MAB: variable choice

Hazel Murray and David Malone

December 2020

Abstract

This is supplementary material for our work on the multi-armed bandit approach to guessing.

1 Choosing variables in the multi-armed bandit model

By creating and guessing simulated password sets we have been able to test the various variables within the set up of our multi-armed bandit model. In this section we will discuss each variable type and discuss the value of this variable that we found worked most effectively. In future experiments, this will allow us to generate results with the model set up effectively.

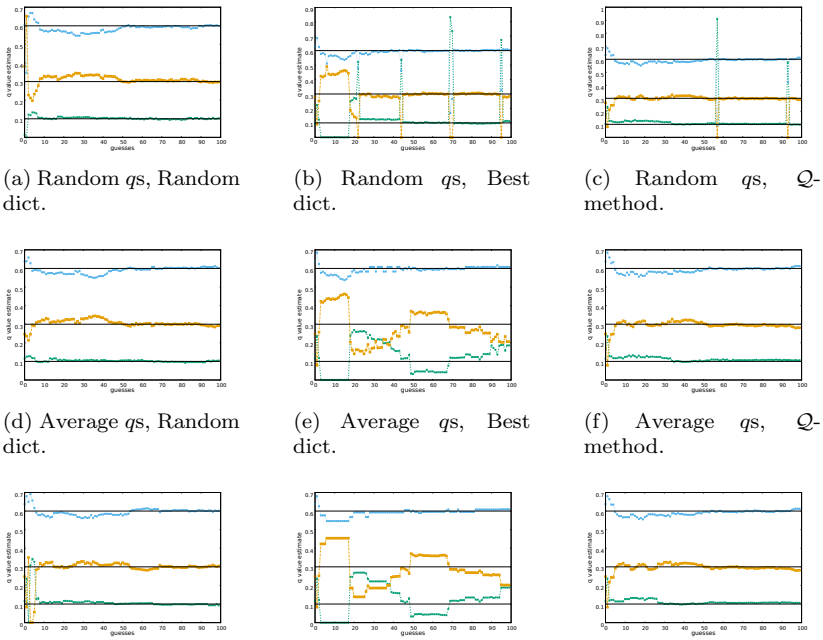
1.1 Variables for estimating the characteristics of the password set

When estimating the q-values assigned to each dictionary, the variables in the multi-armed bandit play an important role.

Let us first look at the nine graphs produced by combinations of the guess choice and initialisation variables for password set 1. The results are shown in Figure 1.

Guess choice: Best dictionary We can see that the best dictionary method for choosing guesses is not effective for determining the characteristics of a password set. This is because of the emphasis on exploitation of the best dictionary over exploring all the dictionaries. Where there is large overlap between the passwords in the dictionaries, the best method will provide information about the distribution of each dictionary. However, when this is not the case we do not learn about the relationship between the password set and all the dictionaries. A particularly poor set up is guessing from the best dictionary and initialising gradient descent with the previous best values. Figure 2 shows the outcome of this set-up for password set 4. Notice that despite incorrectly approximating the 0.55:0.3:0.1:0.05 split of the password set, it does not diverge from its initial approximation. Below we will explain why this can occur in the model.

Initialisation: Best qs When the number of guesses is less than the number of dictionaries the likelihood function can be degenerate. This means multiple combinations of q-values can maximise the likelihood function. This is not as much of an issue in the Average and Random initialisation methods but in the Best qs initialisation method we are seeding the next guess with the best q-values from our previous guess. But because we are constrained by the probability simplex defined by



(a) Random qs, Random dict.

(b) Random qs, Best dict.

(c) Random qs, $\mathcal{Q}$-method.

(d) Average qs, Random dict.

(e) Average qs, Best dict.

(f) Average qs, $\mathcal{Q}$-method.

(g) Best qs, Random dict.

(h) Best qs, Best dict.

(i) Best qs, $\mathcal{Q}$-method.

Figure 1: Password set 1 q-value estimates. Shown for each combination of initialisation and guess choice method.

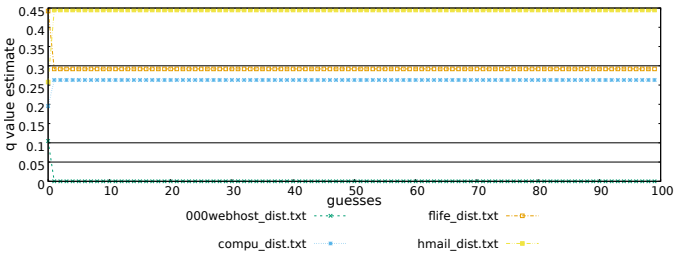


Figure 2: Password set 4 q-value estimates. Initialization: best $\hat{q}$-values, Guessing: best dictionary.

