluation function might be supplied by a closed form solution, but assuming the engineer had been out of school for a few years and had access to a thermal analyzer, a finite element or finite difference model could be quickly built. A parametric 1D finite difference SINDA/FLUINT model (Ref 13) was used to generate the results discussed below.

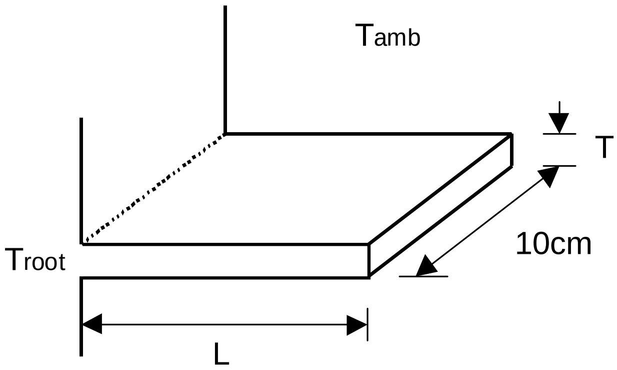

Figure 2: Geometry for Sample Fin

In about 30 to 40 evaluations of candidate designs (the exact number is sensitive to the initial conditions), using NLP algorithms derived from Reference 8, a result is found: L=21cm, and T=3.3mm. As might be expected intuitively, the constraint of 25W is active: the final design rejects exactly 25W. (In a more realistic case with thousands of constraints, only a few will be "active:" influencing the final answer.)

Once a parametric model has been built, many alternate questions could be posed. For example: "What is the minimum mass fin that rejects 25W with a root temperature of no more than 100°C?" This yields the same answer as the previous question, but mathematically it is a different optimization problem. Other completely different problems include:

1. What is the minimum mass fin that has a fin effectiveness of at least 85%?
2. What is the smallest volume fin varying the width and length but keeping the thickness constant?
3. What is the maximum ambient temperature that can be withstood without the root exceeding 125°C, nor the width exceeding 25cm, nor the volume of the fin exceeding 50cc?

Optimizations could themselves be a subproblem of an even larger consideration. For example: "Plot the optimum lengths and thickesses for a range of widths from 5cm to 25cm."

Uncertainties and tolerancing can also be included when evaluating optimum designs (Ref 13). For example, consider that the emissivity and ambient temperature are not deterministic but are instead given by probability distributions or other tolerancing. In this case, a reliability estimation (based on statistical methods, Ref 10) can be embedded in the evaluation procedure for candidate designs. One might then find the minimum weight fin that has 99% chance of success (i.e., reliability), or one might estimate the allowable tolerance on emissivity and ambient temperature.

Example: Designing a Ducted Heat Sink with Fan

To illustrate the utility of automated design synthesis on a realistic application, consider a component dissipating 25W that is located on a 70mm by 100mm by 3mm copper plate. The plate is bonded to 25mm tall aluminum fins, forming a ducted (enclosed) heat sink cooled by air from a fan.

In the hot case ambient environment (38°C), the initial design is not quite able to keep the component below its maximum temperature of 60°C with an inlet air velocity of 2 m/s. The initial design and the temperature gradients are displayed as Figure 3.

Optimization was used to find a better design that met the same performance objectives and that occupied the same 70x100mm footprint. Three "design variables" were chosen: (1) the thickness of the copper baseplate, (2) the height of the aluminum fins, and (3) the inlet air velocity provided by the fan. Reasonable limits were placed on the possible range of variation of these variables (e.g., 0.1 m/s to 4 m/s for the air velocity). While the number of fins could have been varied along with their thickness, for simplicity these variations were not explored. Instead, the number of fins was kept constant and the thickness of the fin was made proportional to its height as a structural integrity constraint.

In addition to "side constraints" on the possible range of variation of the three design variables, a "performance constraint" was also imposed: the temperature of the dissipative component was specified as not to exceed 60°C. It should be noted that, in the course of exploring the design space, this threshhold is occasionally violated (since the thermal/fluid software cannot know that the trial design is infeasible until it is attempted). All that this constraint guarantees is that the *final* design produced by the optimization algorithm *will* meet this criterion.

