Paper 171-2007

Optimization with SAS/OR®: What It Is, What's New, and How It Adds Value

Ed Hughes and Trevor Kearney, SAS Institute Inc., Cary, NC

ABSTRACT

SAS/OR® 9.1.3, Release 3.1 continues the initial release of a completely new suite of powerful, accessible optimization procedures. This roster of new procedures includes production versions of OPTMODEL for powerful and accessible building and solution of optimization models, OPTLP for linear optimization, and OPTQP for nonlinear optimization. New experimental features include the OPTMILP procedure for mixed-integer optimization and mixed-integer capabilities for the OPTMODEL procedure.

This paper provides background information about the role of optimization and surveys the new SAS/OR optimization advances, highlighting their importance for organizations facing larger and more diverse problems in resource allocation, supply chain planning, and many other domains in which optimization plays a key role.

INTRODUCTION

SAS® software includes a diverse and well-established array of optimization capabilities designed to assist in building and solving many different types of optimization models. The greatest concentration of modeling, analysis, and problem-solving capabilities is found in SAS/OR, but some optimization features are also present in SAS/STAT®, SAS/IML®, and SAS® Enterprise Miner®; SAS/ETS® also provides considerable modeling capabilities in PROC MODEL. Users at a wide range of major U.S. and international corporations have found these capabilities to be indispensable in building decision-support and decision-guiding solutions aimed at identifying the best and most profitable strategies and action plans for their enterprises.

SAS/OR software includes a completely new generation of optimization modules (procedures) that make it easier to tackle a broad range of optimization problems. The foremost of these is the OPTMODEL procedure, incorporating a powerful symbolic, interactive modeling language that enables you to work through the iterative process of building, solving, and refining optimization models with clarity and precision. New procedures for solving specific types of optimization problems (specified using input data sets) are also provided. Supporting these procedures is a new suite of optimization solvers, using state-of-the-art algorithms and delivering great improvements in performance and scalability.

Complementing this optimization technology are several current and upcoming SAS solutions that include optimization as either a core function or a critical enabling technology, leveraging the decision-guiding power and insight added by SAS optimization. Examples include SAS Marketing Optimization, SAS Revenue Optimization, SAS Inventory Optimization, SAS Service Parts Optimization, and SAS Credit Risk Management.

This paper describes the nature and purpose of optimization and considers the various types of optimization problems that can be solved with SAS/OR software. We explore the value that optimization adds to enterprise data, to a range of analytical technologies, and to business intelligence. We look at the unique advantages that SAS delivers to optimization modelers. We consider the new optimization features available with SAS/OR and see how they make the building and solving of optimization models with SAS easier, more transparent, and more scalable than ever before.

WHAT IS OPTIMIZATION? HOW ARE OPTIMIZATION MODELS BUILT?

Simply put, optimization is the process of choosing the permissible actions that result in the best outcome. No matter how complex the means by which optimization is implemented, this is the underlying concept. On these terms, virtually all activity—whether of enterprises, groups, individuals, or microorganisms—can be viewed as some form of optimization. Whether the goal is survival of the species through effective propagation, maximization of profits via production decisions, minimization of risk with purchasing strategies, or something as mundane as finding the shortest route from one place to another, we are all striving to optimize something — or, more likely, several things at once.

Of course, not all attempts at optimization succeed; often success occurs in widely varying degrees of completeness.

A number of factors can influence the degree to which optimization succeeds. On the most basic level, we might not have the freedom to take the actions that produce the best results. We might have incomplete or incorrect information about how various actions interact with each other, how they are limited by our circumstances, or how they influence the outcomes. There may be so many choices of actions that it is impossible to evaluate them all. The outcome itself might be inappropriately or inaccurately measured. We might not even have a reliable way to determine if a particular outcome is remotely close to being the best outcome.

These are the kinds of difficulties that plague informal, intuitive, unstructured attempts at optimization. What is needed in most cases is a more precise description of the decision problem and a more disciplined approach to optimization. This can mitigate or eliminate many of the problems we have described.

ELEMENTS AND CATEGORIES OF OPTIMIZATION PROBLEMS

A well-defined approach to optimization begins with a rigorous description of the three key elements of any optimization problem:

- **Decision variables,** which are numerical representations of the available actions or choices. Examples include production levels, price settings, and capital or human resource allocations.