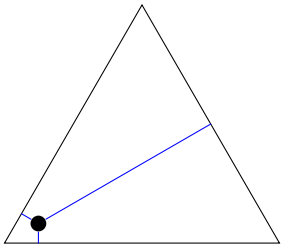


Figure 3: Diagram impression of a probability simplex for a 3-dictionary set. Depicts q-values starting in one corner of the simplex.

$$\sum_{i}^{n} q_i = 1 \qquad \text{and} \qquad q_i \geq 0, \tag{1}$$

movement to leave this initial approximation can be constrained. See Figure 3 depicting a 3-dictionary probability simplex which shows $\vec{q}$ in a corner of the simplex. q-values in a corner position are limited in the valid directions they can move in. This creates the potential to become "stuck". It is further emphasised when the Best dictionary method is used for guess choice as it is compounded now by limiting the information gathered about any password set other than the one estimated as the best. A simple solution is to reset the q-values before each descent as is done in the Average values method. A more complex solution could involve avoiding seeding guesses until a non-degenerate likelihood function can be derived.

Initialisation: Random qs Notice the spikes in two of the graphs which show initialisation with random q-values.[1] These spikes represent a failure to converge when the randomly chosen initial $\hat{q}$ values are far from the true values. In our other simulated password sets these spikes were also present, in the graphs which used random qs for initialisation and a random dictionary for guessing. For this reason we will avoid using the random initialisation method when determining the make-up of a password set.

1.1.1 Conclusion

Based on this analysis, we find that the results from the models initialised using both Random q-values and previous Best q-values are not reliable. In addition when guesses are chosen from the Best dictionary only, the model does not fare well at approximating the q-values. The Average q initialisation method paired with either the Random dictionary guess choice or the $\mathcal{Q}$-method guess choice performs consistently well at approximating characteristics. One advantage of the $\mathcal{Q}$-method over the random dictionary choice is that it is deterministic.

[1] Because the random initialisation is non-deterministic, these spikes can occur to a greater or lesser extent in different iterations. To demonstrate the existence of the spikes, we have intentionally chosen graphs from an iteration in which the spikes are prominent.

1.2 Variables for Gradient descent step size

As well as different options for initialisation and guess choice, we also have a number of choices for the gradient descent step sizes. In our paper, we introduced three popular options for the step size variable within gradient descent literature [1]. Each method had advantages and disadvantages and it is important to choose a method that works for a given set-up of the multi-armed bandit.

We implemented these three options in four ways:

- Constant alpha
- Constant step size
- Starting with constant alpha and then adapting step size
- Starting with constant step size and then adapting.

Recall the adaptation involves increasing the step size if it will result in an increased likelihood value and similarly decreasing the step size incrementally until the resulting likelihood is an improvement on the last likelihood. Thus we can guarantee that each step results in an increase to the likelihood function.

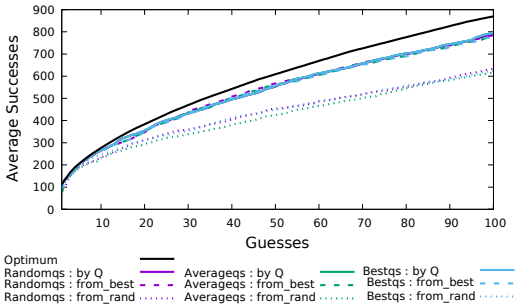
1.2.1 Successes

Figure 4 shows the success plots for password set 1 using the 4 different step size methods. There does not appear to be a significant difference in the success rates for the different options. Respectively they finish with 775, 795, 769, 794 compromised users after 100 guesses. In this example, the constant method fares marginally better, but this is not consistent across the other three password sets.

