



---

## Optimization with SAS® Software

What is optimization and how can it add value?

---



## Table of Contents

<b>Introduction</b> .....	<b>1</b>
<b>What Is Optimization?</b> .....	<b>2</b>
<b>How Are Optimization Models Built and Solved?</b> .....	<b>2</b>
Elements and Categories of Optimization Problems .....	3
Beyond Traditional Optimization: Genetic Algorithms .....	4
Beyond Traditional Optimization: Constraint Programming .....	5
<b>From Business Scenario to Optimization Model</b> .....	<b>6</b>
The Role of Data and Analytics .....	7
<b>Benefits of Optimization</b> .....	<b>8</b>
Direct Benefits of Optimization .....	8
Tactical and Strategic Benefits of Optimization .....	9
Notes on Optimization and Business Intelligence .....	11
<b>Optimization with SAS®</b> .....	<b>12</b>
Trends in SAS/OR® Use for Optimization .....	12
<b>Optimization with SAS/OR®</b> .....	<b>13</b>
PROC OPTMODEL and the OPTMODEL Language .....	14
PROC OPTMODEL: Production Planning Example .....	16
PROC OPTMODEL: Facility Location Example .....	19
PROC OPTMODEL in Context .....	22
PROC OPTLP: Linear Optimization .....	22
PROC OPTQP: Quadratic Optimization .....	22
PROC OPTMILP: Mixed Integer Linear Optimization .....	23
PROC GA: Genetic Algorithms for Optimization .....	23
PROC CLP: Constraint Programming for Constraint Satisfaction Problems .....	24
Migration from Older SAS/OR® Optimization Procedures .....	24
<b>Optimization in SAS® Solutions</b> .....	<b>24</b>
SAS® Marketing Optimization .....	25
SAS® Revenue Optimization Suite .....	25
SAS® Service Parts Optimization .....	26
SAS® Size Optimization .....	26
<b>Conclusion: The SAS® Advantage</b> .....	<b>26</b>
<b>References</b> .....	<b>27</b>

## Introduction

---

SAS software includes a diverse and well-established array of optimization capabilities designed to assist in solving many different types of optimization problems. 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.

Users at a wide range of major corporations have found these capabilities to be indispensable in building decision-support and decision-guidance solutions aimed at identifying the best and most profitable strategies and action plans for their enterprises.

SAS/OR includes an extensive suite of optimization modules (procedures) that make it easier than ever to tackle a broad range of optimization problems. The foremost of these is the OPTMODEL procedure, which incorporates 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. Procedures for solving specific types of optimization problems (specified by using input data sets) are also provided: OPTLP for linear optimization, OPTMILP for mixed-integer optimization and OPTQP for quadratic optimization. Supporting these procedures is a set of optimization solvers that use state-of-the-art algorithms and deliver excellent performance and scalability.

Two additional procedures address problems that can be difficult to model and solve with classical optimization methods. The GA procedure uses genetic algorithms to solve optimization problems, and the CLP procedure solves constraint satisfaction problems.

Complementing SAS/OR are several SAS solutions that employ optimization as either a core function or a critical enabling technology. Examples include: SAS Marketing Optimization, SAS Revenue Optimization Suite, SAS Size 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. 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 software delivers to optimization modelers. We consider the SAS/OR optimization features and see how they make the building and solving of optimization models easier, more transparent and more scalable than ever before.

---

## What Is Optimization?

---

Simply put, optimization is the process of choosing the permissible actions that result in the best outcome. This is the underlying concept, no matter how complex the means by which optimization is implemented. 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 might 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 or unstructured attempts at optimization. In most cases a more precise description of the decision problem and a more disciplined approach to optimization is needed. This more rigorous approach (termed *mathematical optimization* or, equivalently, *mathematical programming*) can mitigate or eliminate many of the problems we have described.

---

## How Are Optimization Models Built and Solved?

---

Optimization has many applications across a limitless range of industries and organizations, but beneath the surface the fundamental logical and mathematical structures of optimization problems follow some well-established patterns. Knowing how to recognize and take advantage of these structures is the key to good modeling and solution methods.

## 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**, which 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**, which specify requirements or rules and place 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, human resources limitations and budgetary restrictions.

Each type of optimization model is characterized 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 (for example,  $4x + 3.5y - 2z$ ) in the constraints and the objective, 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  $x^2$ ,  $\sin(y)$ ,  $1/x$ , and others) in the constraints or the objective (or both).
- **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 are tailored to suit each type of optimization model, but in general all begin with an initial solution and then seek to improve it until no further improvement can be found (or until some other predefined criterion for halting is satisfied). At this point the current solution is deemed optimal. If the problem is infeasible (the constraints can never be satisfied) or unbounded (the objective function can be made infinitely positive or negative while still obeying the constraints), the solution process will detect it.