- **An objective** that is the goal of the optimization; something to be achieved. This goal must be measurable. Examples include maximizing profit, minimizing distance traveled, and minimizing unused raw materials.

- **Constraints** specifying requirements or rules, placing limits on how the objective can be pursued by limiting the permissible values of the decision variables. Some examples are machine processing capacity per hour, customer demand by sales territory, raw materials availability, bills of material in manufacturing or assembly, and budgetary restrictions.

Each type of optimization model is defined by how its decision variables are defined and by the types of mathematical expressions that are used in defining its objective and constraints:

- **Linear programs (LPs)** use only linear expressions in constraints and the objective (for example, 4x + 3.5y – 2z<=7), and the decision variables can take on any value in specified allowable ranges.
- **Mixed-integer linear programs (MILPs)** are linear programs in which some of the decision variables must have integer values.
- **Integer programs (IPs)** are linear programs in which all of the decision variables must have integer values.
- **Nonlinear programs (NLPs)** use nonlinear expressions (such as x2, cos (y), 1/x, and others) in the constraints and/or the objective.
- **Quadratic programs (QPs)** are a specific type of NLP with a quadratic objective (involving squares of decision variables or the product of two decision variables) and typically linear constraints.

Within this framework of decision variables, objective, and constraints, the purpose of optimization is to maximize or minimize, as appropriate, the performance metric in the objective by assigning values to the decision variables that satisfy the constraints. Solution methods generally are tailored to suit each type of optimization model.

CREATING AN OPTIMIZATION MODEL FROM A BUSINESS PROBLEM

The outlines of the process for creating optimization models are very straightforward. The first step is to build a conceptual model of your decision problem, which means that you need to identify the important factors in the decision-making process (such as the amount of each product to make, the associated costs or profits, the required materials and equipment, etc.) and describe how those factors interact with each other.

Next you describe this conceptual model with more rigor and detail by representing the conceptual model in mathematical terms. This is the start of formally defining the key elements of the optimization model (decision variables, objective, and constraints), and it requires a more detailed, more rigorous understanding of the decision problem than the first conceptual stage. Here the objective is defined as a measurable function of the decision variables, and constraints are defined as equalities or inequalities involving functions of the decision variables. Decision variables represent decisions by ranges of allowable values, each corresponding to a permissible decision.

This is the point at which the artistic aspect of optimization modeling first becomes prominent, since there is no single “right” way to use mathematical expressions to represent the elements of a decision problem. Every model

represents a compromise since no workable model can reflect every minute detail of the decision problem. First, such a model would likely be too large to solve efficiently; and second, its directives would be so detailed that they would amount to "micro-management," likely to be selectively ignored by the people tasked with implementing the model's recommended decisions. No software can determine the most appropriate representation of a decision problem by an optimization model; software is a tool in the hands of optimization modelers whose training and experience guide their choice of the best model.

Once you've built the model and solved it (as described later) to produce an optimal solution, you need to consider whether the optimal solution is suitable for the original business problem. At this point it's not unusual to discover that some key element of the model has been overlooked or misconstrued, making the optimal solution (and the decisions that it represents) unsuitable. The model might be correct, but some data used by the model might be incorrect. Your understanding of the original business problem might be flawed. In these and other such cases you need to step back through the modeling process, address the difficulty, and then move forward with the improved model. This iterative process is quite common and represents another aspect of the "art" of optimization modeling.

THE ROLE OF DATA AND ANALYTICS

The optimization modeling process always involves data; after all, how can you describe the elements of the model mathematically without detailed knowledge of the relationships that you're attempting to describe? You cannot, for example, attach a cost of $1.50 to the production of each unit of product A unless you have data stating that the cost is in fact $1.50.

This means that optimization modeling requires that significant effort be directed toward working with data—gathering descriptive data from across the organization, extracting data from its native format into a common format, cleansing or repairing data that may be incomplete or contradictory, and storing or warehousing it for centralized access. Often, too, statistical analyses need to be performed on the data, data mining must be performed, and forecasts and econometric models must be created in order to compile the descriptive and predictive information that characterizes the important relationships that are reflected in the objective and constraints of the optimization model.

Thus optimization never occurs in a vacuum, isolated from other data and analytic concerns. Rather, optimization is part of a continuum, beginning with simple access to raw data, moving through standard and customized reporting, to descriptive and predictive analytics, and finally to optimization. Each of these stages forms the foundation upon which the next stage is built, and each stage adds value to the data and information received from the preceding stages. This integral relationship between data, analytics, and optimization is depicted in Figure 1.