While it may be obvious that a design that fails to provide adequate rejection is infeasible, the choice of what consistutes a "better" design is complex and perhaps subjective, and yet it must still be posed as numerical value for each trial design. Such an "objective function" or figure of merit might include quantified considerations of cost, manufacturability, reliability, etc. For this design study, it was decided to find the smallest suitable design.

The selected objective function included not only the structural weight of the fins and baseplate, it also included a penalty function for a large fan to avoid designs that required unrealistically high air velocities in order to satisfy the thermal design requirements. This fan penalty function was based upon the electric power required by the fan: $G^*\Delta P/\eta$, where G is the volumetric flowrate (m^3/s), ΔP is the fin total pressure drop (Pa), and η is the fan efficiency (about 40% in this case). In order to be summed with the structural mass, this power penalty was converted into an effective mass using an estimate of the "mass cost" of the power in terms of batteries, etc. Such complex and subjective manipulations are commonly done in most trade-off studies. Nonetheless, it is important to remember that a reformulation of this objective function would yield a different design.

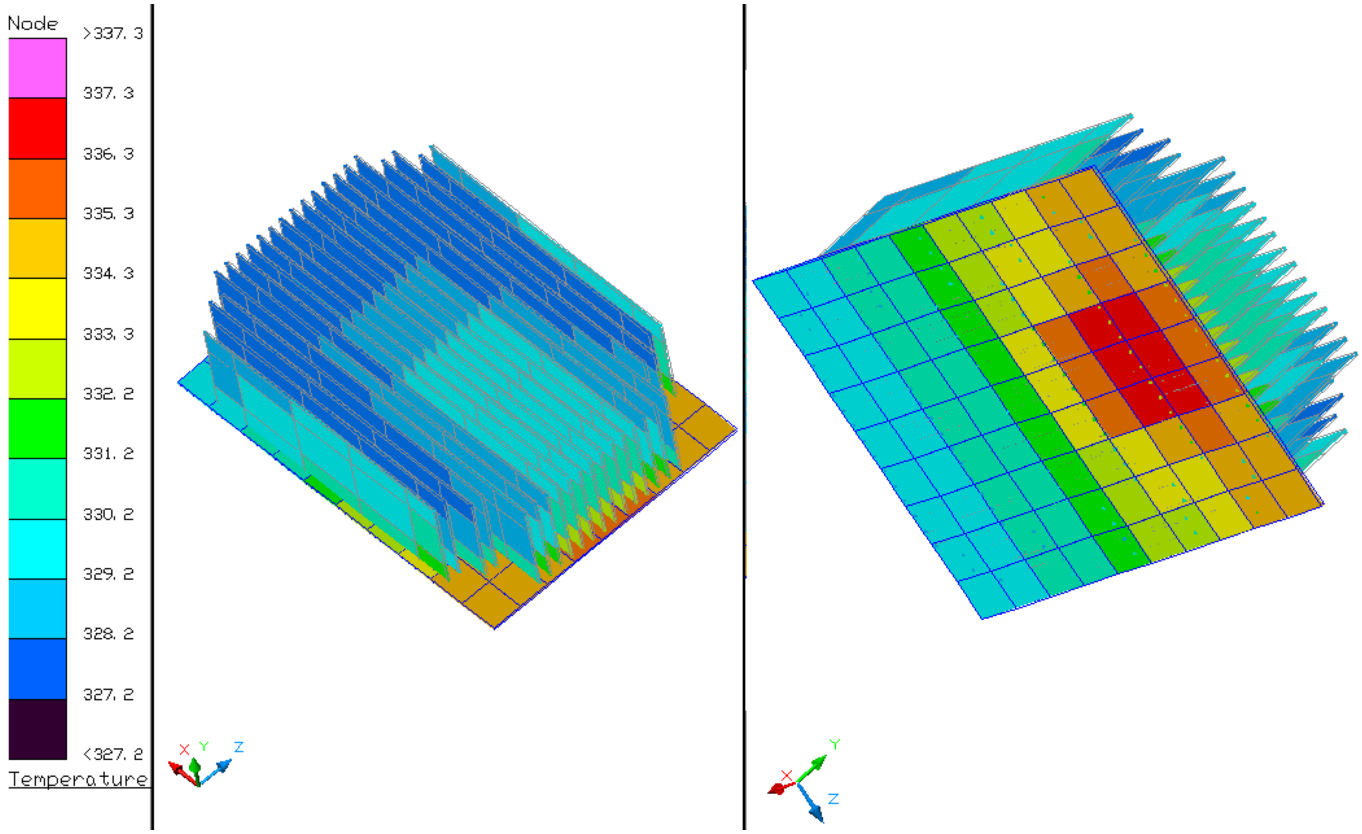

Figure 3: Initial Configuration/Performance of Heat Sink (Cover Removed)