## Beyond Traditional Optimization: Genetic Algorithms

Many compelling business problems do not fit neatly into any of the categories previously described. Some optimization problems have an objective function that is not sufficiently “well-behaved” for traditional modeling and solution methods to work well – or at all. The objective function might not be continuous or might not be smooth due to the nature of the real-world performance metric that it represents. Instead of a single “global” optimal value, the function might have many “local” optimal values (so that a graph of the objective resembles a mountain range with many small peaks instead of one large peak). There might be multiple objective functions. The objective might not even be expressible as an algebraic function; perhaps you can determine its value only by consulting a table and finding the value that corresponds to a particular set of values for the decision variables.

Further difficulties can arise with decision variables; they might be constrained to take on only integer or sequence values, creating a new set of challenges to the use of traditional methods. These sorts of challenges can arise in areas such as securities management, production sequencing and route planning.

In these cases, genetic algorithms can be enormously useful because they make no assumptions about the structure of the problem being solved. Genetic algorithms work with a set of solutions to the problem, treat them like a population of organisms and use functions called genetic operators to evolve the population of solutions over successive generations. These genetic operators are: “crossover operators,” which determine how solutions in one generation combine to create new solutions in the next generation; and “mutation operators,” which introduce random variation into this process. Another important element of genetic algorithms is the selection process that spells out the rules by which solutions from the current generation are chosen to either parent the next generation or be passed on to the next generation directly.

The selection process tends to favor the “fittest” solutions as measured by their corresponding objective function values, and it is these solutions that are most likely to survive and combine to form new solutions that inherit their characteristics. Selective pressure, favoring the fittest solutions, is balanced against genetic diversity with the mutation operator introducing random variation in the solution population. The goal is for the algorithm to reach a point at which the population collectively takes on the characteristics of an optimal solution. This point might be difficult to reach in practice, but at the very least genetic algorithms can be used to find good solutions to difficult optimization problems in a reasonable amount of time.

One of the greatest strengths of genetic algorithms is their versatility, owing to the fact that they do not assume that the problem being solved has any special properties or any particular structure. Thus, for problems where there is no optimization method that can exploit its specific characteristics, genetic algorithms might be the best approach. However, if such a method does exist, then it is probably quicker and less computationally intensive than genetic algorithms.

## Beyond Traditional Optimization: Constraint Programming

There are other types of business problems that do not, strictly speaking, manifest themselves as optimization problems because there is no compelling objective to maximize or minimize. Instead the focus is on finding one or more (or all) solutions that satisfy the stated constraints. Not surprisingly these are termed “constraint satisfaction problems” (CSPs).

A CSP is defined solely by its decision variables, their respective domains (or sets of permissible values) and the constraints. Therefore, in theory you could simply describe a CSP as an optimization problem where the objective function is a constant, unchanged by changes in the values of the decision variables, and solve it with standard methods. Unfortunately, in CSPs both the decision variables and the constraints typically have properties that make it difficult to use traditional optimization methods to describe and solve the problem.

One common difficulty concerns the domain for each decision variable. Usually, decision variables in CSPs are defined over finite domains (there is a finite number of permissible values), but these domains might not be something easy to depict, such as ranges of consecutive integers (whole numbers). For example, a domain might contain all even numbers from 0 to 20 except for 14. For scheduling problems, which make up a large percentage of CSPs, the decision variables might be multidimensional. For example, the decision variables might represent the starting time of each activity in the schedule and the resource assigned to each activity at the indicated start time. Many such domains cannot be represented effectively with traditional optimization models. Others require so much additional work that the modifications obscure the basic structure of the problem, making it more difficult to understand and potentially much more difficult to solve.

Constraints in CSPs present their own challenges to traditional optimization modeling and solution methods because they can comprise intricate logical restrictions or relationships that are cumbersome to represent in a traditional optimization model. In scheduling and sequencing problems, which constitute a substantial segment of CSPs, specialized constraints specifying that events must occur at different times or with a lag time of at least a certain duration elapsing between them can also place unreasonable burdens on the traditional optimization methods. The resulting optimization models might be possible to construct, but they are often far too large to solve effectively.

A class of solution methods that works well with CSPs is referred to as constraint programming and differs very significantly from the traditional approach of iteratively improving a single solution until reaching optimality. Constraint programming instead builds a solution element by element, assigning a value to one decision variable at a time and tracking the effect of each such assignment on the permissible values for the unassigned decision variables. This is accomplished via a technique known as constraint propagation because assigning a value to one decision variable affects only the decision variables that appear in one or more constraints with the just-assigned decision variable. For example, assume that  $x$  and  $y$  are decision variables and each can take on an integer value between 1 and 5. Also assume that the constraint  $x+2y \leq 10$  is part of the CSP. If you assign a value of  $x = 4$ , then the domain for  $y$  is reduced to integers between 1 and 3; any greater value for  $y$  violates the constraint.

For larger sets of decision variables and constraints, this is a much more complex process. It is also possible that at some point the values assigned so far for decision variables will eliminate, via constraint propagation, all of the permissible values for at least one of the unassigned variables. Accordingly, constraint programming also includes backtracking methods for dealing with such conflicts and consistency techniques, selection strategies, and assignment strategies for avoiding conflicts. Constraint programming also includes techniques for expressing constraints and domains that are attuned to the characteristics of these elements of CSPs. The end result is that a constraint programming formulation of a CSP is often much smaller and more direct than other formulations such as an equivalent formulation that uses integer programming methods. Constraint programming is adept at producing **feasible solutions** to complex CSPs.