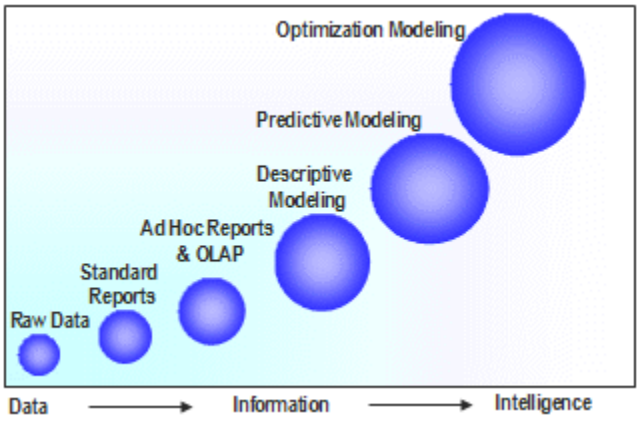

Figure 1. The Data and Analytic Continuum Supporting Optimization

In light of this relationship it makes sense to think of optimization not as distinct from analytical technologies but as one more type of analytics. Building on the decision support of descriptive and predictive analytics, optimization provides directive or prescriptive analytics, supplying proactive decision guidance. It's also worth noting that while it's widely recognized that optimization is linked intrinsically to data and analytic technologies, SAS is the only software company that provides this entire spectrum of data, analytic, and optimization technologies within a single language and under a common data format.

THE BENEFITS OF OPTIMIZATION

DIRECT BENEFITS OF OPTIMIZATION

In an organizational sense, optimization identifies a set of decisions or actions that collectively fulfill all requirements while making the most productive use of resources. In many cases this translates to either saving or earning as much money as possible for the organization.

This is accomplished by using one of a number of optimization algorithms, most of which are essentially structured search methods for the "solution space" that consists of all possible combinations of values for the decision variables. In devising an optimization algorithm, one goal is to make the search method "smart" enough so that relatively little of the overall solution space needs to be examined directly. This enables an optimal set of decisions to be identified much more quickly.

Even for conceptually "small" optimization problems, this structured approach generates enormous savings in time. For example, in a problem in which any of 5 possible production levels may be chosen for each of 8 products, there are over 32,000 possible sets of decisions. Just to evaluate an objective for each alternative and to determine if the alternative satisfies all constraints would take an inordinate amount of time, but a good optimization algorithm may well finish in a fraction of a second.

Optimization can also produce great improvements in the quality and effectiveness of the decisions that are identified. Since manually evaluating all possible alternatives is unworkable, manual decision-making methods necessarily consider only a small fraction of the possibilities, but without the structured approach of mathematical optimization. Usually organizations using manual methods rely on intuition and experience that, while providing useful insights, are limited in their scope. Optimization offers a fresh, unbiased view of the decision problem that often identifies alternative actions that have never before been imagined.

For problems with more decisions, more choices per decision, greater detail, longer time frames, or more constraints, the benefits of optimization vs. manual or intuitive decision-making are even more dramatic. Optimization can help organizations to identify the most productive decisions much more quickly, even for huge, highly complex decision problems.

OPTIMIZATION'S STRATEGIC BENEFITS

In addition to solving the problem as stated above, optimization can produce a great deal of useful auxiliary information, yielding insights into the decision problem and thus delivering benefits on a higher level. This information can help in addressing questions such as these:

- Is what we are trying to accomplish possible?
- What are the most/least restrictive constraints or restrictions that we face?
- Can we improve on our current performance? If so, how?
- What's the best performance that we can achieve?
- How do changing conditions affect our decisions and our performance?

These fundamental questions are relevant across departmental, geographical, and industrial boundaries. This is true simply because addressing such questions effectively is critical to good strategic and tactical planning and can have a significant impact on operational planning as well.

On a higher level, optimization, when employed thoughtfully, can transform the entire decision-making process. For organizations making decisions purely on an intuitive, experience-oriented basis, the decision-making power is intrinsically linked to the individuals in the organization who have amassed the experience and intuition. Too often this knowledge isn't expressed or documented in any formal sense, meaning that without the participation of the key individuals the decision-making process is severely handicapped.