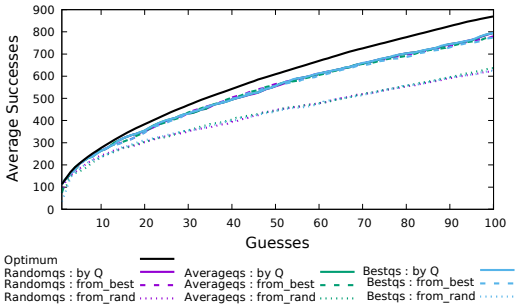
1.2.2 Estimating the q-values

Our goal is to achieve an accurate prediction of the q-values for a selection of dictionaries to help us determine the characteristics of the password set. We therefore compare how effectively the step size options estimate the q-values. We can do this in two ways. By looking directly at the q-values and visually comparing them against the actual values. Alternatively by plotting the distance the q-values are from the optimal and analysing this graph. Below we do both and see that both show us that the constant alpha approach seems to work best with our model.

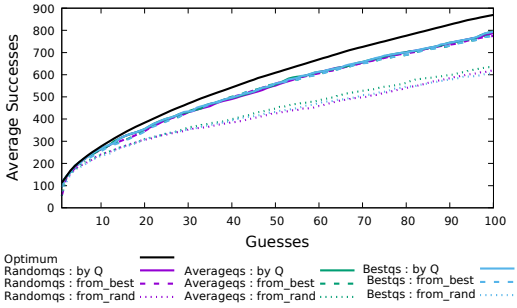
For a single initialisation and guess choice In the previous section we determined that the Average qs Random dictionary setup was a reliable set-up for estimating the q values. Therefore, in Figure 5 we show the Average qs Random dictionary set-up of the model for estimating password set 1 for the four step size options. We can see that constant alpha seems to give the best approximation of the true q-values for this set-up of the initialisation and guess choice. Both adaptive options give a poor estimation and the constant step size option performs only slightly worse than constant alpha.



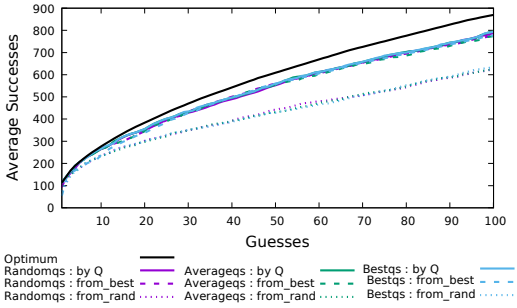
(a) Constant alpha.



(b) Constant step size.

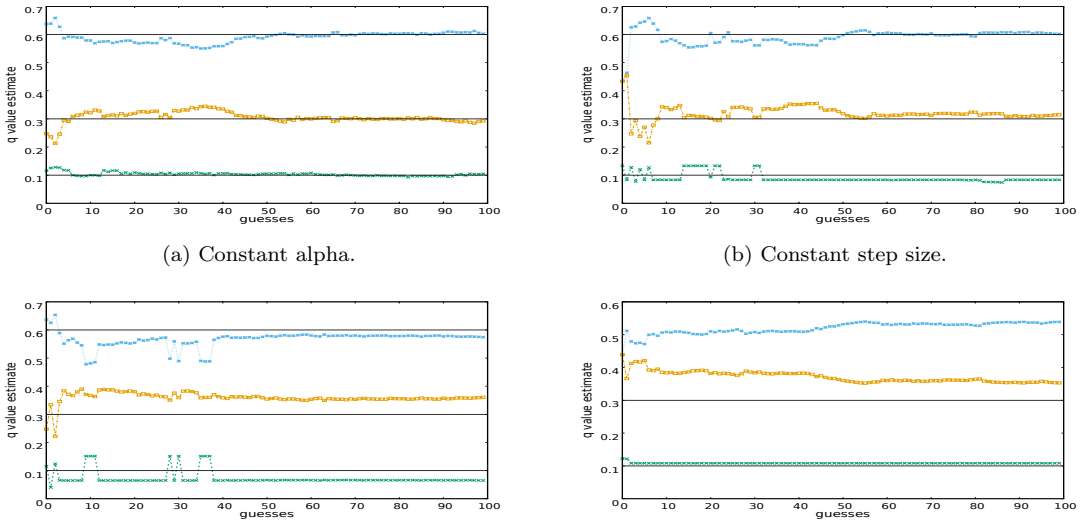


(c) Adapting constant alpha.



(d) Adapting constant step size.

Figure 4: Password set 1 guessing successes. Shown for each combination of gradient descent step size methods.



(a) Constant alpha.

(b) Constant step size.

(c) Adapting constant alpha.

(d) Adapting constant step size.

Figure 5: Password set 1 q-value estimates. Shown for each combination of gradient descent step size methods.