---

## From Business Scenario to Optimization Model

---

The outlines of the process for creating optimization models are 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.

The next step is to describe this conceptual model more precisely 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 sets or ranges of allowable values, each corresponding to a permissible decision.

- 
- A **feasible solution** to an optimization problem or a CSP is a set of values for the decision variables that satisfy all of the stated constraints. Often finding a feasible solution is a challenge in and of itself, highlighting the importance of CSPs – in which satisfying the constraints is the sole purpose.
-

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

This means that once you've built the model and solved it (as described later), you need to consider whether the optimal solution that has been found represents a workable response to the original business problem. If not, you may have overlooked or misconstrued a key element of the problem. Your model might be correct, but some of the data used by the model might not be accurate. The interpretation of the original business problem could be flawed. In these and other such cases you need to step back carefully through the modeling process, address the difficulties and then move forward with the improved model. This [iterative process fosters continuous learning](#) and represents another aspect of the art of optimization modeling.

## The Role of Data and Analytics

The optimization modeling process always involves data. Data is what brings an optimization model to life. Data populates the logical structure of an optimization model and makes it relevant to the decision problem at hand.

Data is critical to optimization, but data can also create its own challenges. Optimization modeling often 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 to compile the descriptive and predictive information that characterizes the important relationships 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 stage 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.

---

■ The optimization modeling process always involves data. Data is what brings an optimization model to life.

---

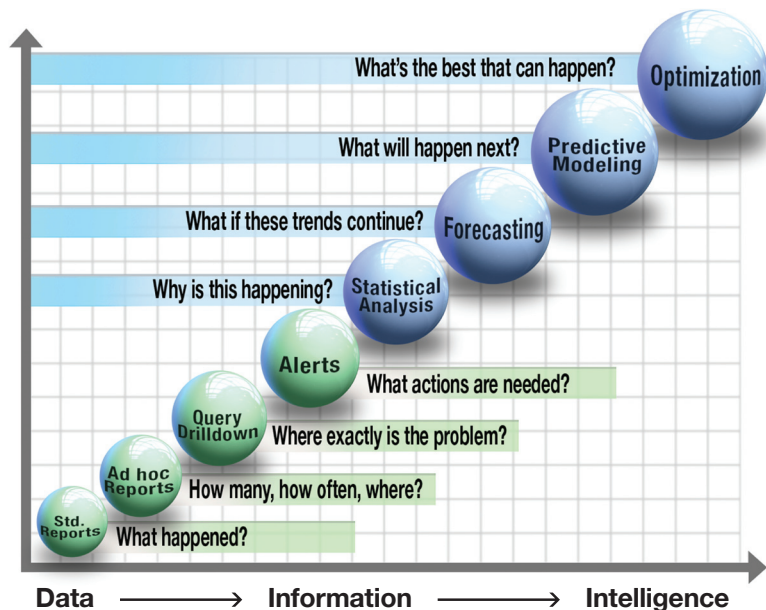


Figure 1: The data and analytic continuum that supports optimization.

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 is also worth noting that while it is 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.

It is worth noting that while it is 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.

## Benefits of Optimization

There are many advantages for organizations that invest time and effort in building optimization into their decision processes. These benefits, however, can reach far beyond the obvious.

### 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 five possible production levels may be chosen for each of eight products, there are more than 39,000 possible combinations of production levels overall. Simply calculating an objective for each combination and determining if it satisfies all constraints would take an inordinate amount of time when attempted manually, while a good optimization algorithm may produce an optimal solution in a matter of seconds.

Optimization can also produce great improvements in the quality and effectiveness of the decisions that are identified. Because 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 that use manual methods rely on intuition and experience, which while providing useful insights are limited in their scope. Optimization offers a fresh, unbiased view of the decision scenario and framework, often identifying alternative actions that never before have been considered.

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

## Tactical and Strategic Benefits of Optimization

In addition to solving the problem as stated above, the optimization process often produces 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:

- 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 effect on operational planning as well.

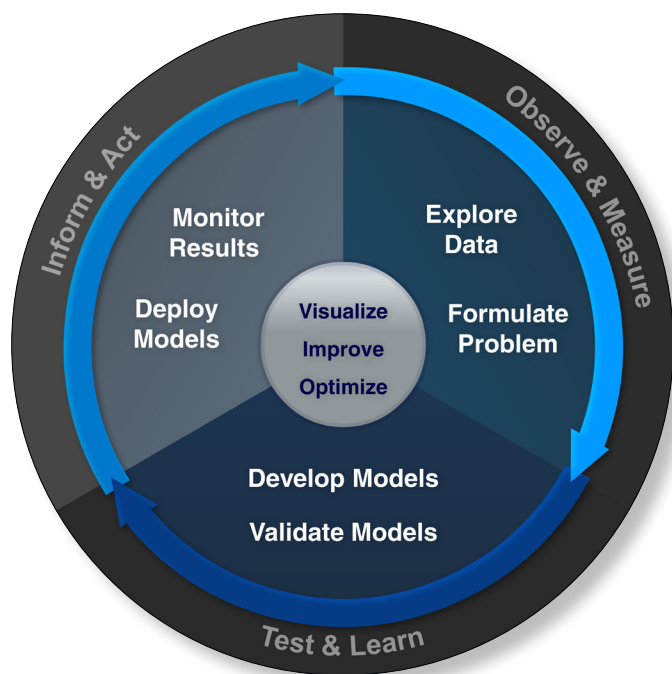
In another sense optimization, when employed thoughtfully, can transform the entire decision-making process. For organizations making decisions purely on an intuitive, experience-oriented basis, the decision-making power is intrinsically linked to the individuals in the organization who have amassed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

Optimization can remedy 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, when employed thoughtfully, can transform the entire decision-making process.