Optimization changes this situation by formalizing the decision-making process and building into optimization models much of the expertise that hitherto belonged only to key people. Optimization also can improve upon intuitive decision-making by adding these critical characteristics:

- **Structure:** Enabling the organization to understand, validate, and document the decision-making criteria that are used in practice.

- **Consistency:** Making certain that various business units, departments, divisions, etc. are all directed at achieving the same or complementary goals.
- **Repeatability:** Consistency along the "time axis," ensuring that the same underlying decision principles are applied over time.
- **Adaptability:** Enabling the model and the decisions that it identifies to react to changing conditions while still adhering to the same decision principles.

In establishing these four attributes, optimization transforms decision-making from an enigmatic art that is the domain of a handful of experts to an open, well-documented process that welcomes participation by a far broader range of stakeholders.

NOTES ON OPTIMIZATION AND BUSINESS INTELLIGENCE

Optimization might be seen as competing with business intelligence since both seek to influence decision-making, but the opposite is true. Optimization and business intelligence can and do work together to overcome their individual weaknesses and magnify their respective strengths.

Business intelligence unquestionably provides great value to organizations by highlighting key relationships, surfacing performance metrics, predicting future performance, identifying opportunities for improvement, and much more. However, it's important to remember that business intelligence, like the descriptive and predictive analytic information that it helps to surface, is limited to better informing the current decision-making process. It has nothing to do with reshaping the decision-making process but instead focuses on making the current process more aware. Unfortunately, if the current process relies too heavily on intuition, time-worn rules, or "business as usual," then the benefits of business intelligence will be limited at best.

Optimization, for its part, assumes that data and information describing critical relationships (between actions and outcomes, and between the actions themselves) will be available. Optimization also includes no direct means for communicating optimal decisions across an organization so that they can be implemented. Something must feed data to optimization and communicate its results; very often that "something" is business intelligence.

Still, optimization's reach far exceeds that of business intelligence because it has the ability to reshape decision-making based upon its structured view of decision problems and its structured approach to identifying better decisions. Optimization takes the awareness delivered by business intelligence to the next level by using it to create awareness of better ways of making decisions.

Therefore the relationship between optimization and business intelligence is not competitive but symbiotic. Each supplies something that the other lacks or needs, and the combination of the two is stronger and more effective than either could be alone. At the enterprise level, structured decision guidance through optimization is the perfect complement to business intelligence.

OPTIMIZATION WITH SAS

SAS/OR: OPERATIONS RESEARCH SOFTWARE

SAS/OR, part of SAS software since 1983, brings together many of the analytical modeling and solution methods that are referred to collectively as "operations research." These techniques share an outlook that advocates understanding the details of a process and using that knowledge to improve decisions and performance. Major areas of operations research work addressed by SAS/OR include the following:

- **Mathematical Optimization:** Support for building and solving a broad range of optimization models.
- **Project and Resource Scheduling:** Critical-path and resource-constrained project scheduling, which determines when to perform individual tasks that are linked by precedence and/or hierarchical relationships. Resource requirements must be satisfied while working within limits on resource availability, and deadlines must be met.
- **Discrete Event Simulation:** Studying the performance of systems such as customer service operations, manufacturing processes, and traffic handling for which critical elements—such as arrival of customers or vehicles, processing or service times, etc.—exhibit random variation. Simulation enables such systems to be studied in a modeling environment in which the long-term effects of alternative configurations and policies can be measured, analyzed statistically, and compared.

Optimization has been a key part of SAS/OR from the beginning. Long-standing optimization procedures in SAS/OR include the following:

- **PROC LP:** Linear, integer, and mixed-integer optimization based on the simplex solution method.
- **PROC INTPOINT:** Linear optimization using an interior-point solution method.
- **PROC NETFLOW:** Network flow optimization for problems involving flows between nodes (locations) along arcs (node-to-node connections). Solves shortest path, minimum-cost flow, and maximum flow problems.
- **PROC NLP:** General nonlinear optimization.

These procedures accept optimization problems specified by the use of input SAS data sets, perform the desired optimization, and report the results in SAS data sets and output files. PROC NLP, due to the special nature of nonlinear optimization problems, also includes some optimization modeling capabilities. For the remainder of these procedures the optimization model is built as the input SAS data set is created, using the SAS DATA step.

RECENT TRENDS IN OPTIMIZATION WITH SAS/OR

In the past several years a number of patterns emerged regarding the use of SAS/OR for optimization. First, optimization was being used much more broadly than ever before. This was due to several causes, one being the general rise in competition among enterprises as consumers became better informed and less committed to single companies or brands. This led to greater price competition, shrinking margins, and a greater need to extract better performance from less time and fewer resources. Thus the user base for SAS/OR's optimization capabilities became much more diverse.

