Learning Individual Motion Preferences From Audience Feedback of Motion Sequences

Junyun Tay[1,2], Manuela Veloso[1] and I-Ming Chen[2]

***Abstract*— A robot performs a sequence of motions to animate a given input, e.g., dancing to music or telling a story. Each input is pre-processed to determine labels, e.g., emotions of the music or words in the story. Each label corresponds to multiple motions, and each motion has multiple labels. Therefore, the robot can choose one sequence from multiple motion sequences to animate the input. We aim to choose the best sequence to animate based on the audience's preferences. The audience prefers some motions over others, and each motion has an initially unknown preference value. At the end of the motion sequence, the audience provides feedback which is the sum of the motions' preference values. However, the observation of the feedback is noisy due to the device used to capture the audience's feedback. To select the most preferred sequence, the robot has to determine the sequence to query the audience with, so as to learn the preference values of individual motions from noisy observations of the audience's feedback. By learning the individual motion preference values, the most preferred sequence can be determined. Moreover, the audience may get bored of watching the same single motion in multiple sequences and the preference value will degrade based on the number of times the motion is viewed. We contribute MAK (Multi-Armed bandit and Kalman filter) and show that MAK outperforms least squares regression in selecting the best sequence with lower degradation in our simulation experiments.**

I. Introduction

A robot animates a pre-processed input signal with a sequence of motions autonomously. For example, a robot dances to music [1] or animates a story [2] autonomously. Given a library of labeled motions where multiple motions are mapped to a label, multiple motion sequences are feasible to animate the input signal. In this paper, we aim to select the sequence which is most preferred by the audience through the feedback of some motion sequences.

Asking the audience to rate their preferences for all sequences or all motions in the sequences is not feasible, because the audience may get bored as they view the same motions repeatedly. We model this effect with a degradation factor based on the number of times a motion is viewed.

We assume that the audience feedback provided at the end of a motion sequence is consistent and the feedback is computed using the sum of the ratings of individual motions in the sequence. However, the observation of the audience feedback at the end of a motion sequence is noisy, i.e., the measurement using a device to capture the feedback is noisy. For example, capturing the volume of the audience's claps via a microphone or capturing the audience's feedback via colored cues with a camera will be inaccurate due to background noise such as chatting or some colored shirts.

Our approach – Multi-Armed bandit and Kalman filter (MAK) is made up of two steps. First, we use a multi-armed bandit algorithm to select the motion sequence to query feedback on and a Kalman filter to estimate the ratings of the individual motions using the feedback. We stop MAK once the stopping conditions we defined are met and show that we are able to determine the best sequence without going through all possible sequences.

We compare our MAK approach against least-squares regression, by using a black box that contains the ratings for the individual motions in a simulation. Through simulation, we ensure that the preference values of individual motions do not vary except when degradation is considered and the feedback of a particular motion sequence is consistent, but observed with noise. Multiple motion sequences can express the labels of the pre-processed input. However, we can only query the black box with a sequence of motions and receive a noisy rating of the sequence. We show that our MAK approach outperforms the least-squares regression in different scenarios, e.g., the audience degrades or does not degrade the rating of a motion each time it is viewed.

II. Related Work

Feedback from the audience ranges from using visual cues such as the audience holding colored markers such as paddles [3] or audio feedback such as the applause or cheers from the audience or surveys at the end of the interaction [4]. Knight et al. model the audience using the features of jokes and selects the best joke for a robot to tell using the audience feedback and the features of the joke [3]. We use the feedback on sequences to model the ratings of individual motions to select the most preferred motion sequence. Knight et al. assume that the audience preference on features do not vary over time, whereas we model the boredom of the audience of viewing the same motion. Addo and Ahamed also use a robot to tell jokes, but use reinforcement learning [4]. However, to learn a good policy, they have to explore all jokes in all the states, whereas we do not have to query all motion sequences to pick the best sequence.