L_1 norms An L_1 norm is the sum of the magnitudes of vectors in a space. It is the sum of the absolute difference of the components of the vectors.

$$||x||_1 = \sum_{i=1}^{n} |x|.$$

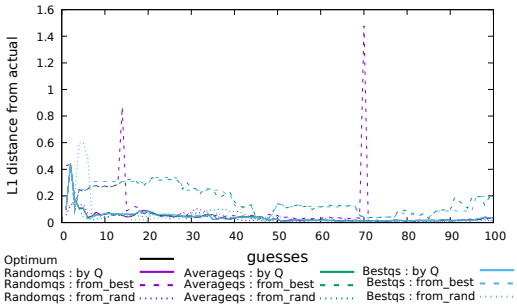
In this norm, all the components of the vector are weighted equally. We can use the L_1 norm to measure the distance between the actual and estimated q values at each guessing point. The result is the sum of the differences between the actual and estimated q for each dictionary.

Figure 6 compares the L_1 norms for the four step methods. In Figure 6a we see the L_1 norms for the constant alpha set up. All of the $\mathcal{Q}$-method approximations have L_1 norms consistently close to zero after an initial peak within the first 3 guesses. We see spikes in the Random qs Best dictionary plot and the other two Best dictionary options perform poorly. Random dictionary choice performs well with an L_1 norm generally less than 0.1.

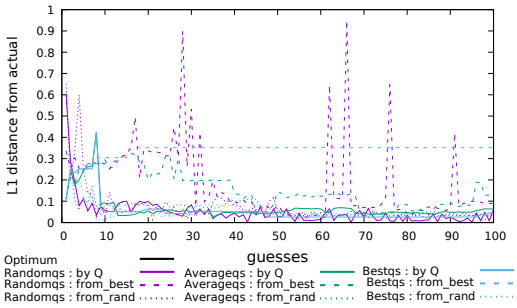
Figure 6b shows the L_1 norms for the constant step size method. We see regular spikes in the Random qs Best dictionary method. There is more variation in the $\mathcal{Q}$-method results. The L_1 norm for the Best qs Best dictionary method never converges to zero.

The general impression from the two adaptive plots (Figures 6c and 6d) is that they are inconsistent in their estimation. We see spikes in all the Random q initialisation methods. The constant step size adapting method reports reasonably small L_1 values for Average q initialisation with $\mathcal{Q}$-method guess choice and for Best q and Random dictionary choice. The constant alpha adapting method only shows low L_1 values for Average q initialisation and $\mathcal{Q}$-method guess choice.

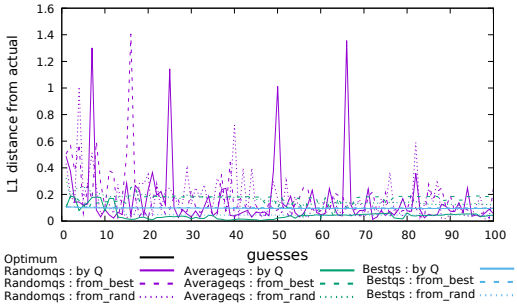
From this analysis we can conclude that the best estimates for the q values are provided using the constant alpha method for determining step size. This method involved a set alpha included in



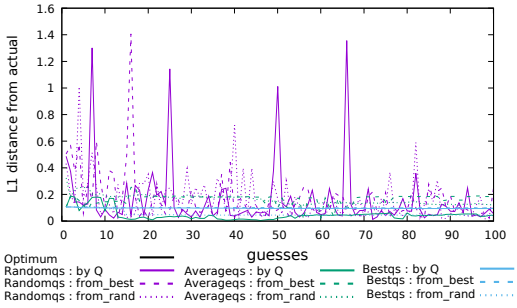
(a) Constant alpha.



(b) Constant step size.



(c) Adapting constant alpha.



(d) Adapting constant step size.

Figure 6: Password set 1 L_1 norms. Shown for each combination of gradient descent step size methods.

the iteration which results in a reduced step size as we approach the maximum. For our model we used the value $\alpha = 0.1$ though other values were also tested including 0.5 and 0.01.

In this section we have covered in detail the variables that are used in our multi-armed bandit model. It is important for us to test the combination of variables on the data we know the actual composition of. This allows us to compare against true values. Going forward we will test our bandit model on password sets for which we hold no information about. It is important that we have tested our model and tailored it in a way that verifies the results of these exotic password sets.

References

[1] M. Toussaint, "Lecture notes: Some notes on gradient descent," 2012.