Next, optimization in SAS/OR was being used more frequently in conjunction with other SAS analytics such as statistics, data mining, and forecasting. Data sets produced by these methods were used as input for optimization, meaning that considerable processing had to occur in order to extract the specific information needed by the optimization model and place it correctly in the input data sets for SAS/OR optimization. Within this cross-functional environment SAS/OR was being used to solve larger and more complex optimization problems. This was a natural result of the benefits of optimization, as organizations hungry for better performance sought to expand their use of optimization in scope and scale.

More SAS/OR users were new—new to SAS/OR itself and new to SAS as a whole. This meant that we could no longer assume that these users were experienced with the SAS language and adept at using the DATA step to manipulate data for input to the SAS/OR optimization procedures.

This created several challenges for optimization with SAS/OR. We needed to provide solution methods that found solutions more quickly so that larger and more complex models could be solved. We needed to make optimization modeling with SAS/OR less an artistic application of data manipulation techniques and more a direct translation from the conceptual models and the algebraic model formulations that optimization professionals build. We needed to make it easier and more transparent to create the structure of an optimization model and populate it with the relevant data, whether the data comes from enterprise repositories, other SAS technologies, or other sources.

We also needed to shorten the "learning curve" for SAS/OR optimization by establishing greater consistency in the user interfaces provided by the various optimization procedures. We needed to provide a simple basic syntax that applied, with only minor variations, to all SAS/OR optimization procedures.

WHAT'S NEW IN OPTIMIZATION WITH SAS/OR SOFTWARE

The SAS/OR research and development team has delivered a set of all-new optimization procedures that meet and exceed all of the preceding requirements. New state-of-the-art optimizers deliver excellent performance and are coupled with the powerful, accessible new OPTMODEL optimization modeling language that not only eases and shortens the initial modeling effort but also makes it far easier to reuse models. This is especially important if one person builds a model and then transfers it to someone else for ongoing implementation and support; by virtue of the transparent OPTMODEL language the transferred model is largely self-documenting.

The lineup of new procedures, all carrying the "OPT" prefix to denote optimization, has been minimized to provide a one-to-one correspondence between a class of optimization problems and the corresponding SAS/OR procedure (* designates experimental features/procedures):

- **PROC OPTMODEL:** Optimization modeling language and access to all new optimization solvers (linear, general nonlinear, quadratic*, mixed-integer linear*)
- **PROC OPTLP:** Linear optimization; primal and dual simplex solvers, iterative interior-point solver
- **PROC OPTQP:** Quadratic optimization; iterative interior-point solver
- **PROC OPTMILP*:** Mixed-integer linear optimization; branch-and-bound solver

The sole exception to the "one procedure, one problem type" rule is PROC OPTMODEL. This procedure is the flagship of the new family of optimization procedures and is intended as a single point of access for building and solving optimization models, regardless of the type of model. Thus, learning to use PROC OPTMODEL equips you to build and solve any type of optimization model supported by SAS/OR. The remaining three procedures (OPTLP, OPTQP, and OPTMILP) are useful if you specify your optimization model directly from an input data set.

All of the new SAS/OR optimization procedures employ new, accelerated optimization solvers that have been built using state-of-the-art optimization techniques. Their performance is excellent, enabling SAS/OR users to solve problems much more quickly and to tackle larger problems than ever before. Aside from PROC OPTMODEL (in which you can create your own optimization model), the new procedures receive optimization models specified in SAS data sets that adhere to industry-standard data formats (MPS for linear and mixed-integer linear optimization, QPS for quadratic optimization). Users with models described using these formats can switch easily to using the SAS/OR procedures to solve them.

The next few sections take a closer look at each of the new SAS/OR optimization procedures.

PROC OPTMODEL AND THE OPTMODEL LANGUAGE

PROC OPTMODEL has been created to enable SAS/OR users to build optimization models easily, using modeling syntax and logical constructs that correspond directly to the algebraic statements used in symbolic formulations of optimization models. The OPTMODEL procedure also provides direct, targeted use of multiple input data sets, enabling much more precise use of data and close integration between optimization and SAS capabilities in data access and cleansing, predictive and descriptive analytics, business intelligence, and reporting.