Abbeel and Ng introduce inverse reinforcement learning, where the reward function is unknown and that it is difficult to specify a reward function, but the "unknown reward function can be expressed as a linear combination of known features" [5]. Our approach is similar in the sense that the

[1]Junyun Tay and Manuela Veloso are with Carnegie Mellon University, Pittsburgh PA, USA, junyunt@andrew.cmu.edu, veloso@cs.cmu.edu

[2]Junyun Tay and I-Ming Chen are with Nanyang Technological University, Singapore, junyunt@andrew.cmu.edu, michen@ntu.edu.sg

audience rating of a sequence is expressed as a sum of the unknown ratings of single motions. We also do not require a Markov decision process made up of states and actions to model the audience preferences and determine the best policy for each state. Instead, we model the ratings of single motions and account for noisy observations.

Akrour et al. explore the use of a Markov decision process and rank sequences of state-action pairs based on the preferences instead of assigning values to the sequences [6]. We do not order preferences through ranking as the magnitude of how much a sequence is preferred over another sequence is lost in the ranking of sequences.

Our approach MAK uses the multi-armed bandit algorithm and Kalman filter. The multi-armed bandit problem is a well-known problem, where the goal is to select the arm to pull to maximize the sum of expected rewards, and Thompson sampling [7] is one of the common algorithms used to optimize the arm-pulling. The multi-armed bandit problem was recently applied to allocating training instances to learning agents, so as to estimate their learning rates and maximize the team performance [8], [9], and Kalman filters were used to estimate the agents' learning rates. In this paper, we use a single Kalman filter to estimate the preference values for single motions, and we use a multi-armed bandit algorithm to select a motion sequence to query for feedback.

III. Problem Description and Assumptions

In this section, we describe the motivating scenarios, and present the formal problem definition and assumptions.

Motivating Scenario

Suppose that a humanoid robot is tasked with animating a story. The story comprises sentences, where each sentence is pre-processed for labels and the labels in each sentence can be animated, e.g., in "John *waved* at the *bird*", the words in italics are *labels* that are be animated.

For each label in a sentence, there may be multiple motions that are applicable, e.g., to animate "*waved*", the robot can wave with its left/right arm, with its palm open/closed, and move quickly/slowly. As such, for each sentence, there are multiple unique sequences of motion that are feasible.

Each *label-motion pair* (e.g., *"waved"*-Wave Slowly With Open Left Hand) has a unique audience preference value, and as such, each sequence of motions for the sentence has an audience preference value, summed over each motion used in the sequence. Further, the audience preference value may degrade each time the audience views a motion, as the audience may get increasingly bored with seeing the same motion multiple times. The goal is to select the sequence of motions for the sentence with the highest audience preference value, while minimizing the number of times the audience is queried so that the degradation is minimal.

Formal Problem Definition

Each label can be mapped to multiple motions and each motion can be mapped to multiple labels.

Definition 3.1: Let m_k be the k^{th} unique label-motion pair, and M be the set of all label-motion pairs.

There exist different sequences of motions for the robot to animate the input s, where the labels of the input match the corresponding labels in the motions and the motions are synchronized to the starting times of the labels in the input.

Definition 3.2: Let $u^s = (m_1, \ldots, m_D)$ be an ordered set of D label-motion pairs for a pre-processed input s, where $D \geq 2$. Let U^s be the set of motion sequences for s.

Though the audience rates each motion sequence consistently based on the motions used in the sequence, the observation of the audience's preference value of the sequence is noisy. The cost of querying the audience for each motion used when there are many possible motions is also higher. Moreover, the audience may get bored from repeatedly viewing the same motion or the robot's animation for the same input. If the preference value for single motion remains consistent across different inputs and we know all the preference values for single motions, we are able to determine the preference value for all motion sequences.

Definition 3.3: Let a_k be the audience preference rating of a label-motion pair m_k. The audience rating of a sequence of motions is $\mathbb{A} : U^s \rightarrow \mathbb{R}^+$, where $\mathbb{A}(u^s) = \sum_{m_k \in u^s} a_k$.

Definition 3.4: Let α_i be the noisy observation of the rating for sequence u_i^s (the i^{th} sequence for input s), i.e., $\alpha_i \sim \mathcal{N}(\mathbb{A}(u_i^s), R_k)$ for some noise variance R_k.

The goal is to find the best sequence, i.e., $\text{argmax}_i\, \mathbb{A}(u_i^s)$.

One approach would be to repeatedly try all possible sequences multiple times to determine the best sequence and account for the noise in the observation. However, there are two issues. If there are many sequences, the audience may get bored or refused to participate in rating the same input. Moreover, if a motion is repeated in different sequences, people get bored when viewing the same animation multiple times and degrade the preference value for the motion based on the number of times the motion is viewed.

As such, we define a model that simulates the effects of boredom, when viewing a label-motion pair repeatedly. We term it the degradation model, where the rating for an individual label-motion pair in the sequence degrades by a factor each time the individual label-motion pair is viewed. This degradation to the rating means that the audience prefers the individual label-motion pair a little less each time the label-motion pair is seen.

Definition 3.5: Let the degradation factor be $\delta \in [0, 1]$.

Assumptions

- The rating for each label-motion pair is independent.
- The observation noise R_k is known.
- The degradation factor δ is known.

IV. Technical Approach

We model the problem as a multi-armed bandit, where each arm represents a motion sequence. At each iteration, we pull an arm and observe a noisy rating of the sequence. Next, we use a Kalman filter to estimate the preference values of the individual label-motion pairs by using the series of noisy observations for different sequences over time.

We model the preference value of each label-motion pair as a distribution with a mean and variance and term preference value as the rating. The lower the variance of the estimated rating (preference value), the more confident we are about the mean ("true value") of the rating.

Definition 4.1: Let $\widetilde{a}_i$ be the estimate of the rating of the label-motion pair m_i and the mean of the rating in our model. Let $\widetilde{v}_i$ be the variance of the estimated rating of the label-motion pair m_i. Let $\widetilde{A}_t$ be the set of estimated ratings for all label-motion pairs, M, at iteration t and $\widetilde{V}_t$ be the set of variances for all label-motion pairs at iteration t.

The ratings of the individual label-motion pairs are modeled as the state variables in the Kalman filter, and the observation is the observed rating for the motion sequence.

At each iteration, we determine the motion sequence to query (arm to pull) using Thompson Sampling, a multi-armed bandit algorithm, which in turn uses the Kalman's estimated states to determine the rating of the sequence. We repeat the process of choosing the motion sequence to query using Thompson Sampling and update our estimates of the individual ratings of the label-motion pairs using the Kalman filter till we reach a stopping condition. We term our approach "MAK" - **M**ulti-**A**rmed bandit and **K**alman filter.

Definition 4.2: Let $\widetilde{a}_i^t$ be the estimated rating of the label-motion pair m_i at iteration t, where $\widetilde{a}_i^t \in \widetilde{A}^t$. Let $\lambda_i = |\widetilde{a}_i^t - \widetilde{a}_i^{t-1}|$ be the absolute difference between the estimated rating of the label-motion pair m_i at iteration t and $t-1$.

Definition 4.3: Let $\widetilde{v}_i^t$ be the variance of the label-motion pair m_i at iteration t, where $\widetilde{v}_i^t \in \widetilde{V}^t$. Let $\lambda_i^v = |\widetilde{v}_i^t - \widetilde{v}_i^{t-1}|$ be the absolute difference between the estimate of the rating of the label-motion pair m_i at iteration t and $t-1$.

MAK will stop when either of the following two conditions is met: (1) the maximum iterations κ has occurred, or (2) the maximum absolute change in the current estimated rating of the label-motion pairs and the previous estimated rating of the label-motion pairs is less than or equals to ϵ, i.e., $\max_i(\lambda_i, \lambda_i^v) \leq \epsilon$, where $i \in \{1, 2, \ldots, |\widetilde{A}|\}$.

MAK Algorithm

We present the algorithm for MAK in Algorithm 1. We will discuss the initialization of $\widetilde{A}, \widetilde{V}$ in the Experiments section. Algorithm 1 uses both Algorithm 2 and Algorithm 3.

Algorithm 2 is a multi-armed bandit algorithm that determines the sequence to query based on the means and variances of the label-motion pairs, $\widetilde{A}, \widetilde{V}$. In Algorithm 2, we use Thompson Sampling as an example.

Algorithm 3 uses the Kalman filter to estimate the rating for each label-motion pair based on the noisy observation of the rating of the label-motion pairs in sequence u_c^s.

We illustrate how Algorithm 1 works with an input s consisting of the following labels (l_1, l_2). There are four possible sequences – $u_1^s, u_2^s, u_3^s, u_4^s$ with a motion library of four label-motion pairs – $m_1 = (l_1, m_1), m_2 = (l_1, m_2), m_3 = (l_2, m_3), m_4 = (l_2, m_4)$. The sequences have the following label-motion pairs:

- $u_1^s = (m_1, m_3)$
- $u_2^s = (m_1, m_4)$

Algorithm 1 Determine the best sequence u

MAK($U^s, \widetilde{A}, \widetilde{V}$)
  $t \leftarrow 0$
  $\Delta \leftarrow$ Infinity
  while $(t \leq \kappa)$ **and** $(\Delta > \epsilon)$ **do**
    $P_t \leftarrow \text{diag}(\widetilde{V})$ // P_t is a diagonal matrix where the values are the variances of the label-motion pairs
    $u_c^s \leftarrow$ MAB($U^s, \widetilde{A}, \widetilde{V}$) // Algo. 2
    $\alpha_c \sim \mathbb{N}(\mathbb{A}(u_c^s), R_k)$ // α_c is the noisy rating of u_c^s
    $[\widetilde{A}_t, \widetilde{V}_t] \leftarrow$ Kalman($\widetilde{A}, \widetilde{V}, u_c^s, \alpha_c, P_t$) // Algo. 3
    $\Delta \leftarrow \max_i(\lambda_i, \lambda_i^v)$
    $\widetilde{A} \leftarrow \widetilde{A}_t$
    $\widetilde{V} \leftarrow \widetilde{V}_t$
    $t \leftarrow t + 1$

Algorithm 2 Determine the next sequence to query based on the means and variances of the label-motion pairs using a multi-armed bandit algorithm – Thompson Sampling

MAB($U^s, \widetilde{A}, \widetilde{V}$)
  $v_{\max} \leftarrow 0$
  for $i = 1$ **to** $\mid U^s \mid$ **do**
    $v_i \leftarrow 0$
    for $j = 1$ **to** $\mid u_i^s \mid$ **do**
      $v_i \leftarrow v_i +$ Random($\widetilde{a}_j, \widetilde{v}_j$) // Random samples from a distribution with a mean, $\widetilde{a}_j$ and variance, $\widetilde{v}_j$
    if $v_{\max} < v_i$ **then**
      $v_i = v_{\max}$
      $u_{\max}^s \leftarrow u_i^s$
  return $u_{\max}^s$

Algorithm 3 Kalman filter where the states are the estimates of the ratings of the individual label-motion pairs, $\widetilde{A}$

Kalman($\widetilde{A}, \widetilde{V}, u_c^s, \alpha_c, P_t$)
  $F_t \leftarrow$ getStateTransition(IndicateLMUsed(u_c^s), δ) // F_t is a diagonal matrix, where 1 is for the label-motion pair(s) not used in u_c^s and δ for the label-motion pairs used in u_c^s
  $\hat{x}_{t|t-1} \leftarrow F_t \widetilde{A}$ // Predicted state estimates
  $H_t \leftarrow$ getObservationModel(IndicateLMUsed(u_c^s)) // H_t is a vector that indicates the label-motion pairs in u_c^s
  $P_{t-1|t-1} \leftarrow \text{diag}(\widetilde{V})$
  $P_{t|t-1} \leftarrow F_t P_{t-1|t-1} F_t^T$ // Predicted covariance estimates
  $\tilde{y}_t \leftarrow \alpha_c - H_t \hat{x}_{t|t-1}$ // Innovation
  $s_t \leftarrow H_t P_{t|t-1} H_t^T + R_t$ // Innovation covariance
  $K_t \leftarrow P_{t|t-1} H_t^T S_t^{-1}$ // Optimal Kalman gain
  $\hat{x}_{t|t} \leftarrow \hat{x}_{t|t-1} + K_t \tilde{y}_t$ // Updated state estimate
  $P_{t|t} \leftarrow (I - K_t H_t) P_{t|t-1}$ // Updated estimate covariance and I is an identity matrix
  $\widetilde{A}_t \leftarrow \hat{x}_{t|t}$
  $\widetilde{V} \leftarrow$ extractVariance($P_{t|t}$)
  return $[\widetilde{A}, \widetilde{V}]$

- $u_3^s = (m_2, m_3)$
- $u_4^s = (m_2, m_4)$

The estimated ratings of the label-motion pairs in our model are $\widetilde{A} = (\widetilde{a}_1, \widetilde{a}_2, \widetilde{a}_3, \widetilde{a}_4)$ and we initialize the estimated ratings to some values at the start. We also initialize $\widetilde{V}$ with a large number since we are not confident about $\widetilde{A}$.

Using Algorithm 2 where Thompson Sampling is used as an example, we sample values using the function Random that uses the model of the estimated rating of the label-motion pairs – $\widetilde{A}$ and $\widetilde{V}$. We compute the sum of these values for each sequence and return the sequence with the highest sampled value. Next, we observe a noisy audience rating, α_c, of the sequence u_c^s using the function $\mathbb{A}$.

We use the observation α_c in Algorithm 3, where the Kalman filter uses the observation to update the estimates of the estimated ratings of the label-motion pairs in our model.

Different sequences are made up of different label-motion pairs.The function IndicateLMUsed in Algorithm 3 takes in a sequence u_c^s and returns a vector whose values indicate if the corresponding unique label-motion pair $m_i \in M$ is used in the sequence u_c^s. If the i^{th} value in the vector is 1, m_i is used in the sequence u_c^s, otherwise the value is 0. For example, the function IndicateLMUsed(u_1^s) returns a vector with values $\begin{bmatrix}1 & 0 & 1 & 0\end{bmatrix}$.

The state transition matrix F_t depends on the label-motion pairs used in the sequence and the degradation factor δ. The function getStateTransition returns a matrix F_t. If u_1^s is being observed and $\delta = 1$, F_t is $\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$. If u_2^s is being observed and $\delta = 0.999$, F_t is $\begin{bmatrix}0.999 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0.999\end{bmatrix}$.

H_t is the observation model which maps the true state space into the observed space. Therefore, H_t depends on the label-motion pairs used in the sequence. For sequence u_3^s, the function getObservationModel returns $H_t = \begin{bmatrix}0 & 1 & 1 & 0\end{bmatrix}$.

We form a covariance matrix using the function diag where the covariance matrix is a diagonal matrix, and the diagonals are the respective $\widetilde{v}_i \in \widetilde{V}$. Algorithm 3 updates the estimates of the mean $\widetilde{A}$ and variance $\widetilde{V}$ for the unique label-motion pairs. We use the function extractVariance to extract the diagonals of the matrix $P_{t|t}$ to determine $\widetilde{V}$.

We use the function $\max_i(\lambda_i, \lambda_i^v)$ to determine Δ and update our model of $\widetilde{A}$ and $\widetilde{V}$. We repeat these steps till we reach the maximum number of iterations κ or $\Delta \leq \epsilon$. The maximum number of iterations is defined by the user and is less than the total number of possible sequences. The maximum number of iterations is to terminate the algorithm in the event that ϵ defined is too small.

V. Comparison – Least Squares Regression

Given that we know the label-motion pairs used in each sequence and the multiple noisy observations that we can make for each sequence, we consider least squares regression as the baseline comparison to estimate the ratings of the individual label-motion pairs.

Least squares regression uses the equation $Ax = B$. x is a $|M| \times 1$ vector containing the list of ratings for each label-motion pair. A is a $n \times |M|$ matrix that indicates the label-motion pairs used in n observed sequences. B is a $n \times 1$ vector containing the noisy observations for n sequences.

Similarly, we illustrate least squares regression with the same example for MAK, where the four possible sequences are $u_1^s, u_2^s, u_3^s, u_4^s$ and we have a motion library of four label-motion pairs – m_1, m_2, m_3, m_4.

For this example, $x = \begin{bmatrix}\widetilde{a}_1\\ \widetilde{a}_2\\ \widetilde{a}_3\\ \widetilde{a}_4\end{bmatrix}$.

When $\delta = 1$ and we observe the four possible sequences in order, $A = \begin{bmatrix}1 & 0 & 1 & 0\\ 1 & 0 & 0 & 1\\ 0 & 1 & 1 & 0\\ 0 & 1 & 0 & 1\end{bmatrix}$.

If $\delta < 1$, we keep track of the number of times the unique label-motion pair is used. We define the number of times the unique label-motion pair, m_i is viewed as v^i. Each row of A will contain the values IndicateLMUsed$(u_c^s) \cdot \delta^{\max(0, v^i - 1)}$, where i is the index of the label-motion pair in sequence u_c^s.

Least squares regression estimates the rating of the individual label-motion pair in the event that there is no observation noise of the audience preference of a sequence. Given p unique motion labels, we need at least p sequences that include all unique label-motion pairs to determine the individual audience preference values. However, since the observation is noisy, we will need at least $30p$ sequences using the Central Limit Theorem in order to get a good estimate of the ratings of the individual label-motion pairs.

Therefore, for least squares regression, we randomly pick p sequences at the start of each trial. These p sequences include all unique label-motion pairs so that the equations formed are not under-constrained. Following that, to determine the next sequence to query, we select the sequence u_{max}^s with the highest cumulative sum using the least squares estimates of the ratings of the label-motion pairs. We obtain a noisy observation of the rating α_{max} to add to B.

We continue adding rows to the matrix A and the vector B till one of the two stopping conditions is met. The first stopping condition is that the maximum iterations κ has occurred and $\kappa \geq |M|$, where $|M|$ is the number of unique label-motion pairs used in all possible sequences U^s. The second stopping condition is that the maximum absolute change in the current estimated rating of the label-motion pairs and the previous estimated rating of the label-motion pairs is less than or equals to ϵ. The second condition uses the equation $\max_i(\lambda_i) \leq \epsilon$, where $i \in \{1, 2, \ldots, |\widetilde{A}|\}$.

VI. Experimental Setup

We consider two models of the audience – Constant and Degradation. Constant is the model where the rating of a label-motion pair remains constant regardless of the number

of times the label-motion pair is viewed, so $\delta = 1$. The model, Degradation, is the model where the rating for a label-motion pair degrades by a constant known factor each time it is seen. Any value between 0 to 1 can be used for δ and we set $\delta = 0.999$ in our experiments.

To evaluate the performance of our approach, MAK, versus the baseline comparison of least squares regression, we created four labels with ten unique label-motions per label, resulting in a total of forty unique label-motion pairs. We also generated a black box where the ratings for the unique label-motion pairs are uniformly randomly generated from 0 to 100 and are hidden from our model of the ratings of the label-motion pairs. The number of possible sequences is based on the number of labels in the input if we assume that there are 10 motions per label. For example, if there are n labels, there are 10^n possible sequences.

We query the black box for the audience rating using the function $\mathbb{A}$ and return a noisy value that is computed based on a sum of the ratings of the label-motion pairs in the sequence. The noise added to the observation is $R_k = 100$. We used the stopping conditions of $\kappa = 100$ and $\epsilon = 0.1$.

We compared MAK against Least Squares regression for each experiment and ran 30 trials for each experiment since there is randomness in the sequences selected for queries. We initialized each label-motion pair with $\widetilde{a}_i = 0$ and $\widetilde{\nu}_i = 100^2$. We varied the following variables:

- Number of label-motion pairs in a sequence: We varied the input by changing the number of label-motion pairs in a sequence. We considered inputs with 2 label-motion pairs, 3 label-motion pairs and 4 label-motion pairs.
- Audience model: We conducted experiments with the two models – Constant where $\delta = 1$ and Degradation where $\delta = 0.999$.

For numerical computations involving matrices, we used GNU Octave. We used Octave's `lsqnonneq` function for least squares regression and the inputs are A, B and the initial ratings for the label-motion pairs as the initial guess.

VII. Experimental Results

Figure 1 shows the result for the experiment where each sequence has two label-motion pairs with a Constant audience model. The X axis shows the number of iterations. The Y axis shows the average rating of the best motion sequence for the corresponding approach at the t^{th} iteration and is averaged across 30 trials. The highest rating for the best sequence in the black box is plotted with a black dashed line and labeled as "Best". Our approach, "MAK", is labeled in blue as compared to "LeastSquares" in red using least squares regression. Figure 1 shows that "MAK" performs better than "LeastSquares" in terms of finding the best sequence as "MAK" converges to "Best", whereas there is a gap between "LeastSquares" and "Best".

Next, we plot the result for the experiment where each sequence has two label-motion pairs for the Degradation audience model in Figure 2. The highest audience rating for the best sequence in the black box using the approach MAK is labeled as "MAKBest" and the highest audience

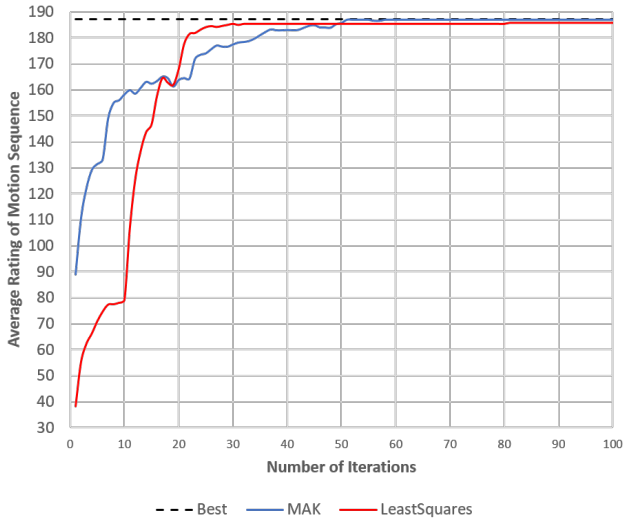

Fig. 1: MAK versus Least Squares for Constant Audience Model

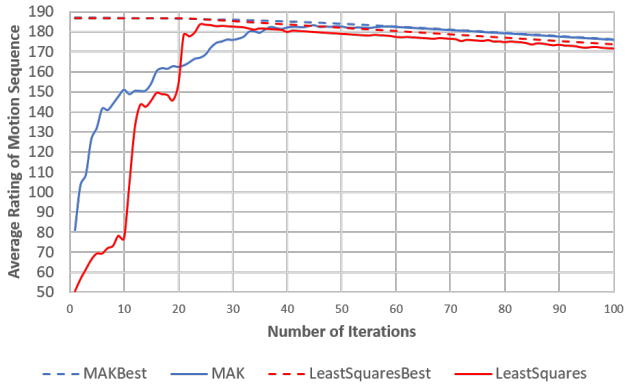

Fig. 2: MAK versus Least Squares For Degradation Audience Model

rating for the best sequence in the black box using the least squares approach is labeled as "LeastSquaresBest". The "Best" value is shown separately for MAK and Least Squares as the number of times different label-motion pairs in the best sequence is queried varies, therefore "MAKBest" and "LeastSquaresBest" have different values.

In Figure 2, MAK degrades less than Least Squares though they use the same degradation factor of 0.999, which shows that MAK is able to account for the degradation factor by selecting sequences that have different label-motion pairs to learn the preference values for individual label-motion pairs. Thus, MAK avoids revisiting the same label-motion pair in the best sequence and "MAKBest" degrades less than "LeastSquaresBest". Figure 2 shows that "MAK" converges to "MAKBest" after 57 iterations, where the absolute difference in the average rating between "MAK" and "MAKBest" is less than 1. "LeastSquares" does not converge to "LeastSquaresBest" at all.

For the experimental results shown in Figure 1 and Fig-

ure 2, we stopped after 100 iterations instead of using the stopping conditions we defined. By doing so, we showed that our approach, MAK, performs better and we know that MAK and Least Squares would converge before 100 iterations shown by our empirical results. However, as we increase the number of label-motion pairs in the experiments, we use the stopping conditions defined so that we do not need to check convergence empirically. Lastly, we present numerical results for the experiments "Comparison of two / three / four label-motion pairs" with the Constant and Degradation audience model in Table I.

We define two measures in Table I:

Definition 7.1: Let the best sequence found by the respective approach (either MAK or Least Squares) be $\widehat{u^*}$ and the best sequence in the black box be u^*. Let the absolute difference between the estimated rating (the ratings of the label-motion pairs in the model using the respective approach) of $\widehat{u^*}$ and the true rating (the ratings of the label-motion pairs in the black box) of $\widehat{u^*}$ be Υ. Let the absolute difference between the true ratings of u^* and $\widehat{u^*}$ be ρ.

For the experiment "Comparison of two / three / four label-motion pairs", we show in Table I that regardless of the number of labels, MAK always finds the best sequence since ρ is 0 for the constant audience model and $\rho \approx 0$ for the degradation model, but Least Squares is unable to find the best sequence given that $\rho > 0$. The model of the ratings of the individual label-motion pairs takes longer to converge for MAK compared to Least Squares, but Least Squares is unable to find the best motion sequence.

Since the number of iterations also refers to the number of times sequences are queried and the number is much less than the possible number of motion sequences for two/three/four labels, we show that we do not have to query all motion sequences for either MAK or least squares regression.

TABLE I: Performance of MAK versus Least Squares

Audience Model	**Label-motion pairs**	**Approach**	**Iterations**	Υ	ρ
Constant	2	MAK	70.3 ± 6.7	2.0 ± 1.4	0 ± 0
		Least Squares	33.2 ± 6.5	1.9 ± 1.2	3.8 ± 5.9
	3	MAK	107.3 ± 9.2	2.0 ± 1.8	0 ± 0
		Least Squares	44.9 ± 6.4	2.8 ± 2.4	6.8 ± 10.8
	4	MAK	162.3 ± 12.3	2.0 ± 1.6	0 ± 0
		Least Squares	56.5 ± 7.1	2.3 ± 2.0	3.9 ± 6.1
Degradation	2	MAK	70.0 ± 7.7	2.8 ± 1.9	0.2 ± 0.0
		Least Squares	38.4 ± 10.7	2.2 ± 2.0	2.8 ± 7.0
	3	MAK	115.0 ± 11.7	1.8 ± 1.9	0.6 ± 2.1
		Least Squares	68.3 ± 32.1	1.6 ± 1.4	2.0 ± 5.4
	4	MAK	172.4 ± 16.4	2.3 ± 1.8	1.1 ± 0.8
		Least Squares	81.1 ± 32.2	1.6 ± 0.9	4.0 ± 4.8

VIII. Conclusion

We show that MAK selects the best motion sequence without querying all possible motion sequences. MAK performs better than least squares regression in terms of selecting the best motion sequence and is capable of using noisy observations of the ratings for different motion sequences and consider degradation.

MAK takes more iterations than Least Squares to converge but Least Squares is unable to find the best motion sequence. Least Squares performs worse in the Degradation audience model and continuously picks motion sequences with similar label-motion pairs since LeastSquaresBest decreases as the number of iterations increase. MAK considers degradation given a Degradation audience model by selecting different motion sequences and is able to find the best motion sequence.

Least Squares may perform better if we randomly choose other motion sequences instead of the best motion sequence known in the model to query, similarly to doing random restarts to avoid getting stuck in local optimal. We note the problem of getting stuck in local optimal is accounted for in MAK as we model the confidence of the individual label-motion pair's rating.

Acknowledgments

Junyun Tay is part of the NTU-CMU Dual PhD Programme in Engineering (Robotics) which is funded by the Economic Development Board of Singapore. We thank Somchaya Liemhetcharat for his feedback. The views and conclusions contained herein are those of the authors only.

References

[1] G. Xia, J. Tay, R. Dannenberg, and M. Veloso, "Autonomous robot dancing driven by beats and emotions of music," in *Autonomous Agents and Multiagent Systems (AAMAS)*, vol. 1, 2012, pp. 205–212.

[2] J. Tay and M. Veloso, "Modeling and composing gestures for human-robot interaction," in *IEEE Int. Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2012, pp. 107–112.

[3] H. Knight, S. Divvala, S. Satkin, and V. Ramakrishna, "A savvy robot standup comic: Online learning through audience tracking," in *Fifth Int. Conf. on Tangible, Embedded and Embodied Interaction*, 2011.

[4] I. D. Addo and S. I. Ahamed, "Applying affective feedback to reinforcement learning in zoei, a comic humanoid robot," in *IEEE RO-MAN*, 2014, pp. 423–428.

[5] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Twenty-first Int. Conf. on Machine Learning*, 2004.

[6] R. Akrour, M. Schoenauer, and M. Sebag, "APRIL: Active preference learning-based reinforcement learning," in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, vol. 7524. Springer, 2012, pp. 116–131.

[7] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[8] S. Liemhetcharat and M. Veloso, "Team formation with learning agents that improve coordination," in *AAMAS*, 2014, pp. 1531–1532.

[9] S. Liemhetcharat and M. Veloso, "Allocating training instances to learning agents that improve coordination for team formation," in *AAMAS workshop on Autonomous Robots and Multirobot Systems (ARMS)*, ser. AAMAS '14, 2014, pp. 1531–1532.