SAS/OR® 14.1 User's Guide: Mathematical Programming

The Quadratic Programming Solver

This document is an individual chapter from *SAS/OR® 14.1 User's Guide: Mathematical Programming*.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2015. *SAS/OR® 14.1 User's Guide: Mathematical Programming*. Cary, NC: SAS Institute Inc.

SAS/OR® 14.1 User's Guide: Mathematical Programming

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

July 2015

Chapter 11
The Quadratic Programming Solver

Contents

Overview: QP Solver

The OPTMODEL procedure provides a framework for specifying and solving quadratic programs.

Mathematically, a quadratic programming (QP) problem can be stated as follows:

$$\begin{aligned} \min \quad & \tfrac{1}{2}\,\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x}\ \{\geq, =, \leq\}\ \mathbf{b} \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u} \end{aligned}$$

where

$\mathbf{Q}$	$\in$	$\mathbb{R}^{n \times n}$	is the quadratic (also known as Hessian) matrix
$\mathbf{A}$	$\in$	$\mathbb{R}^{m \times n}$	is the constraints matrix
$\mathbf{x}$	$\in$	$\mathbb{R}^{n}$	is the vector of decision variables
$\mathbf{c}$	$\in$	$\mathbb{R}^{n}$	is the vector of linear objective function coefficients
$\mathbf{b}$	$\in$	$\mathbb{R}^{m}$	is the vector of constraints right-hand sides (RHS)
$\mathbf{l}$	$\in$	$\mathbb{R}^{n}$	is the vector of lower bounds on the decision variables
$\mathbf{u}$	$\in$	$\mathbb{R}^{n}$	is the vector of upper bounds on the decision variables

The quadratic matrix $\mathbf{Q}$ is assumed to be symmetric; that is,

$$q_{ij} = q_{ji}, \quad \forall i, j = 1, \ldots, n$$

Indeed, it is easy to show that even if $\mathbf{Q} \neq \mathbf{Q}^{\mathrm{T}}$, then the simple modification

$$\tilde{\mathbf{Q}} = \frac{1}{2}(\mathbf{Q} + \mathbf{Q}^{\mathrm{T}})$$

produces an equivalent formulation $\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} \equiv \mathbf{x}^{\mathrm{T}}\tilde{\mathbf{Q}}\mathbf{x}$; hence symmetry is assumed. When you specify a quadratic matrix, it suffices to list only lower triangular coefficients.

In addition to being symmetric, $\mathbf{Q}$ is also required to be positive semidefinite for minimization type of models:

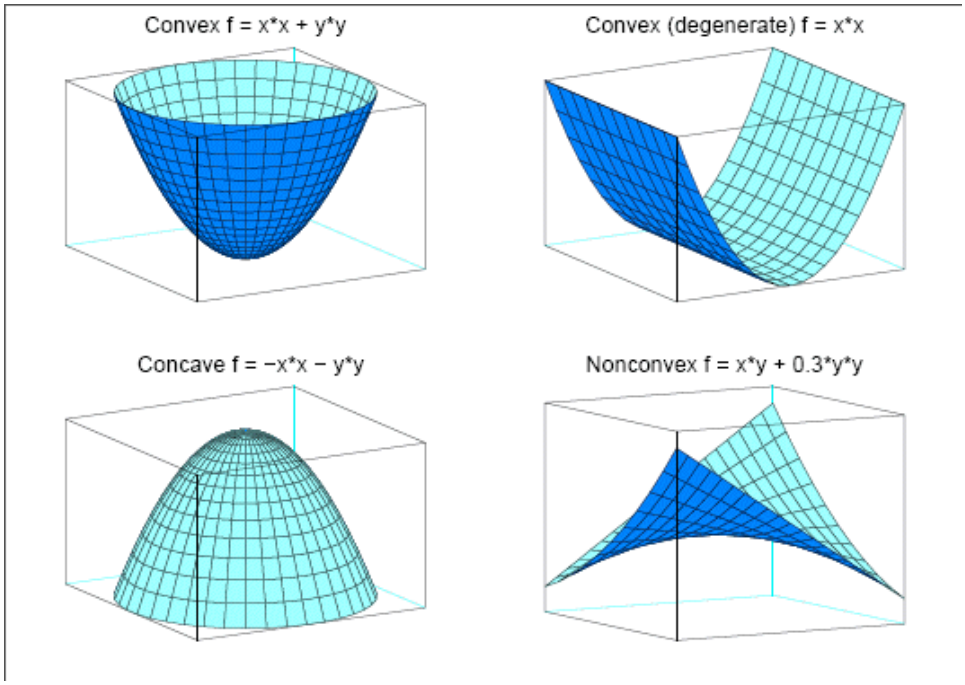
$$\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} \geq 0, \quad \forall \mathbf{x} \in \mathbb{R}^{n}$$

$\mathbf{Q}$ is required to be negative semidefinite for maximization type of models. Convexity can come as a result of a matrix-matrix multiplication

$$\mathbf{Q} = \mathbf{L}\mathbf{L}^{\mathrm{T}}$$

or as a consequence of physical laws, and so on. See Figure 11.1 for examples of convex, concave, and nonconvex objective functions.

Figure 11.1 Examples of Convex, Concave, and Nonconvex Objective Functions



The order of constraints is insignificant. Some or all components of $\mathbf{l}$ or $\mathbf{u}$ (lower and upper bounds, respectively) can be omitted.

Getting Started: QP Solver

Consider a small illustrative example. Suppose you want to minimize a two-variable quadratic function $f(x_1, x_2)$ on the nonnegative quadrant, subject to two constraints:

$$\begin{array}{rrcrcrcrcl} \min & 2x_1 & + & 3x_2 & + & x_1^2 & + & 10x_2^2 & + & 2.5x_1x_2 \\ \text{subject to} & x_1 & - & x_2 & \leq & 1 & & & & \\ & x_1 & + & 2x_2 & \geq & 100 & & & & \\ & x_1 & & & \geq & 0 & & & & \\ & & & x_2 & \geq & 0 & & & & \end{array}$$

To use the OPTMODEL procedure, it is not necessary to fit this problem into the general QP formulation mentioned in the section "Overview: QP Solver" on page 545 and to compute the corresponding parameters. However, since these parameters are closely related to the data set that is used by the OPTQP procedure and has a quadratic programming system (QPS) format, you can compute these parameters as follows. The linear objective function coefficients, vector of right-hand sides, and lower and upper bounds are identified immediately as

$$\mathbf{c} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 100 \end{bmatrix}, \quad \mathbf{l} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} +\infty \\ +\infty \end{bmatrix}$$

Carefully construct the quadratic matrix $\mathbf{Q}$. Observe that you can use symmetry to separate the main-diagonal and off-diagonal elements:

$$\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} \equiv \frac{1}{2}\sum_{i,j=1}^{n} x_i\, q_{ij}\, x_j = \frac{1}{2}\sum_{i=1}^{n} q_{ii}\, x_i^2 + \sum_{i>j} x_i\, q_{ij}\, x_j$$

The first expression

$$\frac{1}{2}\sum_{i=1}^{n} q_{ii}\, x_i^2$$

sums the main-diagonal elements. Thus, in this case you have

$$q_{11} = 2, \quad q_{22} = 20$$

Notice that the main-diagonal values are doubled in order to accommodate the 1/2 factor. Now the second term

$$\sum_{i>j} x_i\, q_{ij}\, x_j$$

sums the off-diagonal elements in the strict lower triangular part of the matrix. The only off-diagonal ($x_i\, x_j,\ i \neq j$) term in the objective function is $2.5\, x_1\, x_2$, so you have

$$q_{21} = 2.5$$

Notice that you do not need to specify the upper triangular part of the quadratic matrix.

Finally, the matrix of constraints is as follows:

$$\mathbf{A} = \begin{bmatrix} 1 & -1 \\ 1 & 2 \end{bmatrix}$$

The following OPTMODEL program formulates the preceding problem in a manner that is very close to the mathematical specification of the given problem:

```
/* getting started */
proc optmodel;
   var x1 >= 0; /* declare nonnegative variable x1 */
   var x2 >= 0; /* declare nonnegative variable x2 */

   /* objective: quadratic function f(x1, x2) */
   minimize f =
       /* the linear objective function coefficients */
       2 * x1 + 3 * x2 +

       /* quadratic <x, Qx> */
       x1 * x1 + 2.5 * x1 * x2 + 10 * x2 * x2;

   /* subject to the following constraints */
   con r1: x1 - x2 <= 1;
   con r2: x1 + 2 * x2 >= 100;
```

```
   /* specify iterative interior point algorithm (QP)
    * in the SOLVE statement */
   solve with qp;

   /* print the optimal solution */
   print x1 x2;
   save qps qpsdata;
quit;
```

The "with qp" clause in the SOLVE statement invokes the QP solver to solve the problem. The output is shown in Figure 11.2.

Figure 11.2 Summaries and Optimal Solution

The OPTMODEL Procedure

Problem Summary	
Objective Sense	Minimization
Objective Function	f
Objective Type	Quadratic
Number of Variables	2
Bounded Above	0
Bounded Below	2
Bounded Below and Above	0
Free	0
Fixed	0
Number of Constraints	2
Linear LE (<=)	1
Linear EQ (=)	0
Linear GE (>=)	1
Linear Range	0
Constraint Coefficients	4

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Figure 11.2 *continued*

Solution Summary	
Solver	QP
Algorithm	Interior Point
Objective Function	f
Solution Status	Optimal
Objective Value	15018
Primal Infeasibility	0
Dual Infeasibility	0
Bound Infeasibility	0
Duality Gap	3.633377E-16
Complementarity	0
Iterations	6
Presolve Time	0.00
Solution Time	0.03

x1	x2
34	33

In this example, the SAVE QPS statement is used to save the QP problem in the QPS-format data set qpsdata, shown in Figure 11.3. The data set is consistent with the parameters of general quadratic programming previously computed. Also, the data set can be used as input to the OPTQP procedure.

Figure 11.3 QPS-Format Data Set

Obs	FIELD1	FIELD2	FIELD3	FIELD4	FIELD5	FIELD6
1	NAME		qpsdata	.		.
2	ROWS			.		.
3	N	f		.		.
4	L	r1		.		.
5	G	r2		.		.
6	COLUMNS			.		.
7		x1	f	2.0	r1	1
8		x1	r2	1.0		.
9		x2	f	3.0	r1	-1
10		x2	r2	2.0		.
11	RHS			.		.
12		.RHS.	r1	1.0		.
13		.RHS.	r2	100.0		.
14	QSECTION			.		.
15		x1	x1	2.0		.
16		x1	x2	2.5		.
17		x2	x2	20.0		.
18	ENDATA			.		.

Syntax: QP Solver

The following statement is available in the OPTMODEL procedure:

SOLVE WITH QP < / *options* > **;**

Functional Summary

Table 11.1 summarizes the list of options available for the SOLVE WITH QP statement, classified by function.

Table 11.1 Options for the QP Solver

Description	Option
Solver Options	
Enables or disables IIS detection	IIS=
Control Options	
Specifies the frequency of printing solution progress	LOGFREQ=
Specifies the maximum number of iterations	MAXITER=
Specifies the time limit for the optimization process	MAXTIME=
Specifies the type of presolve	PRESOLVER=
Interior Point Algorithm Options	
Specifies the stopping criterion based on duality gap	STOP_DG=
Specifies the stopping criterion based on dual infeasibility	STOP_DI=
Specifies the stopping criterion based on primal infeasibility	STOP_PI=
Specifies units of CPU time or real time	TIMETYPE=

QP Solver Options

This section describes the options recognized by the QP solver. These options can be specified after a forward slash (/) in the SOLVE statement, provided that the QP solver is explicitly specified using a WITH clause.

The QP solver does not provide an intermediate solution if the solver terminates before reaching optimality.

Solver Options

IIS=*number* | *string*

specifies whether the QP solver attempts to identify a set of constraints and variables that form an irreducible infeasible set (IIS). Table 11.2 describes the valid values of the IIS= option.

Table 11.2 Values for IIS= Option

number	*string*	**Description**
0	OFF	Disables IIS detection.
1	ON	Enables IIS detection.

If an IIS is found, you can find information about the infeasibilities in the .status values of the constraints and variables. The default value of this option is OFF. See the section "Irreducible Infeasible Set" on page 557 for details about the IIS= option. See the section "Suffixes" on page 134 for details about the .status suffix.

Control Options

LOGFREQ=*k*

PRINTFREQ=*k*

specifies that the printing of the solution progress to the iteration log is to occur after every *k* iterations. The print frequency, *k*, is an integer between zero and the largest four-byte signed integer, which is $2^{31} - 1$.

The value $k = 0$ disables the printing of the progress of the solution. The default value of this option is 1.

MAXITER=*k*

specifies the maximum number of iterations. The value *k* can be any integer between one and the largest four-byte signed integer, which is $2^{31} - 1$. If you do not specify this option, the procedure does not stop based on the number of iterations performed.

MAXTIME=*t*

specifies an upper limit of *t* units of time for the optimization process, including problem generation time and solution time. The value of the TIMETYPE= option determines the type of units used. If you do not specify the MAXTIME= option, the solver does not stop based on the amount of time elapsed. The value of *t* can be any positive number; the default value is the positive number that has the largest absolute value that can be represented in your operating environment.

PRESOLVER=*number* | *string*

PRESOL=*number* | *string*

specifies one of the following presolve options:

number	*string*	**Description**
0	NONE	Disables presolver.
–1	AUTOMATIC	Applies presolver by using default setting.

You can specify the PRESOLVER= value either by a character-valued option or by an integer. The default option is AUTOMATIC.

Interior Point Algorithm Options

STOP_DG=δ

specifies the desired relative duality gap, $\delta \in$ [1E–9, 1E–4]. This is the relative difference between the primal and dual objective function values and is the primary solution quality parameter. The default value is 1E–6. See the section "Interior Point Algorithm: Overview" on page 554 for details.

STOP_DI=β

specifies the maximum allowed relative dual constraints violation, $\beta \in$ [1E–9, 1E–4]. The default value is 1E–6. See the section "Interior Point Algorithm: Overview" on page 554 for details.

STOP_PI=α

specifies the maximum allowed relative bound and primal constraints violation, $\alpha \in$ [1E–9, 1E–4]. The default value is 1E–6. See the section "Interior Point Algorithm: Overview" on page 554 for details.

TIMETYPE=*number* | *string*

specifies the units of time used by the MAXTIME= option and reported by the PRESOLVE_TIME and SOLUTION_TIME terms in the _OROPTMODEL_ macro variable. Table 11.4 describes the valid values of the TIMETYPE= option.

Table 11.4 Values for TIMETYPE= Option

number	*string*	**Description**
0	CPU	Specifies units of CPU time.
1	REAL	Specifies units of real time.

The "Optimization Statistics" table, an output of the OPTMODEL procedure if you specify PRINTLEVEL=2 in the PROC OPTMODEL statement, also includes the same time units for Presolver Time and Solver Time. The other times (such as Problem Generation Time) in the "Optimization Statistics" table are also in the same units.

The default value of the TIMETYPE= option depends on the value of the NTHREADS= option in the PERFORMANCE statement of the OPTMODEL procedure. Table 11.5 describes the detailed logic for determining the default; the first context in the table that applies determines the default value. For more information about the NTHREADS= option, see the section "PERFORMANCE Statement" on page 21 in Chapter 4, "Shared Concepts and Topics."

Table 11.5 Default Value for TIMETYPE= Option

Context	**Default**
Solver is invoked in an OPTMODEL COFOR loop	REAL
NTHREADS= value is greater than 1	REAL
NTHREADS= 1	CPU

Details: QP Solver

Interior Point Algorithm: Overview

The QP solver implements an infeasible primal-dual predictor-corrector interior point algorithm. To illustrate the algorithm and the concepts of duality and dual infeasibility, consider the following QP formulation (the primal):

$$\begin{aligned} \min \quad & \tfrac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

The corresponding dual formulation is

$$\begin{array}{rrcrcrcl} \max & -\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} & + & \mathbf{b}^{\mathrm{T}}\mathbf{y} & & & & \\ \text{subject to} & -\mathbf{Q}\mathbf{x} & + & \mathbf{A}^{\mathrm{T}}\mathbf{y} & + & \mathbf{w} & = & \mathbf{c} \\ & & & & & \mathbf{y} & \geq & \mathbf{0} \\ & & & & & \mathbf{w} & \geq & \mathbf{0} \end{array}$$

where $\mathbf{y} \in \mathbb{R}^m$ refers to the vector of dual variables and $\mathbf{w} \in \mathbb{R}^n$ refers to the vector of dual slack variables.

The dual makes an important contribution to the certificate of optimality for the primal. The primal and dual constraints combined with complementarity conditions define the first-order optimality conditions, also known as KKT (Karush-Kuhn-Tucker) conditions, which can be stated as follows where $\mathbf{e} \equiv (1, \ldots, 1)^{\mathrm{T}}$ of appropriate dimension and $\mathbf{s} \in \mathbb{R}^m$ is the vector of primal *slack* variables:

$$\begin{array}{rcll} \mathbf{A}\mathbf{x} - \mathbf{s} & = & \mathbf{b} & (\text{primal feasibility}) \\ -\mathbf{Q}\mathbf{x} + \mathbf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{w} & = & \mathbf{c} & (\text{dual feasibility}) \\ \mathbf{W}\mathbf{X}\mathbf{e} & = & \mathbf{0} & (\text{complementarity}) \\ \mathbf{S}\mathbf{Y}\mathbf{e} & = & \mathbf{0} & (\text{complementarity}) \\ \mathbf{x},\, \mathbf{y},\, \mathbf{w},\, \mathbf{s} & \geq & \mathbf{0} & \end{array}$$

NOTE: Slack variables (the *s* vector) are automatically introduced by the solver when necessary; it is therefore recommended that you not introduce any slack variables explicitly. This enables the solver to handle slack variables much more efficiently.

The letters $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{W}$, and $\mathbf{S}$ denote matrices with corresponding x, y, w, and s on the main diagonal and zero elsewhere, as in the following example:

$$\mathbf{X} \equiv \begin{bmatrix} x_1 & 0 & \cdots & 0 \\ 0 & x_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_n \end{bmatrix}$$

If $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{w}^*, \mathbf{s}^*)$ is a solution of the previously defined system of equations that represent the KKT conditions, then $\mathbf{x}^*$ is also an optimal solution to the original QP model.

At each iteration the interior point algorithm solves a large, sparse system of linear equations,

$$\begin{bmatrix} \mathbf{Y}^{-1}\mathbf{S} & \mathbf{A} \\ \mathbf{A}^{\mathrm{T}} & -\mathbf{Q} - \mathbf{X}^{-1}\mathbf{W} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y} \\ \Delta\mathbf{x} \end{bmatrix} = \begin{bmatrix} \Xi \\ \Theta \end{bmatrix}$$

where $\Delta\mathbf{x}$ and $\Delta\mathbf{y}$ denote the vector of *search directions* in the primal and dual spaces, respectively, and Θ and Ξ constitute the vector of the right-hand sides.

The preceding system is known as the reduced KKT system. The QP solver uses a preconditioned quasi-minimum residual algorithm to solve this system of equations efficiently.

An important feature of the interior point algorithm is that it takes full advantage of the sparsity in the constraint and quadratic matrices, thereby enabling it to efficiently solve large-scale quadratic programs.

The interior point algorithm works simultaneously in the primal and dual spaces. It attains optimality when both primal and dual feasibility are achieved and when complementarity conditions hold. Therefore, it is of interest to observe the following four measures where $\|v\|_2$ is the Euclidean norm of the vector v:

- relative primal infeasibility measure α:

$$\alpha = \frac{\|\mathbf{Ax} - \mathbf{b} - \mathbf{s}\|_2}{\|\mathbf{b}\|_2 + 1}$$

- relative dual infeasibility measure β:

$$\beta = \frac{\|\mathbf{Qx} + \mathbf{c} - \mathbf{A}^{\mathrm{T}}\mathbf{y} - \mathbf{w}\|_2}{\|\mathbf{c}\|_2 + 1}$$

- relative duality gap δ:

$$\delta = \frac{|\mathbf{x}^{\mathrm{T}}\mathbf{Qx} + \mathbf{c}^{\mathrm{T}}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{y}|}{|\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Qx} + \mathbf{c}^{\mathrm{T}}\mathbf{x}| + 1}$$

- absolute complementarity γ:

$$\gamma = \sum_{i=1}^{n} x_i w_i + \sum_{i=1}^{m} y_i s_i$$

These measures are displayed in the iteration log.

Parallel Processing

The interior point algorithm can be run in single-machine mode (in single-machine mode, the computation is executed by multiple threads on a single computer). You can specify options for parallel processing in the PERFORMANCE statement, which is documented in the section "PERFORMANCE Statement" on page 21 in Chapter 4, "Shared Concepts and Topics."

Iteration Log

The following information is displayed in the iteration log:

Iter	indicates the iteration number.
Complement	indicates the (absolute) complementarity.
Duality Gap	indicates the (relative) duality gap.
Primal Infeas	indicates the (relative) primal infeasibility measure.
Bound Infeas	indicates the (relative) bound infeasibility measure.
Dual Infeas	indicates the (relative) dual infeasibility measure.
Time	indicates the time elapsed (in seconds).

If the sequence of solutions converges to an optimal solution of the problem, you should see all columns in the iteration log converge to zero or very close to zero. Nonconvergence can be the result of insufficient iterations being performed to reach optimality. In this case, you might need to increase the value that you specify in the MAXITER= or MAXTIME= option. If the complementarity or the duality gap does not converge, the problem might be infeasible or unbounded. If the infeasibility columns do not converge, the problem might be infeasible.

Problem Statistics

Optimizers can encounter difficulty when solving poorly formulated models. Information about data magnitude provides a simple gauge to determine how well a model is formulated. For example, a model whose constraint matrix contains one very large entry (on the order of 10^9) can cause difficulty when the remaining entries are single-digit numbers. The PRINTLEVEL=2 option in the OPTMODEL procedure causes the ODS table ProblemStatistics to be generated when the QP solver is called. This table provides basic data magnitude information that enables you to improve the formulation of your models.

The example output in Figure 11.4 demonstrates the contents of the ODS table ProblemStatistics.

Figure 11.4 ODS Table ProblemStatistics

The OPTMODEL Procedure

Problem Statistics	
Number of Constraint Matrix Nonzeros	4
Maximum Constraint Matrix Coefficient	2
Minimum Constraint Matrix Coefficient	1
Average Constraint Matrix Coefficient	1.25
Number of Linear Objective Nonzeros	2
Maximum Linear Objective Coefficient	3
Minimum Linear Objective Coefficient	2
Average Linear Objective Coefficient	2.5
Number of Lower Triangular Hessian Nonzeros	1
Number of Diagonal Hessian Nonzeros	2
Maximum Hessian Coefficient	20
Minimum Hessian Coefficient	2
Average Hessian Coefficient	6.75
Number of RHS Nonzeros	2
Maximum RHS	100
Minimum RHS	1
Average RHS	50.5
Maximum Number of Nonzeros per Column	2
Minimum Number of Nonzeros per Column	2
Average Number of Nonzeros per Column	2
Maximum Number of Nonzeros per Row	2
Minimum Number of Nonzeros per Row	2
Average Number of Nonzeros per Row	2

Irreducible Infeasible Set

For a quadratic programming problem, an irreducible infeasible set (IIS) is an infeasible subset of constraints and variable bounds that becomes feasible if any single constraint or variable bound is removed. It is possible to have more than one IIS in an infeasible QP. Identifying an IIS can help isolate the structural infeasibility in a QP. The IIS=ON option directs the QP solver to search for an IIS in a specified QP.

Whether a quadratic programming problem is feasible or infeasible is determined by its constraints and variable bounds, which have nothing to do with its objective function. When you specify the IIS=ON option, the QP solver treats this problem as a linear programming problem by ignoring its objective function. Then finding IIS is the same as what the LP solver does with the IIS=ON option. See the section "Irreducible Infeasible Set" on page 274 in Chapter 7, "The Linear Programming Solver," for more information about the irreducible infeasible set.

Macro Variable _OROPTMODEL_

The OPTMODEL procedure always creates and initializes a SAS macro called _OROPTMODEL_. This variable contains a character string. After each PROC OROPTMODEL run, you can examine this macro by specifying `%put &_OROPTMODEL_;` and check the execution of the most recently invoked solver from the value of the macro variable. The various terms of the variable after the QP solver is called are interpreted as follows.

STATUS
indicates the solver status at termination. It can take one of the following values:

OK	The solver terminated normally.
SYNTAX_ERROR	Incorrect syntax was used.
DATA_ERROR	The input data were inconsistent.
OUT_OF_MEMORY	Insufficient memory was allocated to the procedure.
IO_ERROR	A problem occurred in reading or writing data.
SEMANTIC_ERROR	An evaluation error, such as an invalid operand type, occurred.
ERROR	The status cannot be classified into any of the preceding categories.

ALGORITHM
indicates the algorithm that produced the solution data in the macro variable. This term only appears when STATUS=OK. It can take the following value:

IP	The interior point algorithm produced the solution data.

SOLUTION_STATUS
indicates the solution status at termination. It can take one of the following values:

OPTIMAL	The solution is optimal.
CONDITIONAL_OPTIMAL	The solution is optimal, but some infeasibilities (primal, dual or bound) exceed tolerances due to scaling or pre-processing.
INFEASIBLE	The problem is infeasible.
UNBOUNDED	The problem is unbounded.
INFEASIBLE_OR_UNBOUNDED	The problem is infeasible or unbounded.
BAD_PROBLEM_TYPE	The problem type is unsupported by the solver.
ITERATION_LIMIT_REACHED	The maximum allowable number of iterations was reached.
TIME_LIMIT_REACHED	The solver reached its execution time limit.
FUNCTION_CALL_LIMIT_REACHED	The solver reached its limit on function evaluations.
INTERRUPTED	The solver was interrupted externally.
FAILED	The solver failed to converge, possibly due to numerical issues.

OBJECTIVE
indicates the objective value obtained by the solver at termination.

PRIMAL_INFEASIBILITY
indicates the (relative) infeasibility of the primal constraints at the solution. See the section "Interior Point Algorithm: Overview" on page 554 for details.

DUAL_INFEASIBILITY
indicates the (relative) infeasibility of the dual constraints at the solution. See the section "Interior Point Algorithm: Overview" on page 554 for details.

BOUND_INFEASIBILITY
indicates the (relative) violation by the solution of the lower or upper bounds (or both). See the section "Interior Point Algorithm: Overview" on page 554 for details.

DUALITY_GAP
indicates the (relative) duality gap. See the section "Interior Point Algorithm: Overview" on page 554 for details.

COMPLEMENTARITY
indicates the (absolute) complementarity at the solution. See the section "Interior Point Algorithm: Overview" on page 554 for details.

ITERATIONS
indicates the number of iterations required to solve the problem.

PRESOLVE_TIME
indicates the time taken for preprocessing (in seconds).

SOLUTION_TIME
indicates the time (in seconds) taken to solve the problem, including preprocessing time.

NOTE: The time that is reported in PRESOLVE_TIME and SOLUTION_TIME is either CPU time or real time. The type is determined by the TIMETYPE= option.

Examples: QP Solver

This section presents examples that illustrate the use of the OPTMODEL procedure to solve quadratic programming problems. Example 11.1 illustrates how to model a linear least squares problem and solve it by using PROC OPTMODEL. Example 11.2 and Example 11.3 show in detail how to model the portfolio optimization and selection problems.

Example 11.1: Linear Least Squares Problem

The linear least squares problem arises in the context of determining a solution to an overdetermined set of linear equations. In practice, these equations could arise in data fitting and estimation problems. An overdetermined system of linear equations can be defined as

$$\mathbf{Ax} = \mathbf{b}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, and $m > n$. Since this system usually does not have a solution, you need to be satisfied with some sort of approximate solution. The most widely used approximation is the least squares solution, which minimizes $\|\mathbf{Ax} - \mathbf{b}\|_2^2$.

This problem is called a least squares problem for the following reason. Let $\mathbf{A}$, $\mathbf{x}$, and $\mathbf{b}$ be defined as previously. Let $k_i(x)$ be the kth component of the vector $\mathbf{Ax} - \mathbf{b}$:

$$k_i(x) = a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - b_i,\ i = 1, 2, \ldots, m$$

By definition of the Euclidean norm, the objective function can be expressed as follows:

$$\|\mathbf{Ax} - \mathbf{b}\|_2^2 = \sum_{i=1}^{m} k_i(x)^2$$

Therefore, the function you minimize is the sum of squares of m terms $k_i(x)$; hence the term least squares. The following example is an illustration of the *linear* least squares problem; that is, each of the terms k_i is a linear function of x .

Consider the following least squares problem defined by

$$\mathbf{A} = \begin{bmatrix} 4 & 0 \\ -1 & 1 \\ 3 & 2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

This translates to the following set of linear equations:

$$4x_1 = 1, \quad -x_1 + x_2 = 0, \quad 3x_1 + 2x_2 = 1$$

The corresponding least squares problem is:

$$\text{minimize} \quad (4x_1 - 1)^2 + (-x_1 + x_2)^2 + (3x_1 + 2x_2 - 1)^2$$

The preceding objective function can be expanded to:

$$\text{minimize} \quad 26x_1^2 + 5x_2^2 + 10x_1x_2 - 14x_1 - 4x_2 + 2$$

In addition, you impose the following constraint so that the equation $3x_1 + 2x_2 = 1$ is satisfied within a tolerance of 0.1:

$$0.9 \leq 3x_1 + 2x_2 \leq 1.1$$

You can use the following SAS statements to solve the least squares problem:

```
/* example 1: linear least-squares problem */
proc optmodel;
   var x1; /* declare free (no explicit bounds) variable x1 */
   var x2; /* declare free (no explicit bounds) variable x2 */
   /* declare slack variable for ranged constraint */
   var w >= 0 <= 0.2;

   /* objective function: minimize is the sum of squares */
   minimize f = 26 * x1 * x1 + 5 * x2 * x2 + 10 * x1 * x2
       - 14 * x1 - 4 * x2 + 2;

   /* subject to the following constraint */
   con L: 3 * x1 + 2 * x2 - w = 0.9;

   solve with qp;

   /* print the optimal solution */
   print x1 x2;
quit;
```

The output is shown in Output 11.1.1.

Output 11.1.1 Summaries and Optimal Solution

The OPTMODEL Procedure

Problem Summary	
Objective Sense	Minimization
Objective Function	f
Objective Type	Quadratic
Number of Variables	3
Bounded Above	0
Bounded Below	0
Bounded Below and Above	1
Free	2
Fixed	0
Number of Constraints	1
Linear LE (<=)	0
Linear EQ (=)	1
Linear GE (>=)	0
Linear Range	0
Constraint Coefficients	3

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Output 11.1.1 *continued*

Solution Summary	
Solver	QP
Algorithm	Interior Point
Objective Function	f
Solution Status	Optimal
Objective Value	0.0095238095
Primal Infeasibility	1.926295E-12
Dual Infeasibility	1.3056209E-9
Bound Infeasibility	0
Duality Gap	4.3533121E-8
Complementarity	1.3603398E-7
Iterations	3
Presolve Time	0.00
Solution Time	0.01

x1	x2
0.2381	0.1619

Example 11.2: Portfolio Optimization

Consider a portfolio optimization example. The two competing goals of investment are (1) long-term growth of capital and (2) low risk. A good portfolio grows steadily without wild fluctuations in value. The Markowitz model is an optimization model for balancing the return and risk of a portfolio. The decision variables are the amounts invested in each asset. The objective is to minimize the variance of the portfolio's total return, subject to the constraints that (1) the expected growth of the portfolio reaches at least some target level and (2) you do not invest more capital than you have.

Let $x_1, \ldots, x_n$ be the amount invested in each asset, $\mathcal{B}$ be the amount of capital you have, $\mathbf{R}$ be the random vector of asset returns over some period, and $\mathbf{r}$ be the expected value of $\mathbf{R}$. Let G be the minimum growth you hope to obtain, and $\mathcal{C}$ be the covariance matrix of $\mathbf{R}$. The objective function is $\mathrm{Var}\left(\sum_{i=1}^{n} x_i R_i\right)$, which can be equivalently denoted as $\mathbf{x}^{\mathrm{T}}\mathcal{C}\mathbf{x}$.

Assume, for example, $n = 4$. Let $\mathcal{B} = 10{,}000$, $G = 1{,}000$, $\mathbf{r} = [0.05, -0.2, 0.15, 0.30]$, and

$$\mathcal{C} = \begin{bmatrix} 0.08 & -0.05 & -0.05 & -0.05 \\ -0.05 & 0.16 & -0.02 & -0.02 \\ -0.05 & -0.02 & 0.35 & 0.06 \\ -0.05 & -0.02 & 0.06 & 0.35 \end{bmatrix}$$

The QP formulation can be written as:

$$\begin{aligned}\min \quad & 0.08x_1^2 - 0.1x_1x_2 - 0.1x_1x_3 - 0.1x_1x_4 + 0.16x_2^2 \\ & -0.04x_2x_3 - 0.04x_2x_4 + 0.35x_3^2 + 0.12x_3x_4 + 0.35x_4^2\end{aligned}$$

$$\begin{array}{lrcl}\text{subject to} & & & \\ \text{(budget)} & x_1 + x_2 + x_3 + x_4 & \leq & 10000 \\ \text{(growth)} & 0.05x_1 - 0.2x_2 + 0.15x_3 + 0.30x_4 & \geq & 1000 \\ & x_1,\ x_2,\ x_3,\ x_4 & \geq & 0\end{array}$$

Use the following SAS statements to solve the problem:

```
/* example 2: portfolio optimization */
proc optmodel;
   /* let x1, x2, x3, x4 be the amount invested in each asset */
   var x{1..4} >= 0;

   num coeff{1..4, 1..4} = [0.08 -.05 -.05 -.05
                            -.05 0.16 -.02 -.02
                            -.05 -.02 0.35 0.06
                            -.05 -.02 0.06 0.35];
   num r{1..4}=[0.05 -.20 0.15 0.30];

   /* minimize the variance of the portfolio's total return */
   minimize f = sum{i in 1..4, j in 1..4}coeff[i,j]*x[i]*x[j];

   /* subject to the following constraints */
   con BUDGET: sum{i in 1..4}x[i] <= 10000;
   con GROWTH: sum{i in 1..4}r[i]*x[i] >= 1000;

   solve with qp;

   /* print the optimal solution */
   print x;
```

The summaries and the optimal solution are shown in Output 11.2.1.

Output 11.2.1 Portfolio Optimization

The OPTMODEL Procedure

Problem Summary	
Objective Sense	Minimization
Objective Function	f
Objective Type	Quadratic
Number of Variables	4
Bounded Above	0
Bounded Below	4
Bounded Below and Above	0
Free	0
Fixed	0
Number of Constraints	2
Linear LE (<=)	1
Linear EQ (=)	0
Linear GE (>=)	1
Linear Range	0
Constraint Coefficients	8

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Solution Summary	
Solver	QP
Algorithm	Interior Point
Objective Function	f
Solution Status	Optimal
Objective Value	2232313.4432
Primal Infeasibility	0
Dual Infeasibility	8.038873E-14
Bound Infeasibility	0
Duality Gap	4.172004E-16
Complementarity	0
Iterations	7
Presolve Time	0.00
Solution Time	0.02

[1]	x
1	3452.9
2	0.0
3	1068.8
4	2223.5

Thus, the minimum variance portfolio that earns an expected return of at least 10% is $x_1 = 3{,}452$, $x_2 = 0$, $x_3 = 1{,}068$, $x_4 = 2,223$. Asset 2 gets nothing because its expected return is -20% and its covariance with the other assets is not sufficiently negative for it to bring any diversification benefits. What if you drop the nonnegativity assumption?

Financially, that means you are allowed to short-sell—that is, sell low-mean-return assets and use the proceeds to invest in high-mean-return assets. In other words, you put a negative portfolio weight in low-mean assets and "more than 100%" in high-mean assets.

To solve the portfolio optimization problem with the short-sale option, continue to submit the following SAS statements:

```
    /* example 2: portfolio optimization with short-sale option */
    /* dropping nonnegativity assumption */
    for {i in 1..4} x[i].lb=-x[i].ub;

    solve with qp;

    /* print the optimal solution */
    print x;
quit;
```

You can see in the optimal solution displayed in Output 11.2.2 that the decision variable x_2, denoting Asset 2, is equal to $-1{,}563.61$, which means short sale of that asset.

Output 11.2.2 Portfolio Optimization with Short-Sale Option

The OPTMODEL Procedure

Solution Summary	
Solver	QP
Algorithm	Interior Point
Objective Function	f
Solution Status	Optimal
Objective Value	1907122.2254
Primal Infeasibility	0
Dual Infeasibility	2.755587E-13
Bound Infeasibility	0
Duality Gap	2.441695E-16
Complementarity	0
Iterations	6
Presolve Time	0.00
Solution Time	0.02

[1]	x
1	1684.35
2	-1563.61
3	682.51
4	1668.95

Example 11.3: Portfolio Selection with Transactions

Consider a portfolio selection problem with a slight modification. You are now required to take into account the current position and transaction costs associated with buying and selling assets. The objective is to find the minimum variance portfolio. In order to understand the scenario better, consider the following data.

You are given three assets. The current holding of the three assets is denoted by the vector **c** = [200, 300, 500], the amount of asset bought and sold is denoted by b_i and s_i, respectively, and the net investment in each asset is denoted by x_i and is defined by the following relation:

$$x_i - b_i + s_i = c_i, \quad i = 1, 2, 3$$

Suppose that you pay a transaction fee of 0.01 every time you buy or sell. Let the covariance matrix $\mathcal{C}$ be defined as

$$\mathcal{C} = \begin{bmatrix} 0.027489 & -0.00874 & -0.00015 \\ -0.00874 & 0.109449 & -0.00012 \\ -0.00015 & -0.00012 & 0.000766 \end{bmatrix}$$

Assume that you hope to obtain at least 12% growth. Let **r** = [1.109048, 1.169048, 1.074286] be the vector of expected return on the three assets, and let $\mathcal{B}$=1000 be the available funds. Mathematically, this problem can be written in the following manner:

$$\begin{aligned} \min \quad & 0.027489x_1^2 - 0.01748x_1x_2 - 0.0003x_1x_3 + 0.109449x_2^2 \\ & -0.00024x_2x_3 + 0.000766x_3^2 \end{aligned}$$

subject to

$$\begin{aligned} \text{(return)} \quad & \sum_{i=1}^{3} r_i x_i \geq 1.12\mathcal{B} \\ \text{(budget)} \quad & \sum_{i=1}^{3} x_i + \sum_{i=1}^{3} 0.01(b_i + s_i) = \mathcal{B} \\ \text{(balance)} \quad & x_i - b_i + s_i = c_i, \quad i = 1, 2, 3 \\ & x_i, b_i, s_i \geq 0, \quad i = 1, 2, 3 \end{aligned}$$

The problem can be solved by the following SAS statements:

```
/* example 3: portfolio selection with transactions */
proc optmodel;
   /* let x1, x2, x3 be the amount invested in each asset */
   var x{1..3} >= 0;
   /* let b1, b2, b3 be the amount of asset bought */
   var b{1..3} >= 0;
   /* let s1, s2, s3 be the amount of asset sold */
   var s{1..3} >= 0;

   /* current holdings */
   num c{1..3}=[ 200 300 500];
   /* covariance matrix */
   num coeff{1..3, 1..3} = [0.027489  -.008740  -.000150
                            -.008740  0.109449  -.000120
                            -.000150  -.000120  0.000766];
   /* returns */
   num r{1..3}=[1.109048 1.169048 1.074286];

   /* minimize the variance of the portfolio's total return */
   minimize f = sum{i in 1..3, j in 1..3}coeff[i,j]*x[i]*x[j];

   /* subject to the following constraints */
   con BUDGET: sum{i in 1..3}(x[i]+.01*b[i]+.01*s[i]) <= 1000;
   con RETURN: sum{i in 1..3}r[i]*x[i] >= 1120;
   con BALANC{i in 1..3}: x[i]-b[i]+s[i]=c[i];

   solve with qp;

   /* print the optimal solution */
   print x;
quit;
```

The output is displayed in Output 11.3.1.

Output 11.3.1 Portfolio Selection with Transactions

The OPTMODEL Procedure

Problem Summary	
Objective Sense	Minimization
Objective Function	f
Objective Type	Quadratic
Number of Variables	9
Bounded Above	0
Bounded Below	9
Bounded Below and Above	0
Free	0
Fixed	0
Number of Constraints	5
Linear LE (<=)	1
Linear EQ (=)	3
Linear GE (>=)	1
Linear Range	0
Constraint Coefficients	21

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Solution Summary	
Solver	QP
Algorithm	Interior Point
Objective Function	f
Solution Status	Optimal
Objective Value	19560.725753
Primal Infeasibility	7.000061E-17
Dual Infeasibility	0
Bound Infeasibility	0
Duality Gap	2.994187E-14
Complementarity	0
Iterations	11
Presolve Time	0.00
Solution Time	0.02

[1]	x
1	397.58
2	406.12
3	190.17

References

Freund, R. W. (1991). "On Polynomial Preconditioning and Asymptotic Convergence Factors for Indefinite Hermitian Matrices." *Linear Algebra and Its Applications* 154–156:259–288.

Freund, R. W., and Jarre, F. (1997). "A QMR-Based Interior Point Algorithm for Solving Linear Programs." *Mathematical Programming* 76:183–210.

Freund, R. W., and Nachtigal, N. M. (1996). "QMRPACK: A Package of QMR Algorithms." *ACM Transactions on Mathematical Software* 22:46–77.

Vanderbei, R. J. (1999). "LOQO: An Interior Point Code for Quadratic Programming." *Optimization Methods and Software* 11:451–484.

Wright, S. J. (1997). *Primal-Dual Interior-Point Methods*. Philadelphia: SIAM.

Subject Index

Syntax Index