The OPTMODEL procedure can access any of the new solvers created for SAS/OR, and in addition it can be used as a modeling interface for the OPTLP, OPTQP, and OPTMILP procedures. This helps the new family of SAS/OR optimization procedures to deliver a consistent user interface in two ways, first by making all optimization modeling resident in one procedure and then by structuring the other new procedures to receive input only in a well-defined, industry-standard format.

The OPTMODEL procedure includes an extensive set of commands for declaring all of the major elements of an optimization model. The most prominent elements, of course, are the decision variables, constraints, and objective. OPTMODEL also makes it easy to declare parameters, which define constant factors in the optimization model. Arrays and index sets for parameters and variables are also supported, enabling you to describe complex models with much greater clarity and simplicity. By establishing this structure in an optimization model, it's much easier to maintain a clear separation between the framework that the model establishes and the specific data that is inserted into this framework for any particular instance of the model.

PROC OPTMODEL supports a very broad range of programming statements and expressions. Input and output statements provide direct, precise, and flexible control over the use of input data in the model and enable you to output data on selected elements of the model or the optimal model solution with equal precision. Looping and control statements permit complex models to be created with few steps while preserving clarity. Expressions including aggregation, set, and conditional operators make it far easier to define which model parameters and which decision variables are affected by a particular constraint or appear in the objective.

The OPTMODEL procedure runs interactively; each command is executed as soon as it is submitted. This enables you to explore the impact of changes to the model (for example, removing or adding constraints) on the optimal solution. You can also monitor the optimization process as it proceeds, pause at selected intervals, and make any needed adjustments to settings for the optimization solvers.

PROC OPTMODEL, via its SOLVE command, enables you to use any of the new SAS/OR optimization solvers. OPTMODEL can itself determine an appropriate solver to use, or you may specify precisely which solver to apply, also choosing fine-tuning controls on the selected solver if you wish. In fact, by using OPTMODEL's programming

statements you can even program a completely customized solver, possibly using combinations of the built-in SAS/OR solvers, a custom-programmed solution method, or a combination of these.

PROC OPTMODEL accesses a set of nonlinear programming solvers. Three (Fletcher-Reeves, limited-memory BFGS, Polak-Ribière) are available for unconstrained nonlinear optimization, three (conjugate gradient, Newton-Raphson, trust region) for linearly constrained nonlinear optimization, and one (sequential quadratic programming) for nonlinearly constrained nonlinear optimization.

The OPTMODEL procedure can be used as a modeling interface for the other new SAS/OR optimization procedures; this is useful if you want to use OPTMODEL to create a baseline model but prefer to make further model revisions by modifying the model data directly. The SAVE MPS and SAVE QPS commands in OPTMODEL save the current optimization model in a SAS data set adhering to the indicated standard format, which can be input directly to the OPTLP, OPTMILP, and OPTQP procedures.

A brief example illustrates the intuitive nature of the OPTMODEL language. Consider a production planning problem in which various grades of cloth are to be produced on various machines for several customers. We know how many yards of each grade of cloth each customer demands along with the corresponding profit per yard (which varies according to the machine on which the cloth is made). We also have data on the available capacity of each machine and on the amount of time needed for each machine to produce one yard of each grade of cloth. The goal is to fill all of the customers' orders at maximum profit, while not exceeding the capacity of any machine.

For this problem the decision variable y_{cgm} represents the yards of cloth of grade g to produce for customer c on machine m. The parameter p_{cgm} represents the corresponding profit per yard, d_{cg} represents the demand of customer c for grade g cloth, t_{gm} represents the time needed on machine m to produce one yard of grade g cloth, and a_m represents the available capacity of machine m. With these definitions the optimization problem can be formulated symbolically as follows:

$$
\begin{array}{lll}
\max & \sum_{c,g,m} p_{cgm} y_{cgm} & (profit) \\
subject\ to & \sum_{m} y_{cgm} = d_{cg} \quad for\ all\ c,g & (demand) \\
 & \sum_{c,g} t_{gm} y_{cgm} \le a_m \quad for\ all\ m & (availability) \\
 & y_{cgm} \ge 0 \quad for\ all\ c,g,m &
\end{array}
$$

This formulation states that we seek to maximize profit, satisfy all demands, and stay within each machine's capacity.

