Response Knowledge Learning of Autonomous Agent

Chi-kin Chow and Hung-tat Tsui
Visual Signal Processing and Communications Laboratory
Department of Electronic Engineering
The Chinese University of Hong Kong
Hong Kong SAR, China
E-mail: {ckchow1, httsui}@ee.cuhk.edu.hk

Abstract – **In robot applications, the performance of a robot agent is measured by the quantity of award received from its response. Many literatures [1-5] define the response as either a state diagram or a neural network. Due to the absence of a desired response, neither of them is applicable to an unstructural environment. In this paper, a novel Response Knowledge Learning algorithm is proposed to handle this domain. By using a set of experiences, the algorithm can extract the contributed experiences to construct the response function. Two sets of environments are provided to illustrate the performance of the proposed algorithm. The results show that it can effectively construct the response function that receives an award which is very close to the true maximum.**

I. Introduction

The essentiality of self-improvement [6, 7] in control systems becomes popular in both structural and unstructural environments, such as robotics, factory automation, and autonomous vehicles. During the processing, an agent makes a response to the environment according to its observation. The response function R represents the agent's behavior and it is modeled as a function of observation from the environment: $\mathbf{p} = R(\mathbf{o})$ where $\mathbf{p}$ is the response vector and $\mathbf{o}$ is the observation vector. In the structural environment domain, state diagram is a popular representation of the response function. In this approach, each observation is defined as a state in which the weights among the states are trained by the statistical learning method such as Reinforcement Learning (RL) [1-4] and Hidden Markov Model (HMM) [5]. In this type of representation, since the observation states are finite and discrete, it is insufficient to describe the environment of real-world applications. For the continuous environment domain, regression approaches [8, 9] are used to model the agent response as a continuous function. Neural network is a common example [10]. Neural network (NN) is able to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. By considering the observations $\mathbf{o}$ and the corresponding desired response $\mathbf{r}$ as a training set $S = [\mathbf{o} \mid \mathbf{r}]$ of the NN, the continuous response function can be computed.

On the other hand, since there is no information about the unstructural environment initially, the desired response of any observation cannot be defined. Therefore, the traditional NN approaches are not applicable. Instead of providing the desired response, the environment only returns the award of a response, i.e. award = $A(\mathbf{o}, \mathbf{r})$ where A is the award function representing the characteristic of the environment. A typical example of the unknown environment application is the Prey and Predator (PP) problem, which is defined as follows:

In this model, the world consists of 2 organisms that form a prey-predator relationship. The predator aims at catching the moving prey with minimum number of steps. In addition, the predator has no knowledge on how to catch the prey initially. Therefore, it has to learn the strategy through a set of experiences which is expressed as:

- Observation o is the direction from prey.
- Response r is the movement direction of predator.
- Award a is the change of distance between them.

Due to the absence of catching strategy, the predator will randomly generate the response to catch the prey. It is expected that some of the responses cause the predator approaching to the prey and some do not. When a sufficient number of random trials are collected, the predator can extract the strategy (response function) from the experiences.

An experience $\mathbf{E} = [\mathbf{o}, \mathbf{r} \mid a]$ of an unknown environment consists of three parts: Observation $\mathbf{o}$, Response $\mathbf{r}$ and Award a. The objective of Response Knowledge Learning (RKL) is to determine a response function that returns a response with maximum award.

$$\text{Response Constriant}: A(\vec{o}, R(\vec{o})) \geq A(\vec{o}, \vec{r}) \text{ for all } \vec{r} \qquad \text{Eq. (1)}$$

This paper is organized as follows. The objective and the proposed RKL training algorithm of are discussed in section 2. In section 3, the architecture of a sequential training radial basis function regression network called "Memory Regression Network MRN" is introduced. Two environments are provided for response knowledge learning in Section 4, to illustrate the performance of the proposed algorithm. A conclusion is drawn in Section 5.

II. Response Knowledge Learning

In the robot applications, the agent keeps interact with the environment by applying a sequence of responses according to its current observation. In order to survive in the environment, the robot agent must make a response to the environment in which a maximum award is collected. The objective of RKL is to determine the response function that satisfies the response constraint described in previous experience. Different from the traditional NN training samples that all of them are benefit to the network; some of the RKL experiences are harmful to it. Therefore, the RKL should learn the function that receives maximum award and minimize the total punishment collected. For example, if an agent collects n experiences $\mathbf{E}_i = [\mathbf{o}, \mathbf{r}_i, a_i]$ for $i \in [1, n]$, only the experience with maximum award is selected for learning.

Since the environment is a black-box function, we have to estimate the target environment $A_e \approx A$ by a regression on the collect experience set $\{\mathbf{E}_i\}$. Moreover, as the estimated environment function A_e is constructed by $\{\mathbf{E}_i\}$, the response constraint Eq. (1) can be further simplified as:

$$A_e(\bar{o}_i, R(\bar{o}_i)) \geq A_e(\bar{o}_i, \bar{r}_i) = a_i \text{ for all } i \qquad \text{Eq. (2)}$$

In the proposed algorithm, both the estimated award function A_e and the response function R are represented as the neural networks, and named as *Award Network* and *Response Network* respectively. *Award Network* is supported by the active experience set (AES) and represents award of the estimated environment while the *Response Network* returns the expected response of a given observation. Both networks are constructed by a novel sequential training radial basis function (RBF) regression network called "Memory Regression Network (MRN)". The complete algorithm is summarized below:

Step 1: Award Function Estimation

By given a set of distinct description about the environment (Experience Set $\{\mathbf{E}_i\}$), we firstly estimate the environment model based on the sequential training network MRN whose details are discussed at Section 3. As more experiences are collected from the environment, a more accurate award function can be estimated and hence a better response function can be established.

Step 2: Maximum Violated Experience (MVE) Extraction

Given an observation $\mathbf{o}$, a well-trained response function should return the expected response $\mathbf{r}_e$ in which the corresponding award is maximized. Therefore, an experience $\mathbf{E}_i$ is regarded as violated from the response constraint if its expected award is smaller than the desired one, i.e. $a_i > A_e(\mathbf{o}_i, \mathbf{r}_e)$ where $\mathbf{r}_e = R(\mathbf{o}_i)$. The violated experience with maximum desired award is regarded as the MVE of current iteration.

Step 3: Retraining of Response Network

After determining the current MVE, it will be appended to the active experience set $\{\mathbf{AE}_i\}$:

$$\text{i.e. } \{\mathbf{AE}_i\} \leftarrow \{\mathbf{AE}_i\} \cup \mathbf{VE}$$

The expanded active experience set will be used to retrain the response network using MRN.

Step 4: Response Network Completeness

The response network is well trained if there is no violated experience extracted. The flow diagram of the proposed RKL algorithm is shown in fig. 1.

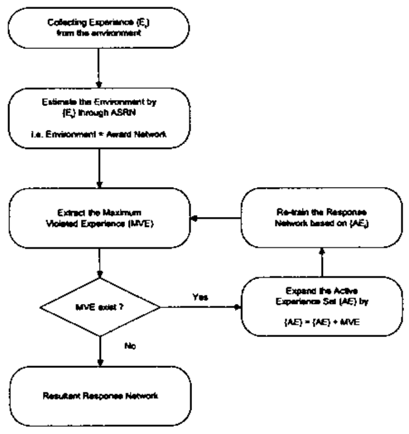

Collecting Experience (E_i) from the environment
Estimate the Environment by {E_i} through ASRN
i.e. Environment = Award Network
Extract the Maximum Violated Experience (MVE)
Re-train the Response Network based on {AE_i}
MVE exist ?
Yes
Expand the Active Experience Set (AE) by
{AE} = {AE} + MVE
No
Resultant Response Network

Figure 1 Flow diagram of the proposed RKL algorithm.

III. Memory Regression Network MRN

A. Architecture of MRN

MRN is a fixed topology RBF network. It involves two types of memory units:

1. *Experience* is a short-term memory unit (STMU), i.e. $\mathbf{P} = [\boldsymbol{\beta}, e]$ where $\boldsymbol{\beta}$ and e store the input $\mathbf{x}$ and output y of a training sample respectively.

2. *Neuron* is a long-term memory unit (LTMU), i.e. $\mathbf{H} = [\boldsymbol{\mu}, \sigma, M]$. Neuron is represented as a Gaussian function with the mean $\boldsymbol{\mu}$, variance σ and weight M.

B. Training Algorithm of MRN

As the present training sample $[\mathbf{x}_s \mid y_s]$ is collected, the MRN will perform one of the 3 training phases:

$$\text{Training Procedure} = \begin{cases} \text{Phase 1} & \text{if } N_s \le N_e \text{ and } N_s \le N_n \\ \text{Phase 2} & \text{if } N_s > N_n \\ \text{Phase 3} & \text{if } N_s > N_e \end{cases}$$

where N_e, N_n and N_s are the number of experience, neuron and collected samples respectively. The details of 3 phases are described as follows:

PHASE 1: SAMPLE MEMORIZING

The parameters of i^{th} experience $\mathbf{P}_i = [\boldsymbol{\beta}_i, e_i]$ are assigned as $\boldsymbol{\beta}_i \leftarrow \mathbf{x}_s$ and $e_i \leftarrow y_s$. In addition, the center and variance of i^{th} neuron $\mathbf{H}_i = [\boldsymbol{\mu}_i, \sigma_i, M_i]$ are assigned as:

$$\bar{\mu}_i \leftarrow \bar{x}_i \quad \sigma_i \leftarrow \sqrt{\frac{\min_{m \ne i} \left\| \bar{\mu}_i - \bar{\beta}_m \right\|}{-8 \ln 0.8}}$$

Consequently, the weights of neuron set $[\mathbf{M}]$ can be determined by the Matrix Inversion method, i.e. $[\mathbf{M}] = [\mathbf{G}]^{-1}[\mathbf{Y}]$ where $G_{i,j} = \exp(-\|\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\| / 2\sigma_j^2)$ and $\mathbf{Y} = [e_i]$.

PHASE 2: KNOWLEDGE GENERALIZATION

The parameters of i^{th} experience $\mathbf{P}_i$ are assigned as $\boldsymbol{\beta}_i \leftarrow \mathbf{x}_s$ and $e_i \leftarrow y_s$s so as to memorize the incoming sample. If there exists the neuron $\mathbf{H}_k = [\boldsymbol{\mu}_k, \sigma_k, M_k]$ such that:

1. $|M_k| \le |M_i|$ for all i
2. and $|M_k| \le E(\mathbf{x}_s, y_s)$

   where $E(\bar{x}_s, y_s) = y_s - \sum_{j=1}^{N_n} M_j \exp\left(- \frac{\left\| \bar{\mu}_j - \bar{x}_s \right\|}{2\sigma_j^2} \right)$

the parameters of $\mathbf{H}_k$ will be assigned as:

$$\bar{\mu}_k \leftarrow \bar{x}_s; \quad \sigma_k \leftarrow \sqrt{\frac{\min_{m \ne k} \left\| \bar{\mu}_k - \bar{\beta}_m \right\|}{-8 \ln 0.8}}; \quad M_k \leftarrow E(\bar{x}_s, y_s)$$

Due to the neuron adjustment process listed in the above, a Gaussian regression error region is occurred. Instead of adjusting all neurons to minimize the error, a subset of them (Excited neuron set $\{\mathbf{O}_i\} = \{\boldsymbol{\mu}_{o,i}, \sigma_{o,i}, M_{o,i}\}$) is extracted for error minimization. The procedures of extracting the excited neuron set are listed in below:

Step 1: Determine the Excitivity of i^{th} neuron (C_i):

$$C_i = \frac{V_i}{U_i}; \quad V_i = \int_0^1 e^{-\frac{\|\bar{\mu}_i - \bar{x}\|}{2\sigma_i^2}} e^{-\frac{\|\bar{\mu}_e - \bar{x}\|}{2\sigma_e^2}} d\bar{x}; \quad U_i = \int_0^1 e^{-\frac{\|\bar{\mu}_i - \bar{x}\|}{2\sigma_i^2}} d\bar{x}$$

where $\boldsymbol{\mu}_e$ and σ_e is the mean and variance of $\mathbf{H}_k$ before the adjustment.

Step 2: Sort $\{C_i\}$ in descending order.
Step 3: Assume that $R(i)$ is the rank of i^{th} neuron $\mathbf{H}_i$.
Step 4: By giving the Correlation constant η, $\mathbf{H}_i$ is defined as the excited neuron if:

$$\sum C_h \le \eta \sum_{j=1}^{p} C_j \text{ for } R(i) \ge R(h) \text{ where } p \text{ is the network size}$$

In addition, the experience $\mathbf{P}_k = [\boldsymbol{\beta}_i, e_i]$ is regarded as the element of excited experience $\{F_i\} = \{\beta_{e,i}, e_{e,i}\}$ if $\|\boldsymbol{\beta}_i - \boldsymbol{\mu}_{o,j}\| \le a\sigma_{o,j}$ for some j. By minimizing R defined at Eq. (1), the network with minimum error can be achieved.

$$R = \sum_{i=1}^{N_p} \left(e_{p,i} - \sum_{j=1}^{N_o} M_{o,j} \exp\left(- \frac{\left\| \bar{\mu}_{o,j} - \bar{\beta}_i \right\|}{2\sigma_{o,i}^2} \right) \right)^2 \qquad \text{Eq. (1)}$$

PHASE 3: KNOWLEDGE SELECTION

If there exists an experience $\mathbf{P}_k = [\boldsymbol{\beta}_k, e_k]$ such that

1. $E(\boldsymbol{\beta}_k, e_k) \le E(\boldsymbol{\beta}_i, e_i)$ for all i
2. and $E(\boldsymbol{\beta}_k, e_k) \le E(\mathbf{x}_s, y_s)$

$\boldsymbol{\beta}_k$ and e_k will be adjusted as $\mathbf{x}_s$ and y_s respectively.

IV. EXPERIMENTAL RESULTS

In this section, we apply the proposed RKL algorithm to extract the responses of two environments. Both environments are the function of 1-Observation and 1-Response:

$$A_1(o,r) = \exp\left(-0.2 \times \left| \exp(-\frac{(o-0.5)^2}{0.08}) - r \right| - \frac{(o-0.5)^2}{0.08} \right)$$

$$A_2(o,r) = \exp\left(-0.2 \times \left| \sin(2\pi o) - r \right| - \frac{(o-0.5)^2}{0.08} \right)$$

where o, $r \in [0, 1]$, and the corresponding desired Response functions are:

$$R_1(o) = \exp(-\frac{(o-0.5)^2}{0.08}$$
$$R_2(o) = \sin(2\pi o)$$

Figure 2 shows the intensity plot of desired award functions A_1 and A_2. The x-axis represents the observation while the y-axis represents the response of agent to the environment. The point with higher intensity indicates its relative larger award. In these experiments, 100 experiences are provided for learning (Fig. 3). The observations/response pairs of the experiences are randomly generated within the range G $\in$ [0, 1] while the corresponding awards are determined by the given award functions. Based on the experience set, the estimated award functions $A_{e,1}$ and $A_{e,2}$ are determined and shown in fig. 4. Tables 1 and 2 list the details of the estimated award functions by the MRN. It take 1 second to construct the award networks of A_1 and A_2 with 52 and 64 neurons respectively. Figure 5 shows the resultant response network output U_1 and U_2 together with the desired response of A_1 and A_2. It takes 2 and 3 seconds to construct the response network for the award function 1 and 2 respectively.

By given any observation **O**, the response network $U(\mathbf{O})$ should return a response with maximum award. Therefore, the performance of a response function can be evaluated from its sum of award P:

$$\text{i.e. } P = \int A(\bar{o}, U(\bar{o}))d\bar{o}$$

Tables 3 and 4 show the response network performance on both A_1 and $A_{e,1}$. It shows that the performance of response network is nearly the same of desired one. In order to illustrate the importance of RKL in robot applications, the performances of two reference response networks are studied for comparison. The first reference response network is called "Random Weight Response Network (RWRN)". The sizes of RWRNs are assigned as the same of U_1 and U_2. Figure 5 shows the examples of RWRNs. In each environment, 50 RWRNs are generated in which the average response performance is calculated. The results show that the award performances have nearly 100% improvements after the RKL process. The second reference response network is called "Max-n Response Network (Max-n)", which is constructed by training MRN with n highest award experiences. For each environment, we exhaustively search the value of n in which the response performance is maximum. Table 3 and 4 listed the values of n together with its response performances. It is shown that the performances of the best Max-n Response Networks are lower than that of the proposed one. All simulations are process at the PC platform with 1.7GHz CPU, and 256MB memory.

V. Conclusion & Discussion

In this article, we presented a novel RKL technique to construct a response function on unknown environment for solving an entirely new problem. The RKL consists 3 major processes: (1) Unknown Environment Modeling, (2) Active Experience Set (AES) Extraction and (3) Iterative Response Network construction. In the modeling process, the estimated award function is constructed by the collected experience set. Afterward, a set of active experiences is extracted by comparing its desired award with the one caused by the expected response. Consequently, the response network with relative maximum reward can be achieved through training the AES with a newly introduced sequential training network MRN. In order to verify the proposed algorithm, two 1-Observation 1-Response environment functions are evolved. The results show that the resultant response functions can receive a nearly maximum reward.

References

[1] Yamaguchi, T.; Masubuchi, M.; Fujihara, K.; Yachida, M.; "Realtime reinforcement learning for a real robot in the real environment", Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems '96, IROS 96, Volume: 3, 4-8 Nov. 1996, Pages: 1321 – 1328.

[2] Pack Kaelbling, L.; "On reinforcement learning for robots", Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems '96, IROS 96, Volume: 3, 4-8 Nov. 1996, Pages: 1319 – 1320.

[3] Howell, M.N.; Gordon, T.J.; "Associative reinforcement learning for discrete-time optimal control", Learning Systems for Control (Ref. No. 2000/069), IEE Seminar, 26 May 2000, Pages: 1/1 - 1/4.

[4] Kaitwanidvilai, S.; Parnichkun, M.; "Active Bayesian feature weighting in reinforcement learning robot", 2002 IEEE International Conference on Industrial Technology, 2002, IEEE ICIT '02, Volume: 2, 11-14 Dec. 2002, Pages: 1090 – 1095.

[5] Ogawara, K.; Takamatsu, J.; Kimura, H.; Ikeuchi, K.; "Modeling manipulation interactions by hidden Markov model", IEEE/RSJ International Conference on Intelligent Robots and System, 2002, Volume: 2, 30 Sept.-5 Oct. 2002, Pages: 1096 – 1101.

[6] Bentivegna, D.C.; Atkeson, C.G.; "Learning from observation using primitives", Proceedings of the IEEE International Conference on Robotics and Automation 2001, ICRA 2001, Volume: 2, 2001, Pages: 1988 – 1993.

[7] Enokida, S.; Ohashi, T.; Yoshida, T.; Ejima, T.; "Stochastic field model for autonomous robot learning", Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics 1999, IEEE SMC '99, Volume: 2, 12-15 Oct. 1999, Pages: 752 – 757.

[8] Schaal, S.; Atkeson, C.G.; "Robot learning by nonparametric regression", Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems '94, Advanced Robotic Systems and the Real World', IROS '94, Volume: 1, 12-16 Sept. 1994, Pages: 478 – 485.

[9] Atkeson, C.G.; "Using locally weighted regression for robot learning", Proceedings of the IEEE International Conference on Robotics and Automation 1991, ICRA 1991, 9-11 April 1991, Pages: 958 - 963.

[10] Gross, H.-M.; Stephan, V.; Krabbes, M.; "A neural field approach to topological reinforcement learning in continuous action spaces", IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on Neural Networks Proceedings, 1998, Volume: 3, 4-9 May 1998, Pages: 1992 – 1997.

TABLE 1
DETAILS OF RESULTANT AWARD NETWORK OF SECTION 4

Award Network	Number of Experiences	Computational Time (Sec.)	No. of non-zeros magnitude agent
A_1	100	1	52
A_2	100	1	64

TABLE 2
DETAILS OF RESULTANT RESPONSE NETWORK OF SECTION 4

Response Network	Computational Time (Sec.)	No. of non-zeros magnitude agent
U_1	2	10
U_2	3	18

TABLE 3
COMPARISON OF SUM OF AWARD AT SECTION4 AWARD FUNCTION 1

Response Network	Desired Award function	Estimated Award function
Random Weight	0.645333 (Average)	0.574367 (Average)
Max-n	0.76541 (n = 12)	0.70384 (n = 16)
The Proposed Algorithm	0.928238	0.802978
Desired	1.000000	0.824573

TABLE 4
COMPARISON OF SUM OF AWARD AT SECTION4 AWARD FUNCTION 2

Response Network	Desired Award function	Estimated Award function
Random Weight	0.446420 (Average)	0.408869 (Average)
Max-n	0.62385 (n = 23)	0.57625 (n = 32)
The Proposed Algorithm	0.890513	0.743906
Desired	1.000000	0.792052

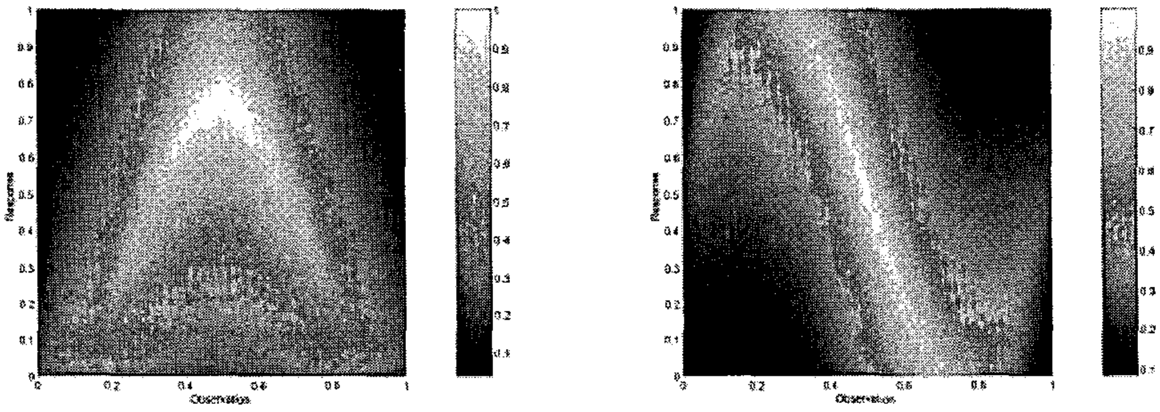


Figure 2 The desired Award Surfaces of A_1 and A_2 at Section 4

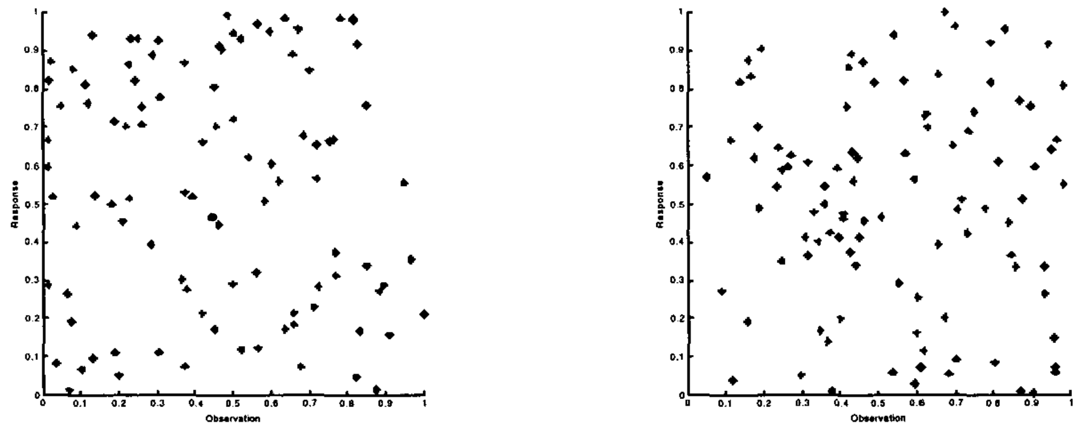
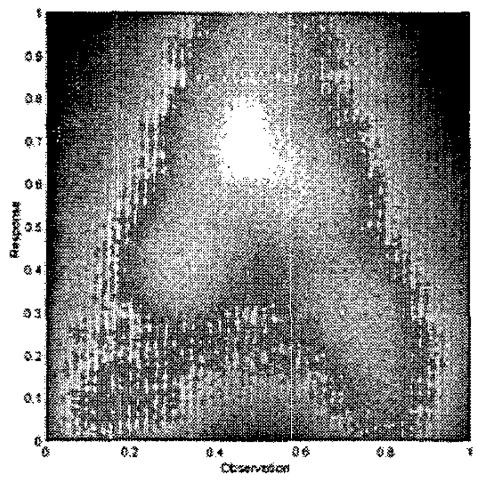


Figure 3 Distribution of experience sets of A_1 and A_2 at Section 4



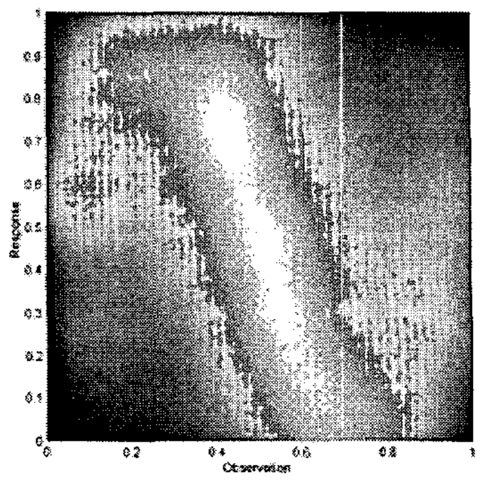
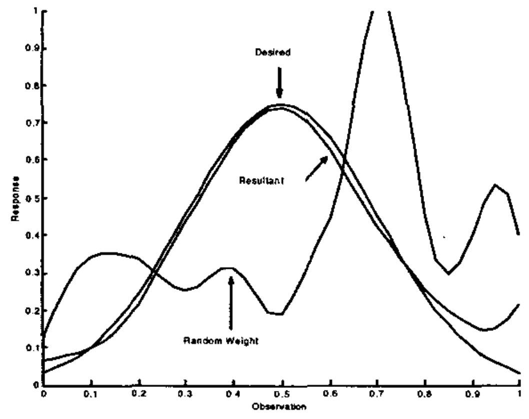


Figure 4 The estimated award surfaces of A_1 and A_2 at Section 4



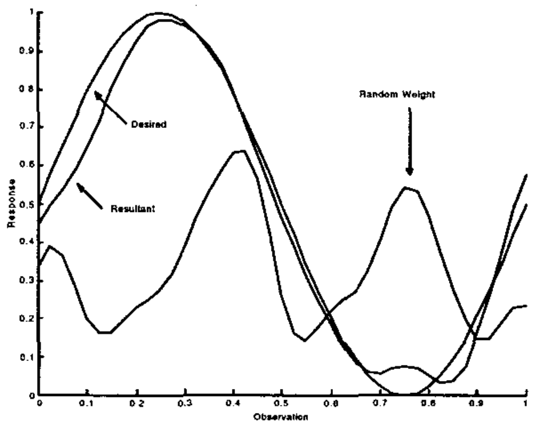


Figure 5 Comparison among the Desired, Radon Weight and Resultant Response network outputs of A_1 and A_2 at Section 4