---



*Figure 2: The analytics cycle of continuous improvement provides a strategic approach to making decisions with an analytics perspective.*

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 as both seek to influence decision making, but instead the opposite is true. Optimization and business intelligence can and do work together to overcome their individual shortcomings and magnify their respective strengths.

Business intelligence unquestionably provides great value to organizations by highlighting key relationships, surfacing performance metrics, looking toward future performance, identifying opportunities for improvement and much more.

However, it is 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 are 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.

---

■ The relationship between optimization and business intelligence is not competitive but symbiotic.

---

## 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 or hierarchical relationships (or both). 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 (including customer service operations, manufacturing processes and medical treatment facilities) for which critical elements – such as arrival of individuals, processing or treatment times, preferences, requirements or other elements – 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.

### Trends in SAS/OR® Use for Optimization

Over the past several years, a number of patterns have emerged regarding the use of SAS/OR for optimization. First, our customers are applying optimization much more broadly than ever before. This is due to several causes, one being the general rise in competition among enterprises as consumers have become better informed and less committed to single companies or brands. This leads 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 optimization capabilities is growing and is becoming much more diverse.

Second, optimization in SAS/OR is being used more frequently in conjunction with other SAS analytic products in areas such as statistics, data mining and forecasting. Data sets produced by these methods are used as input for optimization, which means that considerable processing must 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 is being used to solve larger and more complex optimization problems. It's not surprising that organizations, as they see benefits from optimization, move to grow their optimization uses in scale and scope.

These are very positive developments because they signify better-coordinated planning and more data-driven decision making in organizations today. Optimization experts seek software that will find solutions as quickly as possible so that they can solve larger and more complex problems. SAS/OR modeling is, as much as possible, a direct translation from the conceptual models that the optimization professional creates to the algebraic representation of their model formulation. It should be a straightforward process 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. SAS has moved to address all of these challenges and will continue to do so.

---

## Optimization with SAS/OR®

---

Optimization has been part of SAS/OR since its inception in 1983, and the optimization capabilities in SAS/OR have grown steadily over time, serving thousands of organizations worldwide. In 2006, optimization with SAS/OR made a significant leap forward with the introduction of the OPTMODEL family of optimization procedures. Based on years of development effort, this group of procedures fully satisfies the needs of optimization professionals and in several ways exceeds them. State-of-the-art optimizers deliver excellent performance and are coupled with the powerful, accessible 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 (as often happens) 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 SAS/OR optimization procedures has been designed to provide a one-to-one correspondence between a class of optimization problems and the corresponding SAS/OR procedure:

- **The OPTMODEL procedure:** optimization modeling language and access to all new optimization solvers.
- **The OPTLP procedure:** linear optimization.
- **The OPTMILP procedure:** mixed-integer linear optimization.
- **The OPTQP procedure:** quadratic optimization.
- **The GA procedure:** optimization via genetic algorithms.
- **The CLP procedure:** solution of constraint satisfaction problems via constraint programming.

Among the SAS/OR procedures that carry the OPT prefix, the sole exception to the “one procedure, one problem type” rule is PROC OPTMODEL. This procedure is the flagship of this 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 traditional optimization model supported by SAS/OR. The remaining three OPT procedures (OPTLP, OPTQP and OPTMILP) are useful if you choose to specify your optimization model using an input data set and do not require the interactive modeling capabilities of PROC OPTMODEL.

The OPTMODEL, OPTLP, OPTMILP and OPTQP procedures use 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 models), the OPT 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 SAS/OR procedures to solve them.

The GA procedure and the experimental CLP procedure address specialized classes of problems. As such, they use specialized modeling and solution methods, contrasting with the more broadly applicable methods used in the OPT procedures.

The next few sections take a closer look at each of these 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 SAS/OR software’s LP, MILP, QP and NLP solvers, and it can also be used as a modeling interface for the OPTLP, OPTMILP and OPTQP procedures. This establishes a consistent user interface in two ways: first, by making all optimization modeling capabilities resident in one procedure; and second, by structuring the other 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 is 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 that include 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. Implicit variables (decision variables defined as expressions of other decision variables) are supported.

PROC OPTMODEL supports the use of named optimization problems to facilitate manipulation of multiple subproblems in, for example, more advanced optimization algorithms. Objective functions may be declared as arrays so they can provide separate objectives for arrays of named problems.

The OPTMODEL procedure runs interactively with each command 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.