Now let's look at the equivalent PROC OPTMODEL code for this problem. After some statements declaring the customers, grades, and machines and reading in the data described above (the full OPTMODEL code appears in the appendix), OPTMODEL describes the core of the optimization model:

```
var y{CUSTOMERS, GRADES, MACHINES} >= 0;

max obj = sum{c in CUSTOMERS, g in GRADES, m in MACHINES} profit[c,g,m]*y[c,g,m];

con req_demand{c in CUSTOMERS, g in GRADES}:
sum{m in MACHINES} y[c,g,m] = demand[c,g];

con req_avail{m in MACHINES}:
sum{c in CUSTOMERS, g in GRADES} time[g,m]*y[c,g,m] <= avail[m];
```

It's easy to see that this code corresponds directly to the symbolic formulation and is in many ways more "readable" than the formulation; after all, it's easier to understand "customers, grades, machines" than "c, g, m." The breadth,

simplicity, and power of the OPTMODEL language mean that you can define optimization models clearly and compactly, with a one-to-one correspondence between its syntax and your symbolic formulation.

It's important to remember that, while the OPTMODEL procedure incorporates a powerful, full-fledged optimization modeling language, it is still a SAS procedure. This means that a call of the OPTMODEL procedure can include SAS macro variables for further parameterization and can itself be included in a SAS macro. OPTMODEL can use virtually any of the dozens of available SAS functions in building an optimization model. It can be combined in a SAS program with other procedure calls and SAS commands that handle the data and analytic work that precedes and supports optimization, and the entire program can be registered as a stored process and called from SAS® Enterprise Guide® or the SAS Add-In for Microsoft Office. In short, in offering an optimization modeling language within a SAS procedure that is part of the SAS language, PROC OPTMODEL gives you the best of both worlds.

The remaining new optimization procedures in SAS/OR have been designed to be compact and straightforward since all of the modeling capabilities have been concentrated in the OPTMODEL procedure. These procedures provide targeted access to specific classes of solvers. They are useful if you prefer to work with a single input data set describing the entire optimization model or if you wish to separate the creation of your model from the solution process.

PROC OPTLP: LINEAR OPTIMIZATION

PROC OPTLP concentrates on the solution of linear programs, offering three solvers: primal simplex, dual simplex, and an experimental iterative interior-point algorithm. It includes an aggressive presolver that can work to reduce the effective size of the optimization model (and thus the time required to solve it) before the solver begins its work. The OPTLP procedure accepts models specified in a SAS data set that uses the MPS data format, which has become a standard in linear optimization.

PROC OPTQP: QUADRATIC OPTIMIZATION

PROC OPTQP solves quadratic programs with an iterative interior-point solver. This solver is especially designed to handle large-scale problems since many quadratic programming problems originate in large corporate settings and are broad in scope. The OPTQP procedure handles both sparse and dense problems well (in a sparse problem each constraint tends to involve relatively few of the decision variables; in a dense problem many decision variables are involved in most of the constraints). For the OPTQP procedure the problem should be defined in a SAS data set adhering to the QPS data format, an extension of the MPS format.

PROC OPTMILP: MIXED-INTEGER LINEAR OPTIMIZATION

PROC OPTMILP is an experimental procedure for solving mixed-integer linear programs by using a branch-and-bound technique based on the simplex method. This technique involves the sequential creation and solution of a series of related linear programs, with new, modified linear programs potentially being generated at each step. The OPTMILP procedure includes a presolver for reducing of the effective size of the model, and also employs cutting planes and primal heuristics, two techniques that help to speed the progress of the branch-and-bound method.

CONCLUSION: THE SAS ADVANTAGE

The new optimization procedures in SAS/OR provide accelerated, accessible optimization modeling and solution capabilities for a broad range of optimization problems. With more and more SAS users reaching the point of readiness (in data/analytic sufficiency and organizational maturity) for optimization and seeking to solve larger and more complex optimization problems, the advances in SAS/OR arrive at just the right time. With OPTMODEL and the other new "OPT" procedures, it's easier than ever for SAS/OR users to build optimization models from enterprise data, find optimal solutions, and report the key elements of those solutions so that they can be implemented successfully.

As we've seen, optimization is far from being a stand-alone technology. Nevertheless, SAS stands alone in providing the full range of data, analytic, and business intelligence capabilities that are essential to building, solving, and applying relevant optimization models. Moreover, SAS/OR offers the fullest range of optimization modeling and solution capabilities. With SAS you can be confident that your optimization modeling environment has a firm analytic foundation, the simplicity and flexibility that enable you to adapt easily to changing needs, and the power not only to inform but to add insight, innovation, and credibility to the decision-making process.