The "evaluation procedure" consists of a single straightforward steady state solution: given the current value of the fin height, the base thickness, and the air velocity, find the temperature of the component, the total pressure drop through the fins, and the volumetric flowrate through the fins. The current mass of the fins and plate are also calculated, although this calculation does not require a thermal/fluid solution. The temperature of the component is used to compare against the sole constraint (60°C), whereas the pressure drop, volumetric flowrate, and structural mass are used to calculate the current value of the objective function.

This problem was posed using Thermal Desktop® and FloCAD® (Ref 15, 16), with SINDA/FLUINT providing both the solution engine and the built-in NLP module. Again using the sequential linear programming (SLP) algorithm (Ref 8), an optimum design was found after automatically exploring 72 trail designs.[1] The total wall clock time was on the order of 5 to 10 minutes on a 1 GHz PC.

The initial and final design are summarized in Table 1, and the final design is depicted in Figure 4 (note that the thickness of the baseplate does not appear to change since it is a 2D CAD object: its thickness varies in the thermal model but not in the drawing).

[1] The actual number of evaluations required can vary depending on the initial conditions, the number of design variables, the NLP algorithm employed, the amount of computational noise present in the underlying solution, etc. Typical values range from 30 to 300. Given slight variations in this problem, the range was about 40 to 100.

As shown in Table 1, the NLP solver was able to reduce the objective function by 10%, resulting in a lighter design. The final design faithfully met its temperature performance obligations of 60°C, whereas the initial design was slightly over the limit at 64°C. Mostly, the baseplate thickness was reduced in exchange for a higher fan speeds, although the aluminum fins were also enlarged somewhat. A heavier penalty on the fan power would have resulted in a lowering of the optimal air velocity and a thickening of the baseplate.

Challenges for the Engineer

Automated searches for optimum designs do not replace an intelligent and experienced engineer. Rather, they shift his or her responsibilities. Enabling a problem to be attacked at a higher level than "how hot does this get?" empowers engineers, but does not absolve them of responsibility for the model nor for the resulting design.

The most difficult part of using automated design synthesis technology is conceptual: posing the problem efficiently. Most engineers are not trained in formal optimization techniques and have not had access to software that lets them approach a problem at a higher level than point design simulation. Objective functions and constraints are often confused: there is a tendency to mathematically add constraints into the singular objective function … to define a complex figure of merit or composite objective with multiple goals, rather than isolating some "desires" as constraints. There is also some confusion created since design variables are *inputs* to the low-level evaluation procedure but *outputs* from the top-level optimization run. As more and more organizations emphasize design automation and are able to exploit software containing these techniques, and if optimization is taught more at the university level, these problems will eventually disappear.

Another problem is that the underlying thermal/fluid model must be built to be run repeatedly while exploring a wide range of possible designs. Furthermore, the model must be accurate enough to avoid confusing the NLP solver with false trends or conflicting information generated from "computation noise." In otherwords, additional emphasis is placed on generating a model that is robust, fast, and accurate … considerations that are often at odds with each other. Alternatives to CFD codes may be required (Ref 16) to enable such automated design explorations.

Finally, the engineer must verify the solution. Multiple solutions are not common because most realistic problems never reach a true optimum. Instead, they are heavily constrained. Nonetheless, the engineer must make sure that the resulting design is sensical and that no other additional constraints need be applied in hindsight. Unfortunately, NLP solvers are prone to stopping prematurely because "lack of progress" is mathematically equivalent to "this is as good as it gets:" both signal problem completion. NLP engines also are sensitive to scaling problems: why bother discerning the difference between 1.0cm and 1.1cm when the initial value of the design variable was 100cm? The easiest way to verify solutions is to rerun the problem using different (but reasonable!) initial values of the design variables, and see if either the same design resulted or if an equally good or better design is found.

Table 1: Summary of Initial (Manual) and Final (Automatically Synthesized) Designs

	Fin Thickness (mm)	Baseplate Thickness (mm)	Air Speed (m/s)	Component Temperature (C)	"Mass" (gm)
Initial	25	3	2	64	264
Final	33.4	1.42	3.69	60	241

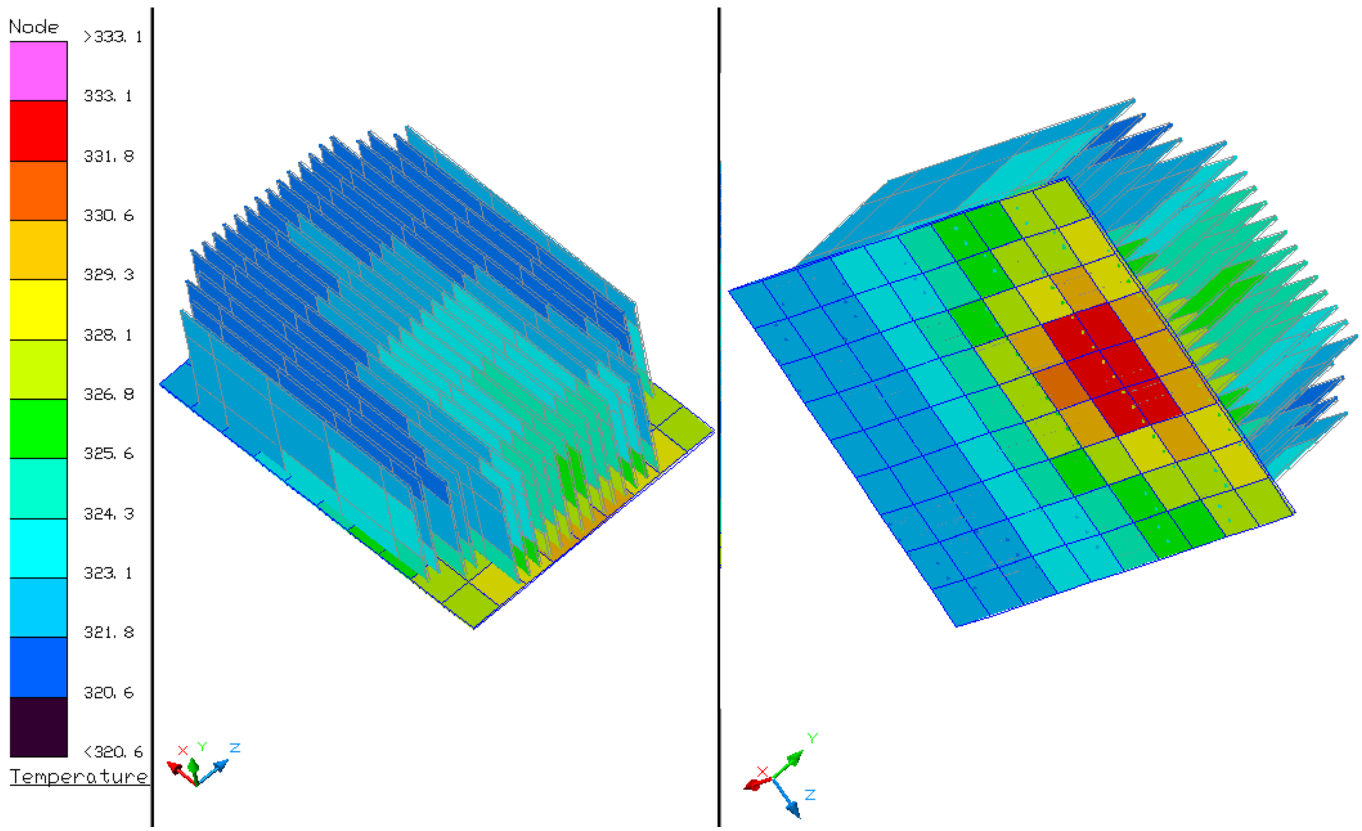

Figure 4: Automatically Optimized Heat Sink (Cover Removed)

Challenges for Analysis Software

How does a thermal design engineer exploit the availability of these advanced techniques using their favorite thermal/fluid analyzer? Optimization techniques involve a higher level of analysis beyond a traditional "point design simulation." Most engineering analysis software is set up to solve a deterministic set of equations, either steady-state or transient, given a fixed set of inputs. In other words, these programs provide predictions of how a single point design performs under specific environments. Automated model calibration, on the other hand, requires either using or creating a software tool that can perform multiple iterative point design evaluations. This section describes three approaches toward achieving such a capability.

The first option uses an in-house development approach. To start with, engineers can write their own optimization engine or purchase one commercially. Next, a means of executing the thermal/fluid analyzer iteratively must be achieved, perhaps via an API (application programmer interface) if available, or perhaps simply by modifying and rewriting text input files and reading text output files. A script can be generated to iteratively run the thermal/fluid analyzer, driving the design variables with the optimization engine such that a optimal design is achieved. This option is cost effective only if software development labor is inexpensive or if an organization is large enough to recoup the investment of the development of a general-purpose utility. Otherwise, considerable effort will be spent rewriting the software every time a new optimization task arises.

As the second option, engineers can acquire a general purpose MDO (multidisciplinary optimization) environment. Examples of such software include Engineous' iSIGHT®, Phoenix Integration's ModelCenter®, MSC Software's RDCS, Synapse' Pointer®, VR&D's VisualDOC®, LMS' Optimus®, and Samtech's BossQuattro. To varying degrees, these programs enable the engineer to set up their favorite thermal/fluid simulation code as part of the evaluation of a candidate design. The advantages are that the thermal/fluid simulation codes need not "know" that they are being used in such an iterative fashion: little to no modifications of the simulation codes and models are required. A very significant benefit of this MDO approach is that it allows integration of diverse programs and models to include cost/risk assessments as well as other specialties such as structures and power management. Also, this approach also has the advantage of providing an infrastructure that reduces the time to create a new optimization task. However, disadvantages of the MDO approach include the often considerable cost of acquiring and learning such codes (both are on par with CFD codes), and the relatively slow speeds resulting from inefficiencies in running the simulation code in such a disconnected fashion. Nonetheless, such an approach is clearly better than the current "manual" methods of evaluating design alternatives.

A third choice is to use a thermal/fluid analyzer that already has these advanced features built-in (Ref 4). This avoids the overhead associated with the first choice, and the additional costs associated with the second choice, and is much faster to execute than either of those choices for various reasons.[2] However, choices are limited for two reasons. First and most important, few thermal/fluid analysts are aware of such capabilities, and hence they more typically demand additional detailed phenomenological modeling power rather than more help with design and calibration tasks. Forgivably, commercial vendors listen to them, and the demand for high-level decision support tools is therefore slack. Second, even after analysts discover these gains in productivity and begin to demand them, software suppliers will find it difficult to accommodate these requests without significant changes in their software. To accommodate high-level analyses such as design optimization and reliability estimation, the software must first become fully parametric instead of expecting single-valued ("hard-wired") design and environment specifications. There is hope, however: structural analysis and CAD software have increasingly emphasized such capabilities in their new releases over the last five years. It is hoped that thermal/fluid analysis tools can follow these examples and catch up once the user community has been educated and the demand for new capabilities is established.

On-Going Developments: Integer/Discrete Variables

Most NLP solvers are based on gradient ascent/descent methods that struggle with discontinuities. One important class of discontinuity is the "selection problem:" one or more design variables whose values are either integers or can only assume discrete values. Examples include sheet metal sizes, pipe sizes, and fans. In other words, most off-the-shelf products may only be purchased in specific (discrete) sizes. In the above heat sink example, the number of fins is an integer.[3]

The current method for overcoming such difficulties is to use a real (continuous) variable, then to round the answer up or down to the nearest available size. If there are multiple integer/discrete variables, then the optimization should ideally be rerun after fixing each variable in succession: the so-called "branch and bound" strategy.

Eventually, however, improved algorithms must be developed. Current candidates include Synthetic Annealing (SA) and Genetic Algorithms (GA). GA, for example, *requires* integer/discrete variables: real variables are actually approximated by using fine resolutions of discrete variables. Unfortunately, these methods currently require an excessive number of design evaluations and so are not realistically useful for many engineering problems. Fortunately, such algorithms represent an active area of research, so the current situation is expected to improve in the next decade. In the meantime, rounding up or down is usually adequate for most

[2] In addition to avoiding interprocess communication and overhead associated with starting and restarting programs, a built-in capability can exploit the advantage that previous steady state solutions (which usually comprise the majority of calibration and reliability assessment tasks) in the search were close to the current solution, and can jump quickly to incremental answers.

[3] In that example, however, there are ways in Thermal Desktop to vary the fin spacing as a real (continuous) variable without varying the actual number of fins: extra fins may be generated such that all fins expand or contract in accordion fashion. Any fins that extend past the width of the baseplate lose their connection to it, and are therefore of no thermal importance.

problems, and certainly represents an improvement over manually iterated designs.

Conclusions

This paper has shown how the existing analytical emphasis on point design evaluation (e.g., “Here’s a model of a component. How hot does it get under these circumstances?”) is based on what existing thermal/fluid software *can* do, instead of what it *should* do: help automate higher-level engineering tasks such as design synthesis. Existing software can be automatically rerun using NLP technology tasked with seeking an improved design. In software designed to include these capabilities as “native,” application of automated design optimization is becoming more and more common as engineers gain experience.

The resulting techniques still require a good model and an experienced engineer who is making sound decisions. However, a significant improvement in productivity has been demonstrated using these technologies in actual commercial applications. These methods are therefore expected to be increasingly common over the next decade.

References

1. Broyden, C.G., “The Convergence of a Class of Double Rank Minimization Algorithms, Parts I and II,” J. Inst. Math. Appl., Vol 6, pp. 76-90,222-231, 1970.
2. Fletcher, R., “ A New Approach to Variable Metric Algorithms,” Computer J., Vol. 13, pp. 317-322, 1970.
3. Goldfarb, D., “A Family of Variable Metric Methods Derived by Variational Means,” Math. Comput., Vol 24, pp 23-26, 1970.
4. Shanno, D.F., “Conditioning of Quasi-Newton Methods for Functional Minimization,” Math. Comput., Vol 24, pp 647-656, 1970.
5. Fletcher R. and Reeves, C.M, “Funtional Minimization by Conjugate Gradients,” Br. Computer J., Vol. 7, No. 2, pp. 149-154, 1964.
6. Zoutendijk, K.G, Methods of Feasible Directions, Elsevier Publishing Co., Amsterdam, Netherlands, 1960.
7. Vanderplaats, G.N., “An Efficient Feasible Direction Algorithm for Design Synthesis,” AIAA Journal, Vol. 22, No. 11, Nov. 1984.
8. Kelly, J.E., “The Cutting Plane Method for Solving Convex Programs,” J. SIAM, Vol. 8, pp. 702-712, 1960.
9. Powell, M.J.D., “Algorithms for Nonlinear Constraints that use Lagrangian Functions,” Math. Prog., Vol. 14, No. 2, pp. 224-248, 1978.
10. Cullimore, B., "Dealing with Uncertainties and Variations in Thermal Design", IPACK2001-15516, InterPack ’01 Proceedings, July 2001.
11. Baumann, J. et al, “Nonlinear Programming Applied to Calibrating Thermal and Fluid Models to Test Data,” SEMI-THERM, March 2002.
12. Lasance, C.J. et al, “Creation and Evolution of Compact Models for Thermal Characterization using Dedicated Optimization Software,” SEMI-THERM XV proceedings, March 1999.
13. SINDA/FLUINT User’s Manual, PDF available at www.crtech.com.
14. Cullimore, B., “Optimization, Data Correlation, and Parametric Analysis Features in SINDA/FLUINT Version 4.0,” SAE 981574, International Conference on Environmental Systems, July 1998.
15. Thermal Desktop, RadCAD, FloCAD User’s Manual, PDF available at www.crtech.com.
16. D. Johnson et al, “Novel Simulation Techniques for Design of Air-cooled Electronics,” IPACK2001-15523, July 2001.