The SOLVE command in PROC OPTMODEL enables you to use many SAS/OR optimization solvers. (See the sections for PROC OPTLP, PROC OPTMILP and PROC OPTQP for details about available linear, mixed integer and quadratic optimization solvers.) PROC OPTMODEL can automatically determine an appropriate solver to use or you can specify precisely which solver to apply, also choosing fine-tuning controls as needed on the selected solver. In fact, by using PROC OPTMODEL's programming statements, you can even design a customized solver or define a hybrid of the built-in solvers and a custom-programmed solution method.

For general nonlinear programming, PROC OPTMODEL can access a number of solvers depending upon the characteristics of the model. Multiple solvers are available for each of unconstrained nonlinear optimization, linearly constrained nonlinear optimization and 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 PROC 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 the OPTMODEL procedure save the current optimization model in a SAS data set that adheres to the indicated standard format, and it can be input directly to the OPTLP, OPTMILP and OPTQP procedures as appropriate.

Two brief examples illustrate the intuitive nature and flexibility of the OPTMODEL language.

### PROC OPTMODEL: Production Planning Example

In this example, various grades of cloth are to be produced on multiple 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:

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

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 that declare the customers, grades and machines and read in the data described above, PROC OPTMODEL describes the core of the optimization model as follows (the full program appears in Appendix 1 of <http://support.sas.com/resources/papers/proceedings09/300-2009.pdf>):

```
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 is easy to see that these statements correspond directly to the symbolic formulation and are in some ways more readable than the formulation; after all, it is easier to understand CUSTOMERS, GRADES and MACHINES than  $c, g$  and  $m$ . 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.

Supporting this transparent optimization code are the READ DATA and CREATE DATA statements of PROC OPTMODEL, which enable you to work easily with SAS data sets to provide input to optimization models and to receive output after an optimal solution has been reached. In the PROC OPTMODEL syntax for the production planning problem, we must read in data on:

- The profit associated with each combination of customer, cloth grade and machine.
- The demand by each customer for each grade of cloth.
- The production time required per yard of cloth on each machine.
- The available production time for each machine.

The READ DATA statement in PROC OPTMODEL accomplishes this with precision and clarity. Consider the data on customer demand, which looks like this:

```
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
;
```

The interpretation of this data is direct. For example, the second observation has a value of 2 for the variable `customer` and a value of 310 for the variable `grade5`. This means that customer 2 demands 310 yards of grade 5 cloth. The READ DATA statement syntax that makes this data available to the optimization model is equally direct:

```
read data demand into CUSTOMERS=[customer]
{g in GRADES} <demand[customer,g]=col("grade" || g)>;
```

This READ DATA statement reads the SAS data set named `demand`. The first part of the statement indicates that the values of the variable `customer` are being used to populate a set, `CUSTOMERS`, that is used by PROC OPTMODEL to store customer identifiers. This makes the set `CUSTOMERS` completely data driven, which is useful because for any period of time the list of customers is likely to change. The second part of the statement imports demand information into the optimization model by populating the array `demand`. This is little more than a formal statement of the explanation of the values in the `demand` data set given in the preceding paragraph.

We use the SOLVE command in PROC OPTMODEL to invoke the primal simplex solver and identify an optimal solution, and we use the CREATE statement in PROC OPTMODEL to output that solution:

```
/* 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;
```

This CREATE DATA statement creates a new SAS data set called `solution` from the optimal values of the  $y_{cgm}$  decision variables. The statement indicates first that the values for the variables `customer`, `grade` and `machine` are to be copied directly from the corresponding values in the optimal solution. The second section of the statement creates a new variable, `amount`, which takes on the value of  $y_{cgm}$  whenever  $y_{cgm}$  has a nonzero value. Thus, we use the CREATE DATA statement to filter information about the optimal solution so that we can focus on the nonzero decision variable values. This limits the output to the actions recommended by the optimization process (the decision variables with values of zero at optimality correspond to actions that are not recommended).

The data set `solution`, which contains information about the optimal production plan, can now be passed to any SAS procedure for tabular or graphical reporting. This data set can be published in any format suitable for SAS data sets (the TABULATE procedure is used in this case). In short, any of the information delivery vehicles supported by SAS can be used at this point. Thus, PROC OPTMODEL can integrate smoothly with SAS reporting and business intelligence.

## PROC OPTMODEL: Facility Location Example

In this example, the challenge is to determine the best locations and customer assignments for facilities in a new distribution network. Information is available about the locations and projected demand levels of the customers that will be served by the new facilities. Information is also available about the locations of the possible sites for new facilities. All such facilities will have the same capacity (maximum number of units that can be supplied to customers).

For each candidate location there is a specific fixed charge that we will incur if we place a new facility at that location. This fixed charge could represent some or all of land acquisition costs, site preparation and building costs. We also need to determine which customers will be served by each of the facilities we choose to build. Serving each customer incurs a per-unit variable charge equal to the distance between the facility site and the customer's location. We must satisfy each customer's demand in full, and each customer must be served by only one facility. We must not over-allocate the capacity of any facility to serve customers. The objective is to minimize total cost. We consider this problem both with and without the fixed charges so that we can gauge their effect on the optimal decisions.

For this problem there are two sets of decision variables. The decision variable  $b_s$  takes a value of 1 if a facility is built at site  $s$  and takes a value of 0 otherwise. Similarly,  $a_{cs}$  takes a value of 1 if customer  $c$  is assigned to site  $s$  and takes a value of 0 otherwise (decision variables that can be set only to 0 or 1 are referred to as *binary variables*). The parameter  $dist_{cs}$  represents the distance between the location of customer  $c$  and the location of site  $s$ ; the parameter  $dem_c$  represents the number of units demanded by customer  $c$ . The capacity of any facility is represented by the parameter  $cap$ , and the fixed charge for site  $s$  is  $f_s$ . The formulation of this optimization problem follows. Note that we provide two versions of the objective, one including fixed charges and one excluding them.

$$\begin{array}{ll}
 \min & \sum_{c,s} dist_{cs} a_{cs} + \sum_s f_s b_s \quad (\text{variable costs} + \text{fixed costs}) \\
 \text{subject to} & \sum_s a_{cs} = 1 \quad \text{for all } c \text{ (one site per customer)} \\
 & a_{cs} \geq 0 \quad \text{for all } c, s \text{ (build each assigned facility)} \\
 & \sum_c dem_c a_{cs} \leq cap \cdot b_s \quad \text{for all } s \text{ (respect site capacities)} \\
 & a_{cs}, b_s \in \{0,1\} \quad \text{for all } c, s \text{ (yes/no decisions)}
 \end{array}$$

As in the previous example, the PROC OPTMODEL statements resemble the symbolic formulation very closely. After the customary declaration statements and some statements that compute the Euclidean (straight-line) distances between customer locations and candidate sites, the following statements define the model (the full OPTMODEL program appears in Appendix 2 of <http://support.sas.com/resources/papers/proceedings09/300-2009.pdf>):

```
var Assign {CUSTOMERS, SITES} binary;
var Build {SITES} binary;

min CostNoFixedCharge
= sum {c in CUSTOMERS, s in SITES} dist[c,s] * Assign[c,s];
min CostFixedCharge
= CostNoFixedCharge + sum {s in SITES} fixed_charge[s] * Build[s];

/* each customer assigned to exactly one site */
con assign_def {c in CUSTOMERS}:
sum {s in SITES} Assign[c,s] = 1;

/* if customer c assigned to site s, facility must be built at site s */
con link {c in CUSTOMERS, s in SITES}:
Assign[c,s] <= Build[s];
/* each site can handle at most &SiteCapacity demand */
con capacity {s in SITES}:
sum {c in CUSTOMERS} demand[c] * Assign[c,s] <= &SiteCapacity * Build[s];
```

Decision variables, two versions of the objective function, and constraints are all clearly defined in the PROC OPTMODEL statements. Note especially that the definition of the total cost objective including fixed charges is able to use the previously defined “no-fixed-charges” objective in its equation.

You can run this optimization model with and without fixed charges simply by issuing a SOLVE command that specifies your desired objective function. In each case the results of the optimization are written to a SAS data set using the CREATE DATA statement and are then passed to the GPLOT procedure from SAS/GRAPH® software for display. If you issue the following command, then you produce a solution in which facilities are built at all candidate sites.

```
solve obj CostNoFixedCharge;
```

This is not surprising. Because the only goal is to limit the distance-related costs of serving customers, it is advantageous to build as many facilities as possible. The PROC GPLOT output is shown in Figure 3.

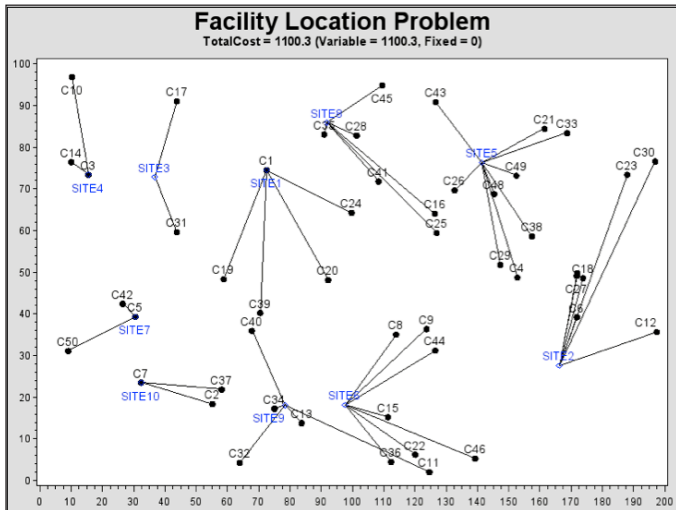


Figure 3: Customer assignment to facility locations without considering fixed costs of building/operating each site.

In Figure 3 the candidate sites are shown as blue squares and the customer locations as black dots, with the line segments indicating the assignments of customers to sites.

If instead you issue the following command, then you produce an optimal solution in which the reduction in variable charges from building more facilities is balanced against the increased fixed charges from doing so.

```
solve obj CostFixedCharge;
```

The net result is that facilities are built at fewer sites, as shown in Figure 4.

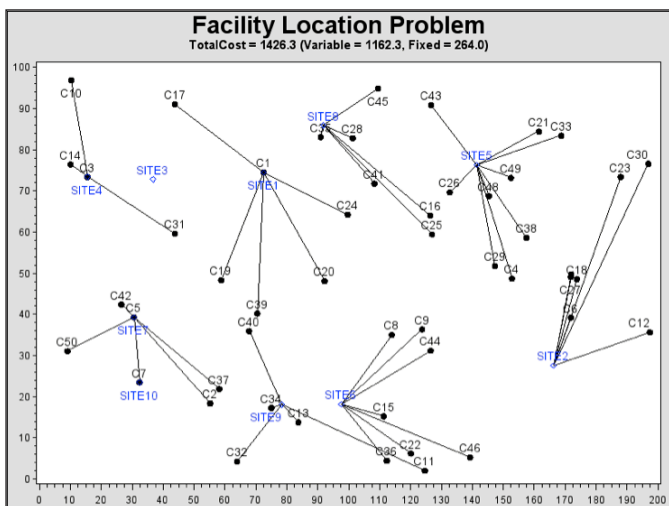


Figure 4: Facility location solution with fixed charges.

Note that sites 3 and 10 no longer host facilities and that the assignments of customers have been adjusted accordingly.

## PROC OPTMODEL in Context

It is 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. PROC OPTMODEL can use virtually any of the dozens of available SAS functions in building an optimization model without relying on scripting languages or other devices. 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. In short, by providing 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 OPT optimization procedures in SAS/OR have been designed to be compact and straightforward because 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 that describes the entire optimization model or if you want 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 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 LP solvers can also help to localize infeasibility when it occurs by identifying an irreducible infeasible set (IIS) of constraints and variable bounds for infeasible problems.

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 interior point solver. This solver is especially designed to handle large-scale problems because 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 that adheres to the QPS data format, which is an extension of the MPS format.

---

■ By providing an optimization modeling language within a SAS procedure that is part of the SAS language, PROC OPTMODEL gives you the best of both worlds.

---

## PROC OPTMILP: Mixed Integer Linear Optimization

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

The OPTMILP procedure accepts models specified in a SAS data set using the MPS format.

Finally, two additional procedures provide more specialized methods.

## PROC GA: Genetic Algorithms for Optimization

PROC GA enables you to specify and solve optimization problems with a genetic algorithms approach. It supports all of the following steps in defining the problem and its solution method:

1. Specify the supporting data for the problem.
2. Specify the basic parameters of the genetic algorithm:
  - a. Encoding: determining how solutions to the problem are represented (integers, real numbers, character strings or a combination of these).
  - b. Objective function.
  - c. Selection process: determines how members of the current generation of solutions are chosen to move unchanged or combine to form the next generation.
  - d. Crossover operator: determines how the attributes of parent solutions in one generation are combined to produce offspring solutions in the next generation.
  - e. Mutation operator: determines how random variation is introduced into offspring solutions in order to maintain genetic diversity in the solution population.
3. Generate an initial population of solutions.
4. Control the execution of the genetic algorithm and record the results.

For the selection process and the crossover and mutation genetic operators, PROC GA enables you to select from predefined methods (with parameters available for tuning) or to specify a completely customized method. The procedure offers similar flexibility in defining the encoding, specifying the objective function, creating the initial population, and setting controls on the execution of the algorithm.

## **PROC CLP: Constraint Programming for Constraint Satisfaction Problems**

---

PROC CLP is a finite-domain constraint programming procedure for use with constraint satisfaction problems (CSPs) featuring linear, logical and/or scheduling constraints. It includes a compact and expressive syntax for specifying the key elements of CSPs: decision variables, decision variable domains and constraints. It also enables you to set several parameters that control the methods used in building a solution to the CSP. PROC CLP is equipped to solve general CSPs and also includes the ability to model and solve scheduling CSPs, in which the goal is to determine a feasible schedule for a group of interrelated activities and associated resources. You can direct PROC CLP to find one solution, a specified number of solutions, or all solutions to a CSP.

## **Migration from Older SAS/OR® Optimization Procedures**

---

For PROC LP, PROC INTPOINT and PROC NETFLOW in SAS/OR, the MPSOUT= option enables you to save a linear or mixed integer linear optimization model built with the older procedure as a SAS data set that conforms to the MPS format requirements. The data set can then be input to PROC OPTLP or PROC OPTMILP, as appropriate. This enables you to take advantage of the newest SAS/OR optimization solvers without the need to recreate prior modeling work. For nonlinear optimization, the PROC OPTMODEL documentation includes an extensive section detailing how to migrate from the older PROC NLP to PROC OPTMODEL.

## **Optimization in SAS® Solutions**

---

Optimization with SAS is by no means limited to the SAS/OR product. A growing number of SAS solutions, tailored to address specific business problems (often within specific industries), rely upon much of the same optimization technology employed by SAS/OR to produce optimal solutions but often implement that technology in innovative ways that are customized to suit the problem at hand. Additionally, a diverse set of SAS products and solutions rely upon optimization as an enabling analytic technology. Interaction between the respective development teams provides feedback on the practical use of SAS optimization technology that is enormously beneficial for all.

A brief look at a few of the other SAS offerings that feature optimization follows.

## SAS® Marketing Optimization

SAS Marketing Optimization assists direct marketing efforts by providing guidance on how best to market multiple products to as many as millions of individual customers, while at the same time accounting for the multiple communications channels (Web, e-mail, text, call center, mail, etc.) that may be used. The solution balances competing business goals, at the same time accounting for multiple marketing programs and constraints such as channel capacity, budget and customer-contact policies. Marketers can tailor campaigns to maximize profitability, response rates, asset levels or any other parameter they choose. In addition, SAS Marketing Optimization provides a what-if environment for modifying assumptions about these critical factors, yielding more insight into the economic impact of business constraints. SAS Marketing Optimization addresses these complex issues by delivering sophisticated optimization methods in an easy-to-use interface designed for the business user.

## SAS® Revenue Optimization Suite

SAS Revenue Optimization Suite helps retailers manage revenue and margin through the entire merchandise life cycle. This suite combines advanced data management, forecasting and optimization capabilities within an easy-to-use interface that helps retailers set and manage regular prices, plan optimal promotions and execute the most successful markdown strategies.

- **SAS Regular Price Optimization.** SAS Regular Price Optimization automates and optimizes the regular price determination process at the product and store level. It uses optimization and forecasting in conjunction with business rules that reflect pricing strategies, sales and margin goals, and competitive data to optimally generate regular price recommendations for products.
- **SAS Promotion Optimization.** SAS Promotion Optimization is a comprehensive promotion planning solution that provides retailers the ability to maximize margin and revenue by accurately determining which items or groups of items to promote, in which locations, at what price points and via which promotional vehicles. The solution helps to maximize return on investment in promotional activities and to minimize lost revenue due to stock-outs on promoted merchandise.
- **SAS Markdown Optimization.** SAS Markdown Optimization provides retailers the ability to determine which items should be marked down, by how much, when and in which markets or stores. Retailers can base this analysis on critical factors such as inventory levels, base-sales volume, price elasticity and local demand or preferences. Retailers can develop and implement optimum pricing strategies at the local market level to maximize revenues and profitability while meeting end-of-season inventory goals.

## SAS® Service Parts Optimization

---

SAS Service Parts Optimization enables companies to optimize their service parts inventories over a multi-echelon service network. By providing service organizations the ability to forecast short-term, new-part and lifecycle parts demand, as well as calculate optimized inventory and order quantities for parts-distribution systems, the solution helps organizations maintain sufficient stock levels, minimize response times and improve customer satisfaction. The solution is scalable to encompass hundreds of thousands or millions of SKUs at each location in the network.

## SAS® Size Optimization

---

SAS Size Optimization is a solution that assists retailers in ordering sized merchandise that is shipped in case-packs and consists of two primary software modules: SAS Size Profiling and SAS Pack Optimization.

- **SAS Size Profiling.** SAS Size Profiling generates accurate size profiles per store in an interactive but still substantially automated manner. These profiles enable retailers to create size distributions for any desired set of merchandise, accounting for both preseason and in-season profiling.
- **SAS Pack Optimization.** SAS Pack Optimization determines the optimal quantities of each case-pack configuration needed to best meet size-level demand forecasts. Specifically, SAS Pack Optimization recommends the profit-maximizing combination of available case-packs to purchase or allocate, taking both inventory mismatch and handling costs into consideration.

---

## Conclusion: The SAS® Advantage

---

The SAS/OR optimization procedures provide scalable, accessible optimization modeling and solution capabilities for a broad range of practical 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 optimization in SAS/OR are well-timed. With PROC OPTMODEL and the other OPT procedures, it is 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. The GA procedure enables the use of genetic algorithms, increasing the range of optimization problems that can be modeled and solved using SAS/OR software. The CLP procedure extends the reach of SAS/OR users in both optimization and project scheduling because it enables you to model and solve both general and scheduling-oriented constraint satisfaction problems.

SAS continues to develop solutions featuring optimization which complement SAS/OR software's optimization capabilities quite well. These solutions each address a specific business problem that arises either in a single industry or across industries. Within specific domains, they demonstrate the promise that SAS/OR holds on a more general basis – that optimization can and should be used to guide better decisions.

As noted, optimization is far from being a standalone technology. Nevertheless, SAS as a software company 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. 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.

---

## References

---

SAS Institute Inc. 2009. *SAS/OR User's Guide: Mathematical Programming*. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2009. *SAS/OR User's Guide: Local Search Optimization*. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2009. *SAS/OR User's Guide: Constraint Programming*. Cary, NC: SAS Institute Inc.

<http://support.sas.com/software/products/or/index.html#documentation>



**SAS Institute Inc. World Headquarters +1 919 677 8000**

To contact your local SAS office, please visit: **[www.sas.com/offices](http://www.sas.com/offices)**

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.  
Copyright © 2010, SAS Institute Inc. All rights reserved. 104496\_S52301.0610