APPENDIX: OPTMODEL PROGRAM FOR PRODUCTION PLANNING PROBLEM

```
/* Revenue per unit of each grade of cloth, per machine, per customer */
data object;
input machine customer
grade1 grade2 grade3 grade4 grade5 grade6;
datalines;
1 1 102 140 105 105 125 148
1 2 115 133 118 118 143 166
1 3 70 108 83 83 88 86
1 4 79 117 87 87 107 105
1 5 77 115 90 90 105 148
2 1 123 150 125 124 154 .
2 2 130 157 132 131 166 .
2 3 103 130 115 114 129 .
2 4 101 128 108 107 137 .
2 5 118 145 130 129 154 .
3 1 83 . . 97 122 147
3 2 119 . . 133 163 180
3 3 67 . . 91 101 101
3 4 85 . . 104 129 129
3 5 90 . . 114 134 179
4 1 108 121 79 . 112 132
4 2 121 132 92 . 130 150
4 3 78 91 59 . 77 72
4 4 100 113 76 . 109 104
4 5 96 109 77 . 105 145
;

/* Demand by each customer for each grade of cloth */
data demand;
input customer
grade1 grade2 grade3 grade4 grade5 grade6;
datalines;
1 100 100 150 150 175 250
2 300 125 300 275 310 325
3 400 0 400 500 340 0
4 250 0 750 750 0 0
5 0 600 300 0 210 360
;

/* Machine time capacities and consumption per unit of each grade */
data resource;
input machine
grade1 grade2 grade3 grade4 grade5 grade6 avail;
datalines;
1 .250 .275 .300 .350 .310 .295 744
2 .300 .300 .305 .315 .320 . 244
3 .350 . . .320 .315 .300 790
4 .280 .275 .260 . .250 .295 672
;

proc optmodel;
set CUSTOMERS;
set GRADES = 1..6;
set MACHINES;

/* parameters */
number profit{CUSTOMERS, GRADES, MACHINES} init 0;
number demand{CUSTOMERS, GRADES};
number time{GRADES,MACHINES} init 0;
number avail{MACHINES} init 0;

/* load the customer set and demands */
read data demand
       into CUSTOMERS=[customer]
       {g in GRADES} <demand[customer,g]=col("grade"||g)>;
```

```
/* load the machine set, time costs, and availability */
read data resource nomiss
       into MACHINES=[machine]
       {g in GRADES} <time[g,machine]=col("grade"||g)> avail;

/* load objective data */
read data object nomiss
       into [machine customer]
       {g in GRADES} <profit[customer,g,machine]=col("grade"||g)>;

/* the model */
var y{CUSTOMERS, GRADES, MACHINES} >= 0;

max obj = sum{c in CUSTOMERS, g in GRADES, m in MACHINES} profit[c,g,m]*y[c,g,m];

con req_demand{c in CUSTOMERS, g in GRADES}:
sum{m in MACHINES} y[c,g,m] = demand[c,g];

con req_avail{m in MACHINES}:
sum{c in CUSTOMERS, g in GRADES} time[g,m]*y[c,g,m] <= avail[m];

/* call the solver and save the results */
solve with lp/solver=primal;

create data solution
       from [customer grade machine]
       ={c in CUSTOMERS, g in GRADES, m in MACHINES: y[c,g,m]^=0}
       amount=y;
quit;

proc print data=solution; run;

proc tabulate data=solution;
class customer grade machine;
var amount;
table (machine*customer), (grade*amount);
run;
```

ACKNOWLEDGMENTS

The authors gratefully acknowledge the contributions of Dr. Manoj K. Chari of the SAS/OR research and development staff, who reviewed and offered many helpful suggestions for improving this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors as follows:

Ed Hughes, SAS/OR Product Manager
SAS Institute Inc.
500 SAS Campus Drive, Office R-5322
Cary, NC 27513
Work Phone: 919-531-6916
E-mail: Ed.Hughes@sas.com
Web: www.sas.com

Trevor Kearney, Manager of Numerical Optimization
SAS Institute Inc.
500 SAS Campus Drive, Office R-5438
Cary, NC 27513
Work Phone: 919-531-7726
E-mail: Trevor.Kearney@sas.com

Web: www.sas.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies.