International Journal of Management Science
Vol 13, No 1, May 2007

Solving Robust EOQ Model Using Genetic Algorithm*

Sungmook Lim**
School of Business Administration, Korea University
Jochiwon-Eup, Yeongi-Gun, Chungnam, 339-700, Republic of Korea

(Received Apr. 2006; Revised Sep. 2006; Accepted Dec. 2007)

ABSTRACT

We consider a (worst-case) robust optimization version of the Economic Order Quantity (EOQ) model. Order setup costs and inventory carrying costs are assumed to have uncertainty in their values, and the uncertainty description of the two parameters is supposed to be given by an ellipsoidal representation. A genetic algorithm combined with Monte Carlo simulation is proposed to approximate the ellipsoidal representation. The objective function of the model under ellipsoidal uncertainty description is derived, and the resulting problem is solved by another genetic algorithm. Computational test results are presented to show the performance of the proposed method.

Keywords: EOQ Model, Uncertainty, Robust Optimization, Genetic Algorithm, Monte Carlo Simulation

1. Introduction

The Economic Order Quantity (EOQ) model is a basic inventory management model introduced in 1951 to decide on an optimal order quantity or lot size, provided that the three parameters of average demand rate, order setup cost, and inventory carrying cost are given. The model has been widely utilized due to its simplicity as well as adaptability to various situations.

In the traditional *deterministic* EOQ model, an optimal order quantity Q_d is sought to minimize the average cost rate C_d which is the sum of both the total of inventory carrying costs and the total of order setup costs,

---

* This work was supported by the Post-doctoral Fellowship Program of Korea Research Foundation (KRF).
** E-mail: sungmook@korea.ac.kr

$$C_d = S_d \frac{D}{Q_d} + h_d \frac{Q_d}{2}, \tag{1}$$

where D is the average demand rate, S_d is the setup cost per order, and h_d is the inventory carrying cost per item per unit of time, with all the three parameters being deterministic. The average cost rate is minimized at the order quantity,

$$Q_d^* = \sqrt{\frac{2S_d D}{h_d}}, \tag{2}$$

and in this case the minimal cost is

$$C_d^* = \sqrt{2S_d D h_d}. \tag{3}$$

It is assumed the values of the parameters are known exactly in the model. In real management environment, however, it is more natural to consider situations where the parameters have some uncertainty in their values. Values for the parameters may have fluctuation out of control under uncertain management environment, or involve some estimation errors. In this study, we consider the case where the parameters S and h have uncertainty in their values. The parameter vector (S, h) is modeled as a random variable vector with a known probabilistic distribution, and the statistical properties of each parameter are given as

$$E(S) = \mu_S, \quad E(h) = \mu_h,$$
$$V(S) = \sigma_S^2, \quad V(h) = \sigma_h^2,$$
$$Cov(S, h) = \sigma_{S,h},$$

where E, V, and Cov stand for expectation, variance, and covariance of given random variables, respectively.

For situations involving estimation errors, many studies have been done about the sensitivity analysis of average cost rate to errors in parameter estimation. Especially, Lowe and Schwartz [8] considered the case where the value ranges of parameters are provided, and deployed a minimax criterion to decide on a worst-case optimal order quantity. The minimax criterion under uncertainty of parameter values is also the basis of (worst-case) robust optimization.

Robust optimization is concerned with optimization problems for which the problem data is not specified exactly [2]. Consider an optimization problem of the form

$$(P) \qquad \begin{array}{ll} \min & f_0(x,\ u) \\ s.t. & f_i(x,\ u) \le 0,\ \ i = 1, \cdots, m, \end{array}$$

where $x \in R^n$ is the decision variable, $u \in R^k$ is the uncertain data(parameter) element of the problem, the function $f_0 : R^n \times R^k \to R$ is the objective function, and the functions $f_i : R^n \times R^k \to R$, $i = 1, \cdots, m$, are the constraint functions.

The idea in robust optimization is to explicitly incorporate a model of data uncertainty in the formulation of an optimization problem, and there are mainly two approaches to robust optimization, stochastic versus worst-case robust optimization, which differ in the way of dealing with uncertainty of parameter values [3].

In the stochastic robust optimization approach, the uncertain parameter vector is modeled as a random variable with a known distribution, and we work with the expected values of the objective and constraint functions as follows:

$$\begin{array}{ll} \min & E_u(f_0(x,\ u)) \\ s.t. & E_u(f_i(x,\ u)) \le 0,\ \ i = 1, \cdots, m, \end{array}$$

where E_u means expectation with respect to the random variable u.

In the worst-case robust optimization approach proposed by Ben-Tal and Nemirovski [1, 2], they assume a "decision environment" which is characterized by

(i) A crude knowledge of the data: it may be partly or fully "uncertain," and all that is known about the data vector u is that it belongs to a given *uncertainty set* $U \in R^k$,
(ii) The constraints $f_i(x,\ u) \le 0$ *must* be satisfied, whatever the actual realization of $u \in U$ is.

In view of (i) and (ii) a vector x is called feasible solution to the uncertain optimization problem *(P)* if x satisfies all possible realizations of the constraints:

$$f_i(x,\ u) \le 0,\ i = 1, \cdots, m,\ \forall u \in U. \qquad (4)$$

The notion of an optimal solution to the uncertain problem *(P)* is defined in the same manner: such a solution must give the best possible *guaranteed value*

$\sup_{u \in U} f_0(x, u)$ of the original objective under constraints (4), i.e., it should be an optimal solution to the following "certain" optimization problem.

$$(P^*) \qquad \min\left\{\sup_{u \in U} f_0(x, u) : f_i(x, u) \le 0,\ i = 1, \cdots m,\ \forall u \in U\right\}.$$

Ben-Tal and Nemirovski [2] argued that the "decision environment" is well motivated by several reasons. They also showed that if the uncertain set is represented as an ellipsoid, then for some of the most important generic convex optimization problems (linear programming, quadratically constrained programming, semidefinite programming and others) the corresponding robust optimization problem is a tractable problem which can be solved by efficient algorithms such as interior point methods. Notice that an EOQ problem is essentially a geometric programming problem, and it is not known whether a general robust geometric programming can be reformulated as a tractable optimization problem that interior point or other algorithms can efficiently solve.

When we are to formulate a problem involving uncertainty via worst-case robust optimization, it is quite natural to think of how the uncertainty set can be effectively constructed. There is, however, no essential output from the field of robust optimization about practically constructing meaningful uncertainty sets, whereas there is much literature on solving worst-case robust optimization problems.

In this study, we investigate the EOQ model with uncertainty in parameters using mainly the framework of worst-case robust optimization proposed by Ben-Tal and Nemirovski. We also propose a method of describing the uncertainty set. As for the method, a genetic algorithm combined with Monte-Carlo simulation is developed and implemented with the MATLAB [9]. Computational test results showing the performance of our approach are provided.

2. Robust Optimization for the EOQ Model

Before we consider the two types of robust optimization formulation for the EOQ problem in this section, we assume again the setup cost S and the inventory carrying cost h have uncertainty in their values with statistical properties as mentioned in the previous section. It is easy to solve the stochastic robust optimization version of the EOQ model because the objective function of the model (1) is linear in (S,h). The optimal order quantity Q_{sr}^* is obtained by replacing S and h by μ_S and μ_h, respectively, in (2) to be

$$Q_{sr}^* = \sqrt{\frac{2\mu_S D}{\mu_h}}. \tag{5}$$

In order to build the worst-case robust optimization version of the EOQ model, we assume that the uncertainty set of the parameter vector (S,h) is represented by a set F. Then, the problem is formulated in the sense of the worst-case robust optimization as

$$\min_{Q>0} \sup_{(S,h)\in F} \frac{SD}{Q}+\frac{hQ}{2}. \tag{6}$$

We also assume the set F is described as an ellipsoid for the purpose of making the problem tractable, and we explore in the next section how to describe the uncertainty of the parameter vector (S,h) by an ellipsoidal representation.

If we further restrict the sign of (S,h) to be positive, which is quite natural and necessary restriction in the EOQ model, we get

$$\min_{Q>0} \sup_{(S,h)\in F^*} \frac{SD}{Q}+\frac{hQ}{2}, \tag{7}$$

where $F^* = F \cap \{(S,h): S>0,\ h>0\}$. Although the problem (7) is convex since the function $SD/Q+hQ/2$ is convex in for each (S,h), it is difficult to obtain an analytical form of $\sup_{(S,h)\in F^*} SD/Q+hQ/2$ with the uncertainty set F^* being in part polyhedral. It is, however, obvious that the supremum is attained at a point with $(S,h)>0$. Therefore, we can conclude the problem (7) is equivalent to (6), and we are allowed to deal with the problem (6), not with (7).

Let the set F be given as

$$F = \{ \begin{pmatrix} \mu_S \\ \mu_h \end{pmatrix} + P\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \mid \|u\| \le 1 \}, \tag{8}$$

where $P \in R^{2\times 2}$ is a nonsingular matrix and the norm for $u = (u_1, u_2)^T$ is the Euclidian norm. If we, for brevity, set $\xi = (S,h)^T$ and $x = (D/Q, Q/2)^T$, then we can rewrite the inner part of (6) as

$$\begin{aligned} \sup_{\xi \in F} \xi^T x &= \sup_u \{ x^T(\mu_\xi + Pu) \mid \|u\| \le 1 \} \\ &= \sup_u \{ x^T \mu_\xi + x^T Pu \mid \|u\| \le 1 \} \\ &= \mu_\xi^T x + \|P^T x\|, \end{aligned} \tag{9}$$

where $\mu_\xi = (\mu_S, \mu_h)^T$.

Using this result, we can rewrite (6) as

$$\begin{aligned} \min_{x>0} \mu_\xi^T x + \|P^T x\| &= \min_{Q>0} \frac{\mu_S D}{Q} + \frac{\mu_h Q}{2} + \|P^T (D/Q, Q/2)^T\| \\ &= \min_{Q>0} \frac{\mu_S D}{Q} + \frac{\mu_h Q}{2} + (\alpha \frac{D^2}{Q^2} + \frac{1}{4}\beta Q^2 + \gamma D)^{\frac{1}{2}}, \end{aligned} \tag{10}$$

where

$$\begin{aligned} P &= \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix}, \\ \alpha &= p_{11}^2 + p_{12}^2, \\ \beta &= p_{21}^2 + p_{22}^2, \\ \gamma &= p_{11}p_{21} + p_{12}p_{22}, \end{aligned}$$

and hereafter we will refer to the function of Q to be minimized in (10) as $C(Q)$.

When we find the first and second derivatives, (11) and (12), of $C(Q)$,

$$C'(Q) = -\frac{\mu_S D}{Q^2} + \frac{\mu_h}{2} + \frac{1}{2}\left(-4\frac{\alpha D^2}{Q^3} + \beta Q\right) \frac{1}{\sqrt{4\frac{\alpha D^2}{Q^2} + \beta Q^2 + 4\gamma D}}, \tag{11}$$

$$\begin{aligned} C''(Q) = {} & 2\frac{\mu_S D}{Q^3} - \frac{1}{2}\left(-4\frac{\alpha D^2}{Q^3} + \beta Q\right)^2 \left(4\frac{\alpha D^2}{Q^2} + \beta Q^2 + 4\gamma D\right)^{-\frac{3}{2}} \\ & + \left(6\frac{\alpha D^2}{Q^4} + \frac{1}{2}\beta\right) \frac{1}{\sqrt{4\frac{\alpha D^2}{Q^2} + \beta Q^2 + 4\gamma D}}, \end{aligned} \tag{12}$$

to explore curvature of the function, we don't get any useful information about easiness of minimization, that is convexity. For example, if P, μ_S, μ_h and D are given by

$$P = \begin{pmatrix} 5 & -4 \\ -4 & 6 \end{pmatrix}, \quad \mu_S = \mu_h = 1, \quad D = 100,$$

the plots of $C(Q)$, $C'(Q)$ and $C''(Q)$ are in Figure 1. Specifically, the two plots of $C''(Q)$ which differ in the range of Q are presented at the bottom of Figure 1, and it is found that $C''(Q)$ assumes negative values at some points of Q between 10 and 20. Therefore, $C(Q)$ is not convex in general.

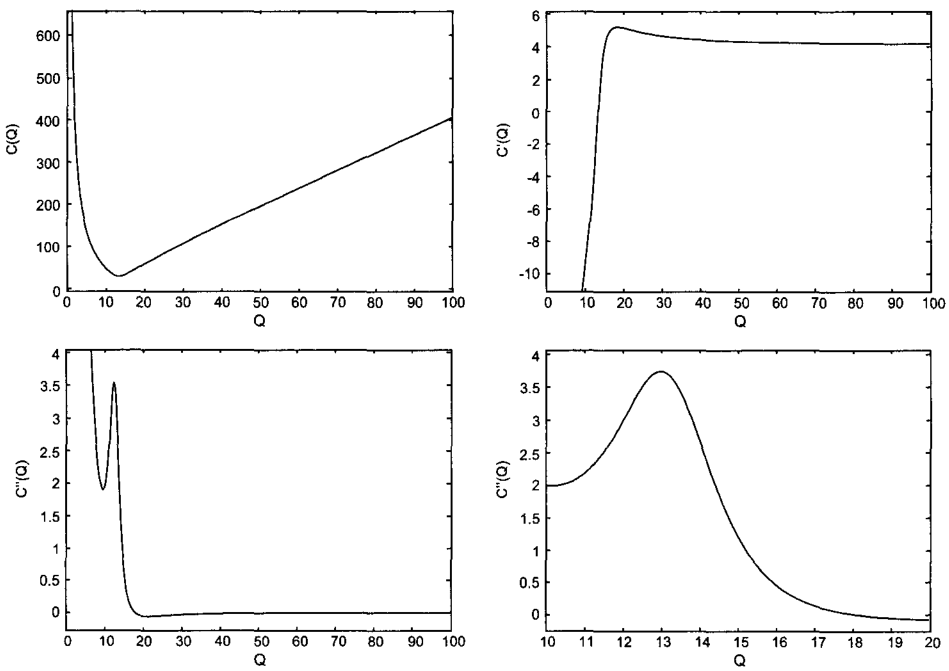

Figure 1. Plots of $C(Q)$, $C'(Q)$ and $C''(Q)$ for the example case

Nonconvex nonlinear optimization problems do not lend themselves to efficient algorithms, and meta-heuristic methods including genetic algorithms have been generally applied to such problems to obtain globally optimal solutions [5, 7]. Therefore, we find it an appropriate methodology to utilize genetic algorithms for solving problems like (8). The problem involves only one variable Q and the fitness for a given chromosome is obviously determined by evaluating $C(Q)$. The detailed design of our genetic algorithm is described in section 4.

3. Description of Uncertainty Set

In this section, we propose a way of representing the uncertainty set F by an ellipsoid of possible values for the parameter vector (S, h) assuming that the uncertain parameter vector follows a known probabilistic distribution. First, we introduce the notion of the *certainty level* of an ellipsoidal representation of an uncertainty set and *p-certainty ellipsoidal representations.*

Definition 1 *Given an ellipsoidal representation F of an uncertain parameter vector v, its* ***certainty level*** *p is*

$$p = \Pr(v \in F).$$

Definition 2 *An ellipsoidal representation F of a given uncertain parameter vector is referred to as a* ***p-certainty ellipsoidal representation*** *of the parameter vector if both the condition (a) and (b) are satisfied.*

(a) The certainty level of F is greater than or equal to p.

(b) The volume of F is minimal among ellipsoidal representations satisfying the condition (a).

These definitions express quite a natural and intuitive idea by which one may build a probabilistically meaningful ellipsoidal representation of the uncertainty set of the parameter vector when a probabilistic distribution of the parameter vector is given.

Now, we address a way to find a ***p*-certainty ellipsoidal representation** of an uncertain parameter vector. In many cases, it is analytically intractable to find the exact representation, so we develop a computational method utilizing a combination of Monte Carlo simulation and genetic algorithm. Monte Carlo simulation is used for determining whether a given ellipsoidal representation satisfies the condition (a) or not. A genetic algorithm is designed to find an ellipsoid of which volume is minimal among ellipsoids satisfying the condition (a). Details of the method will be covered in the next subsections, and hereafter we assume the uncertain parameter vector (S, h) follows jointly normal distribution.

3.1 Calculation of certainty level using Monte Carlo simulation

Given an ellipsoid F, it is needed in the proposed method to calculate the certainty level p_F of F,

$$p_F = \int_{\xi \in F} f(\xi) dS\, dh \tag{13}$$

where, $f(\xi)$ is the probability density function of jointly normal distribution of the parameter vector $\xi = (S, h)^T$. It is not easy to calculate (13) exactly so that Monte Carlo simulation technique is utilized for approximation, instead.

In the proposed method, a number of pairs of two random values which are jointly normal distributed and represent S and h, are generated, and tested if each pair is contained in the given ellipsoid F. Then we approximate p_F to be the proportion of the pairs which are contained in F.

If we set the number of pairs randomly generated to n and let the proportion of the pairs which are contained in F be $\hat{p}$, then from the theory of interval estimation, we can conclude that we have $(1-\alpha)\%$ confidence that the interval,

$$\hat{p} - z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}} < p_F < \hat{p} + z_{\alpha/2}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}},$$

contains the true p_F, where $z_{\alpha/2}$ is the $(1-\alpha/2)$ percentile of the standard normal distribution. For example, if we obtain $\hat{p} = 0.9$ from 3,000 randomly generated pairs, then we have 99% confidence that the interval from 0.886 to 0.914 contains the true p_F.

3.2 A genetic algorithm for finding *p*-certainty ellipsoidal representation

Given the procedure for checking whether a given ellipsoid satisfies (approximately) the condition (a), it is needed to find as small an ellipsoid as possible in order to come up with an approximate *p*-certainty ellipsoidal representation. In other words, we should find a minimum-volume ellipsoid which satisfies the condition (a), where the certainty level of an ellipsoid is calculated in the way described in section 3.1. Since the problem has extremely hard combinatorial nature and a way of analytical analysis is not obvious, we find it is appropriate to utilize the framework of genetic algorithms. We refer the readers to [6] for general theory of genetic algorithms.

In the proposed genetic algorithm, the defining matrix P in the p-certainty ellipsoidal representation is to be approximated. At each generation, a new set of approximations P's is created by applying genetic operations to the old set. Individuals, or current approximations, are encoded as binary strings, called chromosomes.

Without loss of generality we can assume P is symmetric and positive definite. For a symmetric positive definite matrix $P \in \Re^{2\times 2}$,

$$P = \begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix}, \quad p_1 > 0, \quad p_3 > 0, \quad p_1 p_3 - p_2^2 > 0, \tag{14}$$

the corresponding chromosome C is represented as

$$C = [\ c_1 \quad c_2 \quad c_3\ ]$$

where c_1, c_2 and c_3 are binary representations for p_1, p_2 and p_3, respectively. As indicated in (14), the positive definiteness of P requires that $p_1 > 0$, $p_3 > 0$ and $p_1 p_3 - p_2^2 > 0$. The first two of these conditions are incorporated in the chromosome representation, but the third condition is treated as a feasibility criterion and takes part in evaluating fitness values.

In genetic algorithms, it is needed to assess the performance, or fitness, of individual members of a population, and this is done through an objective function that characterizes an individual's performance in the problem domain. Thus, the objective function establishes the basis for selection of pairs of individuals that will be mated together during the reproduction [4]. In our genetic algorithm, the objective function is designed to incorporate both the infeasibility penalty of individuals and the volumes of ellipsoids represented by individuals. The volume of an ellipsoid is proportional to the determinant of the defining matrix. So, the objective function of the algorithm in the sense of minimization is

$$\begin{aligned} f(x) &= M, && \text{if } p_1 p_3 - p_2^2 \le 0 \text{ or } p_x < p \\ &= p_1 p_3 - p_2^2, && \text{otherwise,} \end{aligned} \tag{15}$$

where x is a given individual, M is a big positive number, and p_x is the certainty level of the ellipsoid represented by x. Note that the certainty level p_x is calculated approximately by Monte Carlo simulation as described in the previous subsection.

It is usual, for a performance reason, to transform the objective function values into a measure of relative fitness,

$$F(x) = g(f(x)), \tag{16}$$

where g transforms the value of the objective function f to a non-negative number and F is the resulting relative fitness. There are many ways of constructing F, and our choice is described in the next section.

4. Implementation with MATLAB

In the implementation of our method, we utilize the general purpose genetic algorithm MATLAB toolbox [4]. The toolbox provides several algorithmic options and allows users to tune parameters to find the best setting. We use genetic algorithms for both optimizing the problem (8) and finding an approximate p-certainty ellipsoidal representation, and adopt the same strategies for genetic operations for both the problems, except the maximum number of generations.

For population representation, we use Gray coding to overcome the hidden representational bias in conventional binary representation as the Hamming distance between adjacent values is constant.

In selection process, rank-based fitness assignment method is adopted to avoid the situation where highly fit individuals in early generations can dominate the reproduction causing rapid convergence to possibly sub-optimal solutions. Also, we use the stochastic universal sampling method provided by the toolbox.

For the recombination process, single-point crossover method with crossover rate 0.7 is used. We use an elitist strategy where one or more of the fittest individuals are deterministically allowed to propagate through successive generations. The value of generation gap, which is the fractional difference between the new and old population sizes, is set to 0.9.

The length of binary string for representing one variable is set to 20, and the number of individuals per population is set to 200. For the case of optimizing the problem (8) we set the maximum number of generations equal to 30, and for the case of finding an approximate *p*-certainty ellipsoidal representation we set the number to 200.

The Monte Carlo simulation procedure is implemented also with the MATLAB. The MATLAB function *mvnrnd* is used for generating jointly normal random values of (S, h). 3,000 pairs of values are generated, and the number of pairs which are contained in a given ellipsoid is counted to produce the approximate certainty level of the given ellipsoid. The confidence interval of the true certainty level and its confidence level are already calculated at the end of subsection 3.1 as an example.

5. Numerical Experiment

We tested our method with various statistical parameter settings. Parameters we change in the experiment are standard deviations of S and h and correlation coefficient between them, as well as certainty level p. The other parameters are fixed such that the average demand rate is fixed to 10,000, and means of S and h are fixed to 1,000 and 10, respectively, because these parameters take no effect on the performance of the method.

We tested with standard deviations being 10%, 20% and 30%, and correlation coefficient being -0.8, -0.5, -0.3, 0, 0.3, 0.5, and 0.8. Along with these settings, certainty level p is set to 0.9 or 0.95. The experimental results are summarized in Table 1. Among the symbols at the top of the table, ρ means the correlation coefficient which is calculated as $\sigma_{S,h}/(\sigma_S\sigma_h)$, and *gain* and *loss* are calculated as

$$gain = \frac{C(Q_d^*)-C(Q^*)}{C(Q^*)}\times 100, \qquad loss = \frac{C_d(Q^*)-C_d(Q_d^*)}{C_d(Q_d^*)}\times 100.$$

Note also that Q_d^* is the optimal order quantity in the deterministic model with order setup cost and inventory carrying cost being fixed to 1,000 and 10, respectively. The amount *gain* means how much more Q^* gets over Q_d^* in terms of the worst-case robust optimization version objective function C. Similarly, the amount *loss* means how much more Q_d^* gets over Q^* in terms of the deterministic objective function C_d.

Table 1. Test results on various statistical parameter settings

No.	σ_S	σ_h	ρ	p	Q^*	$C(Q^*)$	$C(Q_d^*)$	*gain*(%)	$C_d(Q^*)$	*loss*(%)
1	100	1	0.8	0.90	1407.623	16818.522	16818.709	0.001	14142.290	0.001
2	100	1	0.5	0.90	1395.407	16617.116	16618.694	0.010	14143.403	0.009
3	100	1	0.3	0.90	1397.610	16656.540	16657.780	0.007	14143.122	0.007
4	100	1	0.0	0.90	1384.706	16226.517	16230.609	0.025	14145.279	0.022
5	200	2	0.8	0.90	1397.610	19689.587	19691.004	0.007	14143.122	0.007
6	200	2	0.5	0.90	1386.881	19454.702	19458.698	0.021	14144.829	0.019
7	200	2	0.3	0.90	1386.080	19134.630	19138.947	0.023	14144.991	0.020
8	200	2	0.0	0.90	1400.871	18341.692	18342.699	0.005	14142.771	0.004
9	300	3	0.8	0.90	1397.839	22331.854	22333.439	0.007	14143.095	0.007
10	300	3	0.5	0.90	1393.862	21761.341	21763.900	0.012	14143.621	0.011
11	300	3	0.3	0.90	1429.853	21065.354	21066.879	0.007	14142.991	0.006
12	300	3	0.0	0.90	1390.085	20358.060	20362.001	0.019	14144.230	0.015
13	100	1	-0.8	0.90	1406.507	15080.185	15080.534	0.002	14142.347	0.001
14	100	1	-0.5	0.90	1413.431	15572.669	15572.672	0.000	14142.138	0.000
15	100	1	-0.3	0.90	1408.882	15918.751	15918.885	0.001	14142.236	0.001
16	200	2	-0.8	0.90	1389.027	16101.140	16106.089	0.031	14144.419	0.016
17	200	2	-0.5	0.90	1358.328	17141.509	17162.571	0.123	14153.632	0.081
18	200	2	-0.3	0.90	1379.185	17765.719	17773.194	0.042	14146.584	0.031
19	300	3	-0.8	0.90	1390.371	16871.523	16877.874	0.038	14144.180	0.014
20	300	3	-0.5	0.90	1429.510	18379.480	18381.409	0.010	14142.954	0.006
21	300	3	-0.3	0.90	1398.096	19561.881	19563.768	0.010	14143.065	0.007
22	100	1	0.8	0.95	1418.095	17273.389	17273.455	0.000	14142.189	0.000
23	100	1	0.5	0.95	1421.957	17120.413	17120.681	0.002	14142.346	0.001
24	100	1	0.3	0.95	1411.142	16848.725	16848.769	0.000	14142.169	0.000
25	100	1	0.0	0.95	1391.487	16652.067	16654.523	0.015	14143.992	0.013
26	200	2	0.8	0.95	1415.520	20354.238	20354.248	0.000	14142.142	0.000
27	200	2	0.5	0.95	1416.664	20225.441	20225.474	0.000	14142.157	0.000
28	200	2	0.3	0.95	1414.175	19432.622	19432.622	0.000	14142.136	0.000
29	200	2	0.0	0.95	1381.044	18932.527	18939.188	0.035	14146.119	0.028
30	300	3	0.8	0.95	1414.289	23559.785	23559.785	0.000	14142.136	0.000
31	300	3	0.5	0.95	1417.494	22628.787	22628.857	0.000	14142.174	0.000
32	300	3	0.3	0.95	1398.869	22271.342	22272.935	0.007	14142.977	0.006
33	300	3	0.0	0.95	1370.974	21448.607	21462.150	0.063	14148.954	0.048
34	100	1	-0.8	0.95	1407.423	15249.945	15250.222	0.002	14142.299	0.001
35	100	1	-0.5	0.95	1423.960	15821.779	15822.271	0.003	14142.469	0.002
36	100	1	-0.3	0.95	1402.359	16195.422	16196.113	0.004	14142.637	0.004
37	200	2	-0.8	0.95	1381.903	16446.531	16455.100	0.052	14145.913	0.027
38	200	2	-0.5	0.95	1409.254	17390.687	17390.863	0.001	14142.223	0.001
39	200	2	-0.3	0.95	1406.937	18038.542	18038.885	0.002	14142.324	0.001
40	300	3	-0.8	0.95	1391.573	17353.583	17359.469	0.034	14143.977	0.013
41	300	3	-0.5	0.95	1418.896	18972.505	18972.701	0.001	14142.213	0.001
42	300	3	-0.3	0.95	1401.958	19987.779	19988.974	0.006	14142.671	0.004

As can be seen in the results, the amounts *gain* and *loss* are very small. Thus, we can conclude that C as well as C_d is insensitive to the change of order quantity around the optimal points. On the other hand, we observe the fact that *gain* is significantly larger than *loss*, which implies the better trade-off performance of Q^* over Q_d^* in the sense of worst-case optimization.

Next, we plot the progress of the two genetic algorithms, the objective function in (8), and the approximate p-certainty ellipsoidal representation for each problem instance presented in Table 1. The plots for several selected representative instances are presented in from Figure 2 to Figure 6 in APPENDIX (Five selected instances in Table 1 are indicated with grey color). In each figure, the progress plot of the genetic algorithm finding an approximate p-certainty ellipsoidal representation is on the top left, and the progress plot of the genetic algorithm solving the problem (8) is on the top right. Also, the approximate p-certainty ellipsoidal representation found by the genetic algorithm is drawn, superimposed with 3,000 randomly generated pairs of (S, h), on the bottom left, and on the bottom right is the objective function $C(Q)$.

From the plots, we observe that the genetic algorithm for solving the problem (8) converges after about 5 generations in most cases, and that the final solutions are nearly optimal, which can be confirmed by checking the plots of the objective functions. We also see that the genetic algorithm for finding an approximate p-certainty ellipsoidal representation takes on average between 100 and 200 generations to converge, and that the approximate p-certainty ellipsoidal representations are properly shaped reflecting well the certainty level and the corresponding statistical properties of (S, h).

Regarding computational times, it takes on average one minute to solve one problem instance. The method is implemented with the MATLAB version 6.5 [9], and we run the implemented program on a laptop computer with 1.73Ghz Intel Pentium Mobile processor and 512M main memory.

6. Concluding Remark

We have shown how a worst-case robust EOQ model can be solved by building up a solution method based on genetic algorithms combined with Monte Carlo simulation.

We couldn't obtain a simple analytic solution to the model as in the deterministic case, because the objective function of the model is not always convex. However, we developed an efficient computational method for the model, which can be easily utilized in the field.

While we implemented the method assuming the uncertain parameter vector is modeled as a random variable vector which follows jointly normal distribution, any other probability distribution for the uncertain parameter vector can be assumed with slight modification to the implementation.

In this research we consider only the case that order setup costs and inventory carrying costs have uncertainty in their values. As a further research issue, the situation, where all of parameters including average demand rate have uncertainty in their values, is worth considering. Because nonlinearity is introduced in that case, more complication is involved than in the case considered in this paper.

References

[1] Ben-Tal, A. and A. Nemirovski, "Robust Solutions of Uncertain Linear Programs," *Oper. Res. Lett.* 25 (1999), 1-13.

[2] Ben-Tal, A. and A. Nemirovski, "Robust Convex Optimization," *Mathematics of Operations Research* 23, 4 (1998), 769-805.

[3] Boyd, S. and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[4] The Department of Automatic Control and Systems Engineering, The University of Sheffield, Genetic Algorithm Toolbox for Use with MATLAB ver 1.2 User's Guide, 1994.

[5] Gallagher, K. and M. Sambridge, "Genetic Algorithms: A Powerful Tool for Large-scale Nonlinear Optimization Problems," *Computers and Geosciences* 20 (1994), 1229-1236.

[6] Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley Publishing Company, 1989.

[7] Keane, A. J., "Genetic Algorithm Optimization of Multi-peak Problems: Studies in Convergence and Robustness," *Artificial Intelligence in Engineering* 9, 2 (1995),

75-83.

[8] Lowe, T. J. and L. B. Schwartz, "Parameter Estimation for the EOQ Lot-Size Model: Minimax and Expected Value Choices," *Naval Research Logistics Quaterly* 30 (1983), 367-376.

[9] The MathWorks, Inc., MATLAB (software package), http://www.mathworks.com, 2004.

APPENDIX

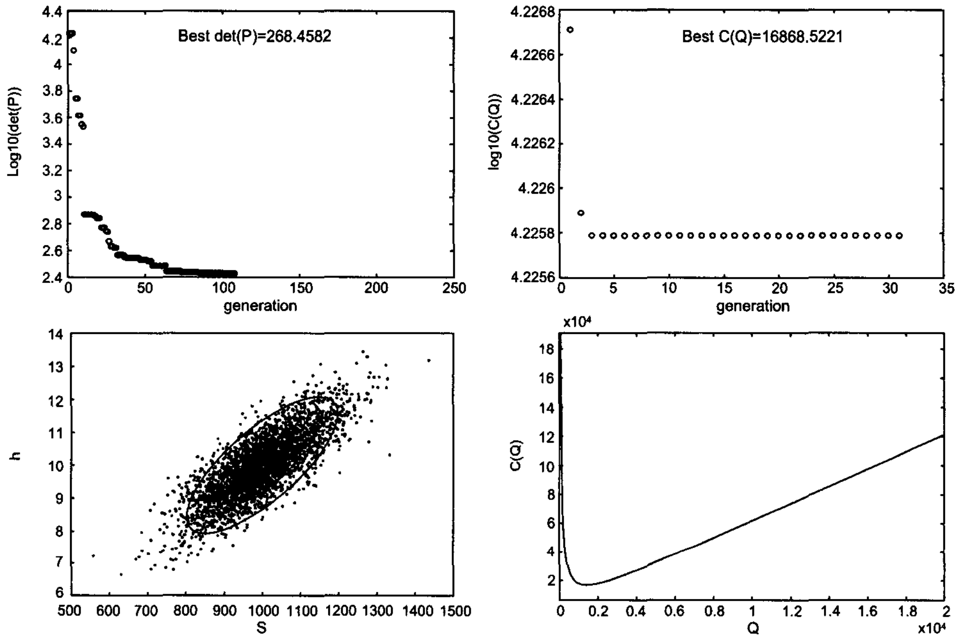

Figure 2. Plots for the No. 1 case

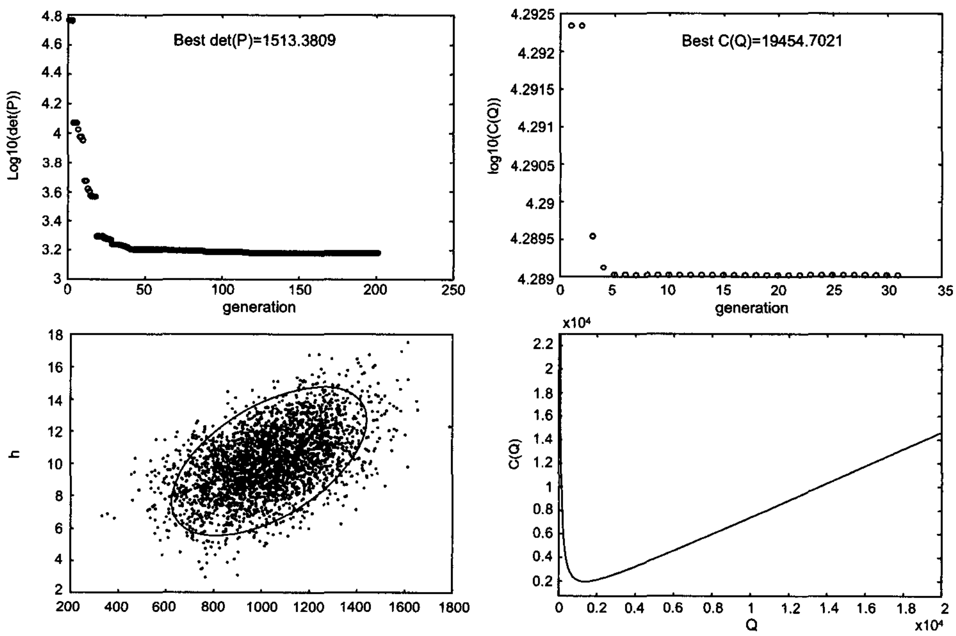

Figure 3. Plots for the No. 6 case

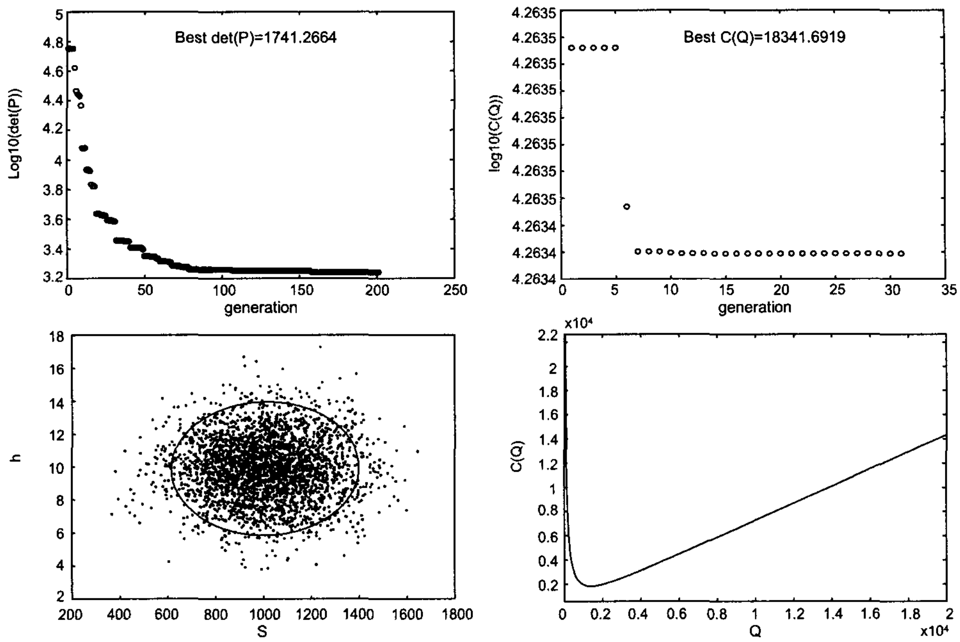

Figure 4. Plots for the No. 8 case

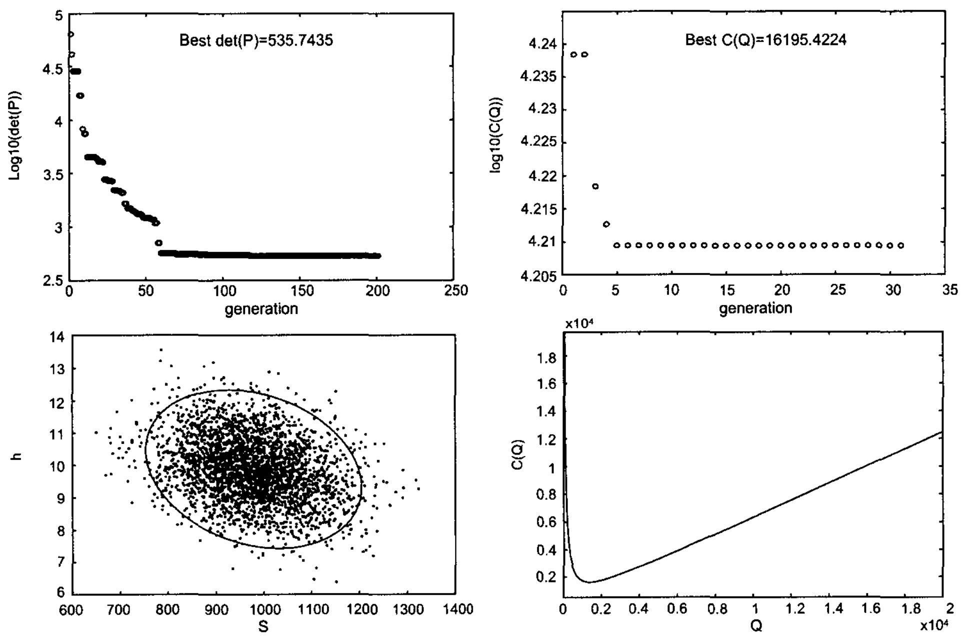

Figure 5. Plots for the No. 36 case

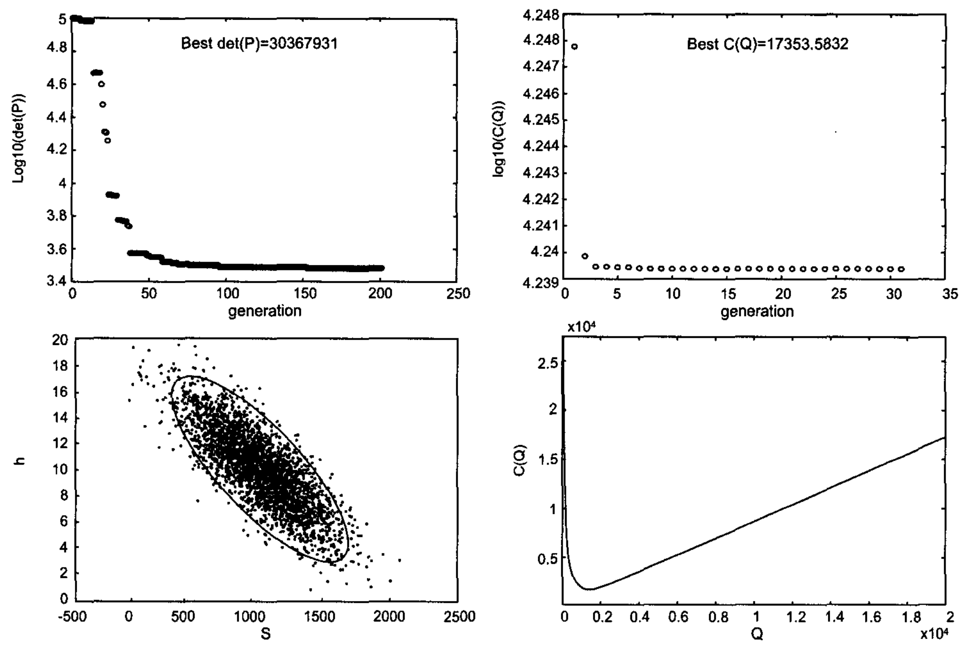

Figure 6. Plots for the No. 40 case