1. Introduction

In this project, we formulate a Stochastic optimization framework to manage the risk from input uncertainty in inventory management.

In the classical inventory management, the stochastic programming is used to find an optimal policy that optimizes the risk-neutral expected cost over a certain planning horizon [3], or the risk from the randomness of the demand [7]. However, to the best of our knowledge, we have not found any study on the risk from the input uncertainty of the parameters in the demand distribution. Therefore, in this project, we develop a Stochastic optimization framework to control the risk from the input uncertainty in inventory management, and provide some numerical studies based on these framework.

We use a coherent Conditional Value at Risk (CVaR) [1] to measure the risk. Due to the difficulties of evaluate an analytical form of CVaR, we formulate a sample average approximation (SAA) for it, and transform the stochastic program into a linear program, which can be solved numerically. We develop the framework under a finite horizon periodic review inventory model without fixed order cost. The empirical study uses a state-dependent Markovian demand process. However, our framework can also be generalized to other inventory models.

Our report is organized as follows. In the next section, we will provide the theoretical formulation of the Stochastic program framework. In section 3, we transform the Stochastic program into a linear program using SAA. In section 4, we introduce the state-dependent Markovian demand process that will be used in the numerical study. And we provide the design of experiment and the result of the numerical study in section 5. We conclude the report and provide some discussions in section 6. We also list the notations in Appendix A and provide the matlab code for the empirical study in Appendix B for the readers’ convenience.

Acknowledgment

We are thankful to Prof. John Mitchell, Prof. Wei Xie, and Prof. Jennifer Ryan for their invaluable discussion and help during this project.

2. Stochastic programming formulation

In this section, we formulate the Stochastic programming for our inventory problem. We first introduce the Stochastic programming formulation for the inventory model, then add the risk measure structure on the
parameter $θ$ and formulate the Stochastic programming problem with input uncertainty.

The inventory model for cost function

We first consider a single item inventory problem with periodic review order-up-to policy. In each period $t$, the order-up-to level is denoted by $S_t$. Then at the beginning of each period, an order of amount $S_t - x_t$ is placed before the demand $D_t$ of the period happens, we assume no lead time or fixed order cost for the orders. The unsatisfied demand are back-ordered and will be satisfied in the following periods. The number of period we look forward is $T$.

Now let $x_t$ denote the on-hand inventory at the beginning of period $t$. With $x_t > 0$ it means that we have left-over inventories from the last period, while $x_t < 0$ indicates that we have backorder. If $g_t(x_t, y, D_t)$ denotes the cost incurred during period $t$, $c$ denotes the purchase cost per unit of the item, $h^+$ denotes holding cost per unit of item per period and $h^-$ denotes the back-order cost per unit of item per period, we then have for given parameter $θ$:

$$g_t(x_t, S_t, D_t; θ) = c(S_t - x_t) + h^+ \max\{S_t - D_t, 0\} + h^- \max\{-(S_t - D_t), 0\}. \tag{1}$$

Notice that $D = (D_1, D_2, \ldots, D_t)$ is a vector of random variable, if $\mathbf{d} = (d_1, d_2, \ldots, d_T)$ is a realization of $D$, the first stage problem we have is to find the inventory levels at the beginning of each period that optimize the total cost over the planning horizon.

The total cost over the planning horizon is

$$Q(x, S, \mathbf{d}; θ) = \sum_{t=1}^{T} r^{t-1}(c(S_t - x_t) + h^+ \max\{S_t - d_t, 0\} + h^- \max\{-(S_t - d_t), 0\}) =$$

$$-cx_1 + \sum_{t=1}^{T} r^{t-1}(cS_t - (h^- + rc)(S_t - d_t) + (h^+ + h^-) \max\{S_t - d_t, 0\}) + r^T c(S_T - d_T), \tag{2}$$

where $r$ is the discount factor from time value of money. Here if $S_t - (S_{t-1} - d_{t-2}) < 0$, we are making a 'purchase' with negative amount. So we assume that if our on-hand inventory exceeds the order-up-to level, we can return the excess inventory to the supplier at the same unit price $c$. And making a purchase with a negative amount is equivalent to returning a positive amount. Such assumption is commonly used in supply chain literatures [2].

So optimizing the total cost, we get the Stochastic programming as follows [6]:

$$\min_{x,S} \quad Q(x, S, \mathbf{d}; θ) = \sum_{t=1}^{T} r^{t-1} g_t(x_t, S_t, d_t)$$

$$= -cx_1 + \sum_{t=1}^{T} r^{t-1}(cS_t - (h^- + rc)(S_t - d_t) + (h^+ + h^-) \max\{S_t - d_t, 0\}) + r^T c(S_T - d_T) \quad \text{(SP1)}$$

$$s.t. \quad x_{t+1} = S_t - d_t, \text{ for } t = 1, 2, \ldots, T - 1$$

Notice that the optimal value $Q^*(S(\mathbf{d}; θ), \mathbf{d}; θ)$ of (SP1) is the minimum cost and it is a function of the order-up-to level $S$ and the random demand variable $D$. So in order to get the order-up-to level $S(\mathbf{d}; θ)$, we need to solve the following Stochastic optimization problem.

$$\min_{S_1, S_2(\cdot), S_3(\cdot), \ldots, S_T(\cdot)} \quad \mathbb{E}[Q^*(S(\mathbf{d}; θ), \mathbf{d}; θ)|θ], \quad \text{(SP2)}$$
Where the expectation in (SP2) is taking over the demand vector $D$. Assuming an buyback contract with the buyback price the same as the unit purchase price, it has been shown that if the demand follows an AR(1) process with $D_t = \beta_0 + \beta_1 D_{t-1} + \epsilon_t$, (SP1) - (SP2) has an closed form optimal solution for $S_t = S_t(d_{t-1})$ for $t=2,3,\ldots,T$. 

The inventory model with risk measure

As we mentioned previously, the demand process we consider is specified by parameters $\theta$. Denote the historic demand data before time $t$ by $d_{t-1}$. The unknown input parameters are estimated from $d_{t-1}$ at time $t$. The estimation error if quantified by the posterior distribution $b(\theta) = p(\theta|d_{t-1})$. With $\Theta \sim b(\theta)$, we have $\Theta$ being a random vector characterizing our belief over the true input parameter $\theta$.

To control the risk from the uncertainties in the parameters, we want to optimize a coherence risk measure of the expected cost. In particular, we want to optimize the Conditional Value-at-Risk (CVaR). For a random cost $C(\Theta)$, the $\alpha-CVaR$ is the expected cost in the worst $1-\alpha$ cases. In particular, the $\alpha-CVaR$ can mathematically be written as follows [1]:

$$CVaR_{\alpha}(C(\Theta)) = \frac{1}{1-\alpha} \int_{\alpha}^{1} \inf\{t \in \mathbb{R}| \mathbb{P}(C(\Theta) \leq t) \geq \beta\} d\beta.$$ (3)

Now the expected cost $\mathbb{E}[Q^*(S(d; \theta), d; \theta)|\theta]$ in (SP2) is in fact depends on the parameter $\theta$. So enforcing the CVaR risk measure which characterizes the impact of input uncertainty, the Stochastic program (SP2) becomes

$$\min_{S_1, S_2, \ldots, S_T} CVaR_{\alpha}(\mathbb{E}[Q^*(S(d; \theta), d; \theta)|\theta]),$$ (SP3)

where the risk measure $CVaR_{\alpha}$ is taken over the parameter $\theta$.

Notice that, for given $\theta$, the optimal policy for the risk-neutral expected cost depends on the particular distribution of the demand process and the previous demand realization. For example, when the demand process is state-dependent Markovian, the policy that optimizes the risk neutral expected cost is an order-up-to policy with order-up-to level $S_t$, which depends on the state of period $t$ [4]. When considering the input uncertainty for the parameters $\theta$, the state $I_t$ depends on the parameter $\theta$. Therefore, considering the input uncertainty of the parameters, the optimal policy for (SP3) should be a function of the parameter $\theta$ and the previous demand realization $d_{t-1}$. So we denote the inventory policy by $S_t = S_t(d_{t-1}; \theta)$ for each period $t$.

The next question is, how to turn this two-stage stochastic programming (SP1) - (SP3) into a 'workable' sample average approximation (SAA) formulation.

3. SAA for the Stochastic programming with iid demand

In this section, we transform the Stochastic program (SP1) - (SP3) into an SAA.

For the inner-loop problem (SP1), the optimal solution is $x_1 = S_1$ and the optimal value is

$$Q^*(S(d; \theta), d; \theta) = \sum_{t=1}^{T} r^{t-1}(cS_t(d_{t-1}; \theta) - (h^- + rc)(S_t(d_{t-1}; \theta) - d_t)$$

$$+ (h^+ + h^-) \max\{S_t(d_{t-1}; \theta) - d_t, 0\}) + r^T c(S_T(d_{T-1}; \theta) - d_T).$$ (4)
Then for each parameter sample $\theta$ approximation (SAA) \cite{5}. In our case, we first sample $K$ and all parameters:

$$
\begin{align*}
    d_t &= d_i(\alpha, j, E) = 1 \ E \ S_t(d_{t-1}; \theta) - (h^- + r c)(S_t(d_{t-1}; \theta) - d_t) + (h^+ + h^-)u_t \\
    &+ r^T c(S_T(d_{T-1}; \theta) - d_T). \quad (5)
\end{align*}
$$

In order to solve the outer-loop problem (SP3), we use a popular Monte Carlo methods - sample average approximation (SAA) \cite{5}. In our case, we first sample $K$ parameters $\theta_1, \theta_2, \ldots, \theta_K$ from the posterior $b(\theta)$. Then for each parameter sample $\theta_i$, where $i = 1, 2, \ldots, K$, we sample $n_i$ different demand scenarios $d^{(i,j)} = (d^{(i,j)}_1, d^{(i,j)}_2, \ldots, d^{(i,j)}_T)$, for $j = 1, 2, \ldots, n_i$.

Now we can approximate the expectation $E[Q(S, D)|\theta]$ using the sample mean as follows. Notice that now $S_t^{(i,j)} = S_t(d^{(i,j)}_{t-1}; \theta_i)$ is a function of the parameter $\theta_i$ and the historic demand data $d^{(i,j)}_{t-1}$.

$$
\begin{align*}
    \hat{E}[Q^*(S(d; \theta), d; \theta)|\theta_i] &= \frac{1}{n_i} \sum_{j=1}^{n_i} Q(S^{(i,j)}, d^{(i,j)}, u^{(i,j)}) \\
    &= \frac{1}{n_i} \sum_{j=1}^{n_i} \left( \sum_{t=1}^{T} r^{t-1}(c S^{(i,j)}_t - (h^- + r c)(S^{(i,j)}_t - d^{(i,j)}_t) + (h^+ + h^-)u^{(i,j)}_t) + r^T c(S^{(i,j)}_T - d^{(i,j)}_T) \right). \quad (6)
\end{align*}
$$

Here $u^{(i,j)}_t \geq 0$ are the auxiliary variables corresponding to $d^{(i,j)}$, which satisfies $S^{(i,j)}_t(\cdot) - d^{(i,j)}_t \leq u^{(i,j)}_t$ for all $j = 1, 2, \ldots, n_i$ and all $t = 1, 2, \ldots, T$.

And the $\text{CVaR}_\alpha$ can be approximated as follows \cite{1}:

$$
\text{CVaR}_\alpha(\hat{E}[Q^*(S(d; \theta), d; \theta)|\theta]) = \min_{\mu} \frac{1}{K} \sum_{i=1}^{K} \max\{\hat{E}[Q^*(S(d; \theta_i), d; \theta_i)|\theta_i] + \mu, 0\} - (1 - \alpha) \mu. \quad (\text{SAA-CVaR})
$$

Plug \cite{6} and \cite{SAA-CVaR} into \cite{SP3}, we get the following SAA formulation of the two-stage Stochastic programming:

$$
\begin{align*}
    \min_{S \geq 0, u \geq 0, \mu \in \mathbb{R}} & \quad z_0 \\
    \text{s.t.} & \quad \frac{1}{K} \sum_{i=1}^{K} \max\{\hat{E}[Q^*(S(d; \theta_i), d; \theta_i)|\theta_i] + \mu, 0\} - (1 - \alpha) \mu \\
    & \quad S^{(i,j)}_t - d^{(i,j)}_t \leq u^{(i,j)}_t, \text{ for all } i, j, t
\end{align*}
$$

Introducing auxiliary decision variables $z_i$ for $i = 0, 1, 2, \ldots, K$, problem \cite{SAA} can be reformulated as the following linear program

$$
\begin{align*}
    \min_{S \geq 0, u \geq 0, \mu \in \mathbb{R}, z} & \quad z_0 \\
    \text{s.t.} & \quad \hat{E}[Q^*(S(d; \theta_i), d; \theta_i)|\theta_i] + \mu \leq z_i, \text{ for all } i \\
    & \quad \frac{1}{K} \sum_{i=1}^{K} z_i - (1 - \alpha) \mu \leq z_0 \\
    & \quad S^{(i,j)}_t - d^{(i,j)}_t \leq u^{(i,j)}_t, \text{ for all } i, j, t \\
    & \quad z_i \geq 0 \text{ for } i = 1, 2, \ldots, K.
\end{align*}
$$
We may notice that both the number of decision variables and the number of constraints are $O(T \sum_{i=1}^{K} n_i)$. So the numerical complexity increases very fast if we want to increase sample size and get a better approximation.

4. State-dependent Markovian demand

The demand process we consider is an state-dependent Markovian demand process. In that case, the demand for each period depends on some underlying common factor, for example, the state of economy.

Let the random variable $I_t$ denote the state of the economy in period $t$, we assume the simplest case where the state space only contains two elements: low and high, and denote the state space by $\Omega = \{l, h\}$. We assume that $I_t$ for $t = 1, 2, \ldots$ is a discrete time Markovian process with transition matrix

$$P = \begin{bmatrix} p_{ll} & p_{lh} \\ p_{hl} & p_{hh} \end{bmatrix}. \quad \text{(P)}$$

Here $p_{ij} = P[I_t = i| I_{t-1} = j]$ is the transition probability that the state in period $t$ is $i$ under the condition that the state in period $t - 1$ is $j$, for $i, j \in \{l, h\}$. Notice that $p_{ll} = 1 - p_{lh}$ and $p_{hh} = 1 - p_{hl}$.

Then the demand distribution depends on the state $I_t$ as follows: if $D^{t_i}_i$ denote the random variable representing the demand of period $t$ at state $I_t$, then the probability density function (when $D^{t_i}_i$ is continuous random variable) or the probability mass function (when $D^{t_i}_i$ is discrete random variable) is denoted by $f_{I_t}(d)$, which depends on the state $I_t$.

Under our assumption, we require that $\mu_h > \mu_l$ where $\mu_h = \mathbb{E}[D^{t_i}_i]$ is the expected demand when the state of economy is high, and $\mu_l = \mathbb{E}[D^{t_i}_i]$ is the expected demand when the state of economy is low. For simplicity, we consider $P(D^{t_i}_i = d_1) = \alpha_1$, $P(D^{t_i}_i = d_2) = 1 - \alpha_1$, $P(D^{t_i}_i = d_3) = \alpha_2$, and $P(D^{t_i}_i = d_4) = 1 - \alpha_2$. Therefore, the input process could be specified by $\theta = (p_{ll}, p_{lh}, \alpha_1, \alpha_2)$. For given $\theta$, the policy that optimizes the risk neutral expected cost is an order-up-to policy with order-up-to level $S^{t_i}_i$, which depends on the state of period $t$ [4].

5. Empirical Study

In this section, we present the empirical study using LP transformation to calculate the $\alpha - C\text{VaR}$ when input uncertainties are taking into account.

For simplicity, we only consider the single item, state-dependent inventory problem. The demand for the only product is driven by an unknown distribution. We know the demand consists of two distinctive states, denoted as state1 and state2, each state further has two possible demands: for state1, $d_1, d_2$ are two possible demands, and for state2, $d_3, d_4$ are the two possible demands. Without loss of generosity, we assume $d_1 \leq d_2 \leq d_3 \leq d_4$. When the state is observed or given, the two possible demands have equal opportunity to happen. At the beginning of a time period, the experimenters first observe the state, then make an ordering decision about how many to order, and only after the decision is made comes the realization of the demand. For example, at time period 1, the experimenter first observes state 1, then she makes an ordering decision, and after which, either d1 or d2 happens with equal chance. And we also assume the possibility of next state depends on and only on the current state. In other words, the demand is a Markovian process, and the possibilities of next state given current one is determined by a time-independent transition matrix $q_\theta$.

The experimenters want to make an optimal decision when considering $T$ time periods, that is to say, the experimenters want to optimize the decisions to minimize the expected cumulative cost occurred in the next $m$ periods. It is proved in classical literature that in this case, the optimal decision within a time period is an order up to policy only depending on current state $i$. For instance, at $jth$ time period, the experimenters first
observe the state, and they make a decision based on the current state. And we also assume that for the first period, the observers know the state already. Notice that in the case, we have \(2m - 1\) decision variables, and each decision solely depends on the current state.

If true transition matrix of the distribution are known, then we only need to optimize our ordering decision to minimize the total expected cost when considering \(m\) time periods ahead. However, the transition matrix is unknown to us, and we only have a limited number of historical observations, which are from the true matrix. By using Bayesian approach, we can obtain a posterior distribution about the matrix conditional on the historical data. We then draw randomly \(K\) different transition matrices from the posterior distribution, which quantifies the input uncertainty. Therefore, our goal is to find an optimal set of policies such that the set would minimize the \(\alpha - CVaR\), i.e. the conditional expectation of \(\alpha\) right tail of expected cost among \(K\) different distributions.

Our focus mainly lies in the current period decision. The reason behind this act is clear, we want make justifications when further information comes in.

In the experiment, we fix \(\alpha = 0.9, d_1 = 100, d_2 = 200, d_3 = 300, d_4 = 400\) and the discount factor \(r\) is 0.99, and choose \(T = 5\), i.e. looking five periods ahead. The holding cost \(h^+\) is set as 5, the backorder cost \(h^-\) is set as 10, with the ordering cost at 100 per item. We first choose \(K = 10\), and when we observe first state is state 1, we get optimal policy as \((200, 200, 400, 200, 400, 200, 400, 200, 400, 0, 0)\). The first entry is the optimal policy for period 1, and the second and third are optimal policy for time period 2 given state 1, 2 respectively and so on and so forth. The optimal value for \(CVaR\) is 4836.9. We view the set as the benchmark set, and we want to see the impacts of those parameters.

First, we increase the number of time periods looking ahead, \(T = 10\) while keeping everything else unchanged. The optimal policy then is \((200, 200, 400, 200, 400, 200, 400, 200, 400, 100, 100, 300, 100, 300, 100, 300, 100, 300, 100, 300, 0, 0)\) with optimal value 10077.3. It is obviously that although longer periods may change the optimal policies in later time periods, the main point of interest, the first optimal decision is the same.

Next, we set \(m\) back to 5, and change holding cost \(h^+\) to 15, keeping everything else constant. In this case, we obtain the optimal decision \((100, 100, 300, 100, 300, 100, 300, 0, 0)\) with optimal value 4433.2. The reason is clear, holding is not a good choice now, the decision makers rather prefer backorder.

We also reduce the back order cost \(h^-\) to 1, and we expect the same optimal decision as the last one, and so is it. The reason is quite similar, backorder becomes more profitable.

When changing the ordering cost \(C\) from 50 to 10, and we get \((200, 400, 200, 400, 200, 400, 1, 2)\). It is easy to see decision makers tend to satisfy the needs as the ordering cost is reduced.

Lastly, we now have \(K = 30\) instead of 10, however the optimal policy is consistent with the benchmark one. Which suggests in the situation when the number of possible demand is very limited, the optimal policy is quite stable.

We want to address the computational capacity issue before ending this section. Due the formation of LPP-SAA problem, a great number of ancillary variables are added, resulting in a great number of new constraints and a sparse coefficient matrix. Combined with limited number of variables and constraints Matlab can deal with, this issue prevents us from research questions on more distribution choices, longer looking ahead time periods, and larger variations of demand choices. One of the potential research direction is to resolve this issue either by finding a new method, or using a more powerful commercial software.

6. Conclusion and discussion

In this project, we develop a Stochastic optimization framework using CVaR to manage the risk in inventory management which comes from the input uncertainty of the parameters in demand process. We overcome the difficulties in the analytic evaluation of CVaR using the sample average approximation, and reduce the
stochastic program to a linear program. We then implement the framework using a state-dependent Markovian demand process, and perform a numerical study to find a numerical approximation of the optimal policy.

The Stochastic optimization framework can be generalized to other demand processes or other inventory models, which is our ongoing research topic.

Appendix A: List of Notations

- $T$: number of periods to look forward.
- $t$: index of the periods. $t = 1, 2, \ldots, T$.
- $g_t(\cdot)$: cost incurred for period $t$.
- $x_t$: on-hand inventory at the beginning of period $t$. $x_t > 0$ means that we have left-over inventories from the last period, while $x_t < 0$ indicates that we have backorder.
- $D_t$: random variable representing the demand in period $t$.
- $D = (D_1, D_2, \ldots, D_T)$: the random vector of demands.
- $d_t$: a realization of demand random variable $D_t$.
- $d = (d_1, d_2, \ldots, d_T)$: a vector of demand realization.
- $\theta$: parameters for demand process.
- $b(\theta)$: the posterior distribution for parameter $\theta$.
- $S_t = S_t(d_t)$: order-up-to level for period $t$.
- $S = (S_1, S_2, \ldots, S_T)$: the vector of the order-up-to levels.
- $Q(S, d)$: the minimum total cost if the demand realization is $d$.
- $c$: purchase cost per unit of the item.
- $h^+$: holding cost per unit of inventory per period
- $h^-$: back-order cost per unit of demand per period
- $r$: discount factor by taking into consideration the time value of money.
- $CVaR$: Conditional Value-at-Risk.
- $u, z$: auxiliary variables.
- $N(\mu, \sigma^2)$: normal distribution with mean $\mu$ and variance $\sigma^2$. 
Appendix B: Matlab codes for empirical study

code 1

function [A,B]=OptCons(alpha,K,m,hplus,hminus,c,r,state0,d1,d2,d3,d4,nTol,Demand,State)
    % this function is to form the linear program constraint
    % record sample paths previous states nt2 is the state of demand in period
    % nt first column represents sample path with S(t) given d1 and so on and so forth.
    A0=zeros(K,K+2+2*m-1+K*nTol*m);
    B0=zeros(K,1);
    % A0 stores the coefficients for first K constraint (Avg cost function
    % constraint for each distribution, B0 is the corresponding scalar vector.
    for i=1:K
        % compute the avg cost function constraint
        nt1=0;
        if (m¿=2)
            nt=zeros(m-1,2);
        % nt stores coeffients for optimal policy, for period 2 onwards,
        % each period can two potential pocilies depending on the
        % observations of the current state
        end
        l=(i-1)*nTol+1;
        b=hminus*(sum(Demand(l:l+nTol-1,1)==1)*d1+sum(Demand(l:l+nTol-1,1)==2)*d2+sum(Demand(l:l+nTol-
        1,1)==3)*d3+sum(Demand(l:l+nTol-1,1)==4)*d4);
        for j=2:m
            b=b+power(r,j-1)*(c*(sum(Demand(l:l+nTol-1,j-1)==1)*d1+sum(Demand(l:l+nTol-1,j-1)==2)*d2+sum(Demand(l:l+nTol-
        1,j-1)==3)*d3+sum(Demand(l:l+nTol-1,j-1)==4)*d4)+hminus*(sum(Demand(l:l+nTol-1,j)==1)*d1+sum(Demand(l:l+nTol-
        1,j)==2)*d2+sum(Demand(l:l+nTol-1,j)==3)*d3+sum(Demand(l:l+nTol-1,j)==4)*d4));
            if (j==2)
                nt(1,1)=r*(c-hminus)*sum(State(l:l+nTol-1,2)==1);
                nt(1,2)=r*(c-hminus)*sum(State(l:l+nTol-1,2)==2);
                nt1=nTol*(c-hminus)-r*c*nTol;
            else
                nt(j-1,1)=power(r,j-1)*(c-hminus)*sum(State(l:l+nTol-1,j)==1);
                nt(j-1,2)=power(r,j-1)*(c-hminus)*sum(State(l:l+nTol-1,j)==2);
                nt(j-2,1)=nt(j-2,1)-power(r,j-1)*c*sum(State(l:l+nTol-1,j-1)==1);
                nt(j-2,2)=nt(j-2,2)-power(r,j-1)*c*sum(State(l:l+nTol-1,j-1)==2);
            end
        end
        % filling the coefficient matrix A0
        temp=1/nTol;
        A0(i,1)=-1;
        % it is -t compared to equaltion 41)
        A0(i,(i+2))=-1;
        A0(i,(K+3))=temp*nt1;
for j=2:m
l_j=K+3+(j-2)*2+1;
A0(i,l_j:(l_j+1))=temp.*nt(j-1,:);
end
l_v=K+2+2*m-1+(i-1)*nTol*m+1;
for j=1:m
A0(i,(l_v+j-1):m:(l_v+nTol*m-1))=temp*power(r,j-1)*(hplus+hminus);
end
B0(i)=-temp*b;
end
%
compute the z0 constraint
A1=zeros(1,K+2+2*m-1+K*nTol*m);
A1(1)=1-alpha;
A1(2)=-1;
A1(3:K+2)=1/K;
temp=K*nTol*m;
%
compute vjt constraint, which is the maximum of (St-dt,0) for each
%
sample path for each distribution at each time period
A2=zeros(temp,2*m-1);
B2=zeros(temp,1);
for j=1:m
if j==1
A2(1:m:temp,1)=1;
B2(1:m:temp)=(Demand(:,1)==1)*d1+(Demand(:,1)==2)*d2+(Demand(:,1)==3)*d3+(Demand(:,1)==4)*d4;
else
l_j=(j-1)*2;
A2(j:m:temp,l_j)=(State(:,j)==1);
A2(j:m:temp,l_j+1)=(State(:,j)==2);
B2(j:m:temp)=(Demand(:,j)==1)*d1+(Demand(:,j)==2)*d2+(Demand(:,j)==3)*d3+(Demand(:,j)==4)*d4;
end
end
A3=[zeros(temp,K+2),A2,-eye(temp)];
A=[A0;A1;A3];
B=[B0;B1;B2];

% code 2

function [s,Cvar]=OptCvar(alpha,K,m,hplus,hminus,c,r,state0,tol,d1,d2,d3,d4)
% s is the set of optimial decision variables, Cvar is the value of cvar,
% NoDis stores the numbers of distributions which fall into alpha tail.
% alpha detimines the alpha-cvar. K is the number of distributions we wanna
% consider, m is the number of time periods in consideration. hplus is the
% holding cost, hminus is the back order cost, c is the order cost per
% unit.r is the time discount factor. state0 is the initial state we
% observe, 1 denotes low state, 2 denotes we are at high state.
% this function optimizes Cvar value and returns the optimal policy

currently, we assume binary state structure. Each state also has two
possible demand levels. i.e. we have four possible demand levels, d1 d2
d3 d4. tol is the tolerance for s(1), if the change of s(1) smaller than
tol, we stop simulation.

Assume at time 0, there is no inventory on hand
requires m¿=3
s=zeros(1,2*m-1);
s defines the optimal set of decisions. s(1) is the first time period
decision. s(2-3) is the second time period decision, each entry represents
given the realization of the first state, notice since we have known the
first state, the only uncertainty is its demand, so we only have two
possible realizations for the first demand, and for time period
thereafter (t-2)*4 is the optimal decision given previous state is d1, (t-2)*4+1 denotes the optimal decision
given d2

Cvar=0;
n0=30;
n1=20;
n0 is the initial number of simulations, n1 is the number of incremental
simulations
% randomly generate transition matrix for each distribution, this can be
% changed for importing transition matrices.
qT=betarnd(10.5,[1 2]);
pT=0.5*ones(1,2);
[StateT,DemandT]=OptSim(state0,qT,pT,1,1,100);
ll=0;
lh=0;
hh=0;
hl=0;
for i=1:99
if (StateT(i)==1)
if (StateT(i+1)==1)
ll=ll+1;
else lh=lh+1;
end
else
if (StateT(i+1)==1)
hl=hl+1;
else hh=hh+1;
end
end
q01=betarnd(1+ll,1+lh,[K 1]);
q02=betarnd(1+hl,1+hh,[K 1]);
q0=[q01,q02];
% each row represents a distribution first column represents the
% probability from state1 to state1, while the second column is the probability
from state 2 to state 1.
\[ p_0 = 0.5 \times \text{ones}(K, 2); \]
% each row represents a distribution first column represents the
% probability of d1 when state is 1, while the second column is the probability
% of d3 when state is 2.
s1old = 0;
% stores old s1
nSys = ceil(K \times (1 - \alpha));
% number of systems in the tail
% the next function body is to find the optimal solution
[State, Demand] = OptSim(state0, q0, p0, n0, K, m);
% total number of simulations done so far
nTol = n0;
% form constraint
[A, B] = OptCons(alpha, K, m, hplus, hminus, c, r, state0, d1, d2, d3, d4, nTol, Demand, State);
f = zeros(K + 2 + 2*n1 - 1 + K*nTol*m, 1);
lb = zeros(K + 2 + 2*n1 - 1 + K*nTol*m, 1);
f(2) = 1;
[x, Cvar] = linprog(f, A, B, [], [], lb, [], [], 'simplex');
s = x(K + 3:K + 1 + 2*n1);
s1old = s(1);%
while 1
% simulate new sample paths
[NewState, NewDemand] = OptSim(state0, q0, p0, n1, K, m);
% combine sample paths
OldDemand = Demand;
OldState = State;
State = zeros(K \times (n1 + nTol), m);
for i = 1:K
i = (i - 1) * n1 + 1;
lold = (i - 1) * nTol + 1;
lsum = (i - 1) * (n1 + nTol) + 1;
Demand(lsum:lsum + n1 - 1,:) = NewDemand(l:l + n1 - 1,:);
Demand(lsum + n1:lsum + n1 + nTol - 1,:) = OldDemand(lold:lold + nTol - 1,:);
State(lsum + n1:lsum + n1 + nTol - 1,:) = OldState(lold:lold + nTol - 1,:);
end
nTol = nTol + n1;
% solve LP
[A, B] = OptCons(alpha, K, m, hplus, hminus, c, r, state0, d1, d2, d3, d4, nTol, Demand, State);
f = zeros(K + 2 + 2*n1 - 1 + K*nTol*m, 1);
lb = zeros(K + 2 + 2*n1 - 1 + K*nTol*m, 1);
f(2) = 1;
[x, Cvar] = linprog(f, A, B, [], [], lb, [], [], 'simplex');
s = x(K + 3:K + 1 + 2*n1);
if abs(s(1)-s1old) < tol
break;
end
s1old=s(1);
% record the no of simulations performed so far
end
end

code 3

function [s,Cvar]=OptCvar(alpha,K,m,hplus,hminus,c,r,state0,tol,d1,d2,d3,d4)
% s is the set of optimal decision variables, Cvar is the value of cvar,
% NoDis stores the numbers of distributions which fall into alpha tail.
% alpha detimines the alpha-cvar. K is the number of distributions we wanna
% consider, m is the number of time periods in consideration. hplus is the
% holding cost, hminus is the back order cost, c is the order cost per
% unit. r is the time discount factor. state0 is the initial state we
% observe, 1 denotes low state, 2 denotes we are at high state.
% this function optimizes Cvar value and returns the optimal policy
% currently, we assume binary state structure. Each state also has two
% possible demand levels. i.e. we have four possible demand levels, d1 d2
d % d3 d4. tol is the tolarence for s(1), if the change of s(1) smaller than
% the tol, we stop simulation.
% Assume at time 0, there is no inventory on hand
% requires m=3
s=zeros(1,2*m-1);
% s defines the optimal set of decisions. s(1) is the first time period
% decision. s(2-3) is the second time period decision, each entry repesents
% given the realization of the first state, notice since we have known the
% first state, the only uncertainty is its demand, so we only have two
% possible realizations for the first demand, and for time period
% thereafter (t-2)*4 is the optimal decision given pevious state is d1, (t-2)*4+1 denotes the optimal decision given d2
Cvar=0;
n0=30;
n1=20;
% n0 is the initial number of simulations, n1 is the number of incremental
% simulations
% randomly generate transition matrix for each distribution, this can be
% changed for importing transition matrices.
qT=betarnd(10,5,[1 2]);
pT=0.5*ones(1,2);
[StateT,DemandT]=OptSim(state0,qT,pT,1,1,100);
l1=0;
lh=0;
hh=0;
hl=0;
for i=1:99
if (StateT(i)==1)
if (StateT(i+1)==1)
ll=ll+1;
else lh=lh+1;
end
else
if (StateT(i+1)==1)
hl=hl+1;
else hh=hh+1;
end
end
q01=betarnd(1+ll,1+lh,[K 1]);
q02=betarnd(1+hl,1+hh,[K 1]);
q0=[q01,q02];
% each row represents a distribution first column represents the
% probability from state1 to state1, while the second column is the probability
% from state 2 to state 1.
p0=0.5*ones(K,2);
% each row represents a distribution first column represents the
% probability of d1 when state is 1, while the second column is the probability
% of d3 when state is 2.
s1old=0;
% stores old s1
nSys=ceil(K*(1-alpha));
% number of systems in the tail
% the next function body is to find the optimal solution
[State,Demand]=OptSim(state0,q0,p0,n0,K,m);
% total number of simulations done so far
nTol=n0;
% form constraint
[A,B]=OptCons(alpha,K,m,hplus,hminus,c,r,state0,d1,d2,d3,d4,nTol,Demand,State);
f=zeros(K+2+2*m-1+K*nTol*m,1);
lb=zeros(K+2+2*m-1+K*nTol*m,1);
f(2)=1;
[x,Cvar] = linprog(f,A,B,[],lb,[],[],'simplex');
s=x(K+3:K+1+2*m);
s1old=s(1);
while 1
% simulate new sample paths
[NewState,NewDemand]=OptSim(state0,q0,p0,n1,K,m);
% combine sample paths
OldDemand=Demand;
OldState=State;
Demand=zeros(K*(n1+nTol),m);
State=zeros(K*(n1+nTol),m);
for i=1:K
l=(i-1)*n1+1;
lold=(i-1)*nTol+1;
lsum=(i-1)*(n1+nTol)+1;
Demand(lsum:lsum+n1-1,:)=NewDemand(l:l+n1-1,:);
Demand(lsum+n1:lsum+n1+nTol-1,:)=OldDemand(lold:lold+nTol-1,:);
State(lsum:lsum+n1-1,:)=NewState(l:l+n1-1,:);
State(lsum+n1:lsum+n1+nTol-1,:)=OldState(lold:lold+nTol-1,:);
end
nTol=nTol+n1;
% solve LP
[A,B]=OptCons(alpha,K,m,hplus,hminus,c,r,state0,d1,d2,d3,d4,nTol,Demand,State);
f=zeros(K+2+2*m-1+K*nTol*m,1);
lb=zeros(K+2+2*m-1+K*nTol*m,1);
f(2)=1;
[x,Cvar] = linprog(f,A,B,[],[],lb,[],[],'simplex');
s=x(K+3:K+1+2*m);
if abs(s(1)-s1old)<tol
break;
end
s1old=s(1);
% record the no of simulations performed so far
end
end

References