Machine Learning, Advanced Course (DD2434)
Assignment 2

Martin Hwasser, hwasser@kth.se

December 29, 2016

Task 2.1

Question 1. In which pairs is one value larger than the other?

Pair 1: $p(t^1|d^1)$ and $p(t^1)$
Pair 2: $p(d^1|t^0)$ and $p(d^1)$
Pair 3: $p(h^1|e^1, f^1)$ and $p(h^1|e^1)$
Pair 5: $p(c^1|h^0)$ and $p(c^1)$

Question 2. Which pairs are equal?

Pair 4: $p(c^1|f^0) = p(c^1)$
Pair 6: $p(d^1|h^1, e^0) = p(d^1|h^1)$
Pair 7: $p(c^1|h^0, f^0) = p(c^1|h^0)$

Question 3. Which pairs are incomparable (i.e., the two values can not be compared based on the information available in the DAG.)

Pair 8: $p(d^1|e^1, f^0, w^1)$ and $p(d^1|e^1, f^0)$
Pair 9: $p(t^1|w^1, f^0)$ and $p(t^1|w^0)$

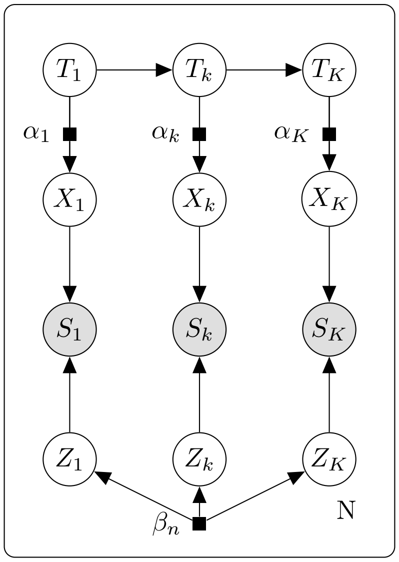


Figure 1: Graphical model of the casino.

Task 2.2

Question 4. Provide a drawing of the Casino model as a graphical model. It should have a variable indicating the table visited in the k:th step, variables for all the dice outcomes, variables for the sums, and plate notation should be used to clarify that N players are involved.

See figure 1.

T_k represents the table in the k:th step, S_k is the sum of the outcomes of the table's dice X_k and the player's dice Z_k at table k. β_n is represents each player's dice distribution, while α_k represents each table's dice distribution.

Question 5. Implement the Casino model (in Matlab or Python).

The Casino model was implemented using Python in about 80 lines of code.

Question 6. Provide data generated using at least three different sets of categorical dice distributions what does it look like for all unbiased dice, i.e., uniform distributions, for example, or if some are biased in the same way, or if some are unbiased and there are two different groups of biased dice.

Figure 2a shows the distribution of outcomes when the dices of both the casino and the players have a uniform distribution.

Figure 2b shows the distribution of outcomes when all players have fair dices, the unprimed tables have fair dices, but the primed tables have a dice distribution like $p(d = 1) = \frac{5}{10}$ and $p(2 \leq d \leq 6) = \frac{1}{10}$.

Figure 2c shows the distribution of outcomes when all tables at the casino have fair dices, but certain players have a biased distribution $p(1 \leq d \leq 5) = \frac{1}{10}$ and $p(d = 6) = \frac{5}{10}$.

Figure 2d shows the distribution of outcomes when the casino have biased dices like in figure 2b, and the players also have a biased distribution $p(1 \leq d \leq 5) = \frac{1}{7}$ and $p(d = 6) = \frac{2}{7}$.

Task 2.3

Question 7. Implement the VI algorithm for the variational distribution in Equation (10.24) in Bishop.

The VI algorithm was implemented in Python and is around 50 lines of code (excluding plotting functions).

Question 8. Describe the exact posterior.

From the paper *Conjugate Bayesian analysis of the Gaussian distribution* by Murphy, the exact posterior is given by:

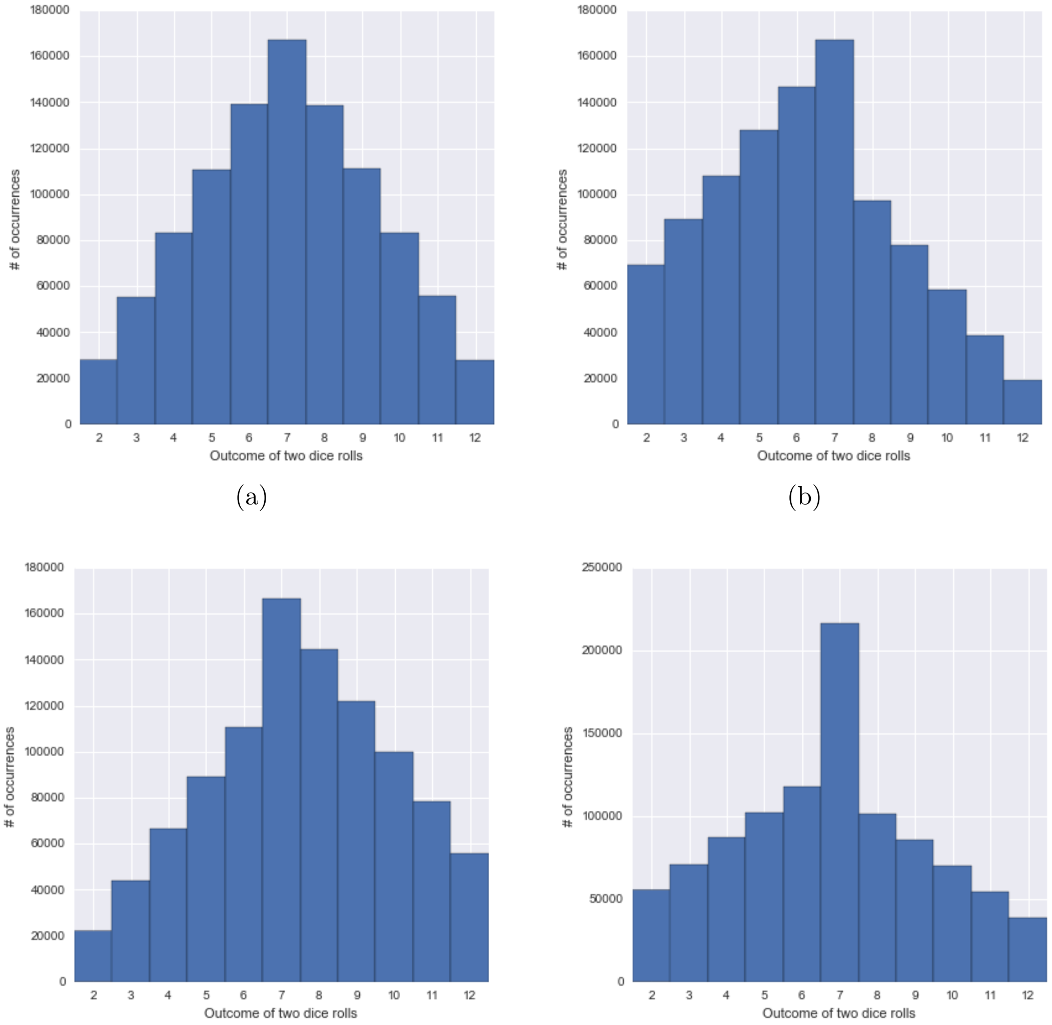
$$p(\mu, \lambda | D) = \mathcal{NG}(\mu, \lambda | \mu_n, \kappa_n, \alpha_n, \beta_n) \tag{1}$$

where

$$\begin{aligned}
\mu_n &= \frac{\kappa_0 \mu_0 + n\bar{x}}{k_0 + n} \\
\kappa_n &= \kappa_0 + n \\
\alpha_n &= \alpha_0 + n/2 \\
\beta_n &= \beta_0 + \frac{1}{2}\sum_{i=1}^{n}(x_i - \bar{x})^2 + \frac{\kappa_0 n (\bar{x} - \mu_0)^2}{2(\kappa_0 + n)}
\end{aligned} \tag{2}$$

Question 9. Compare the variational distribution with the exact posterior. Run the inference for a couple of interesting cases and describe the difference.

To illustrate the data we used a normal distribution with zero mean and unit variance. Countours of the true posterior are shown in green, while the contours in blue are the approximations, which are colored red after convergence.



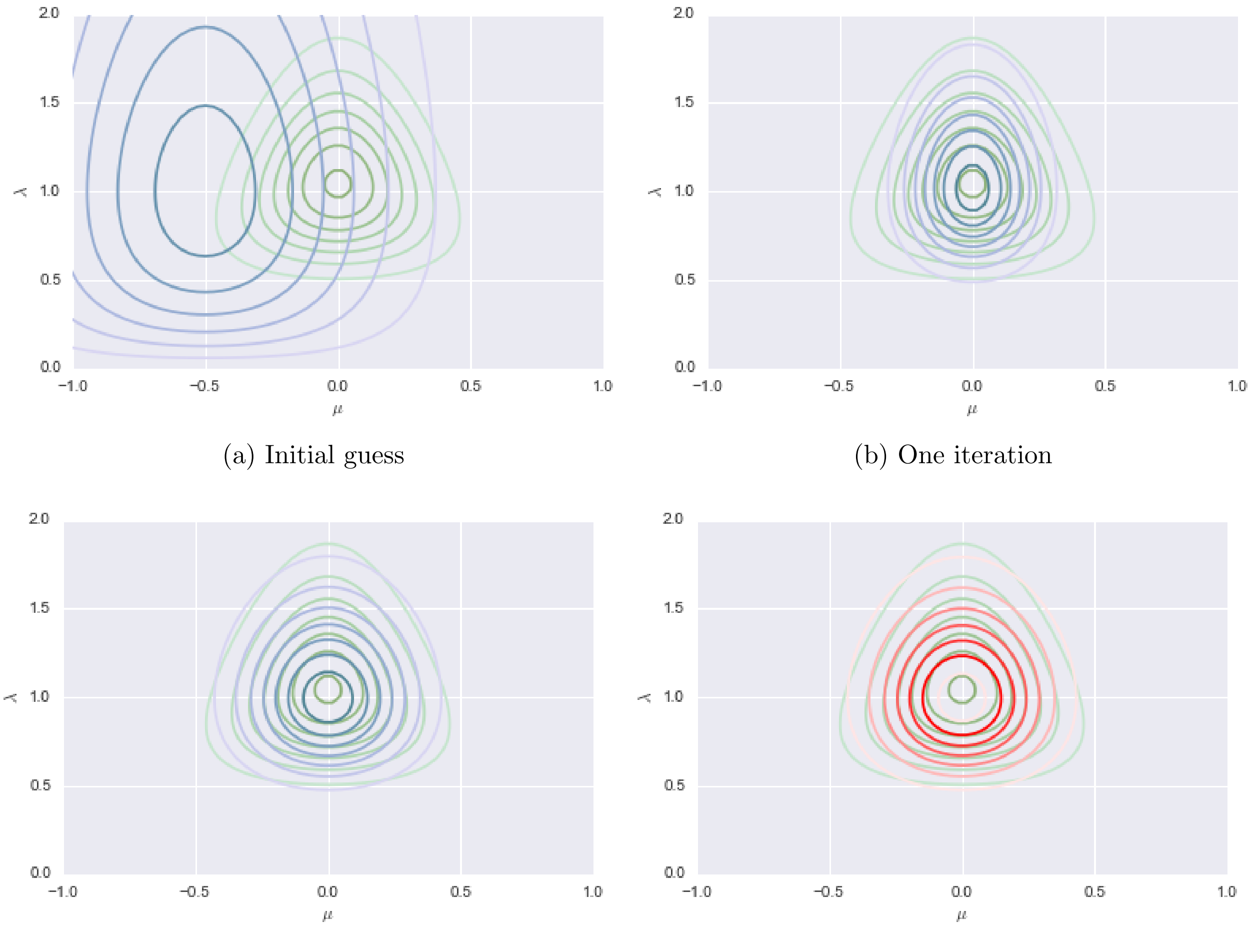
(a)

(b)

(c)

(d)

Figure 2: Both casino and player have fair dices in 2a. In 2b, primed tables have unfair dices and all players have fair dices. All tables have fair dices and certain players have unfair dices in 2b. And finally, 2d shows a mix where primed tables and certain players have unfair dices.



(a) Initial guess

(b) One iteration

(c) Two iterations

(d) Converged

Figure 3: Variational inference of 20 samples.

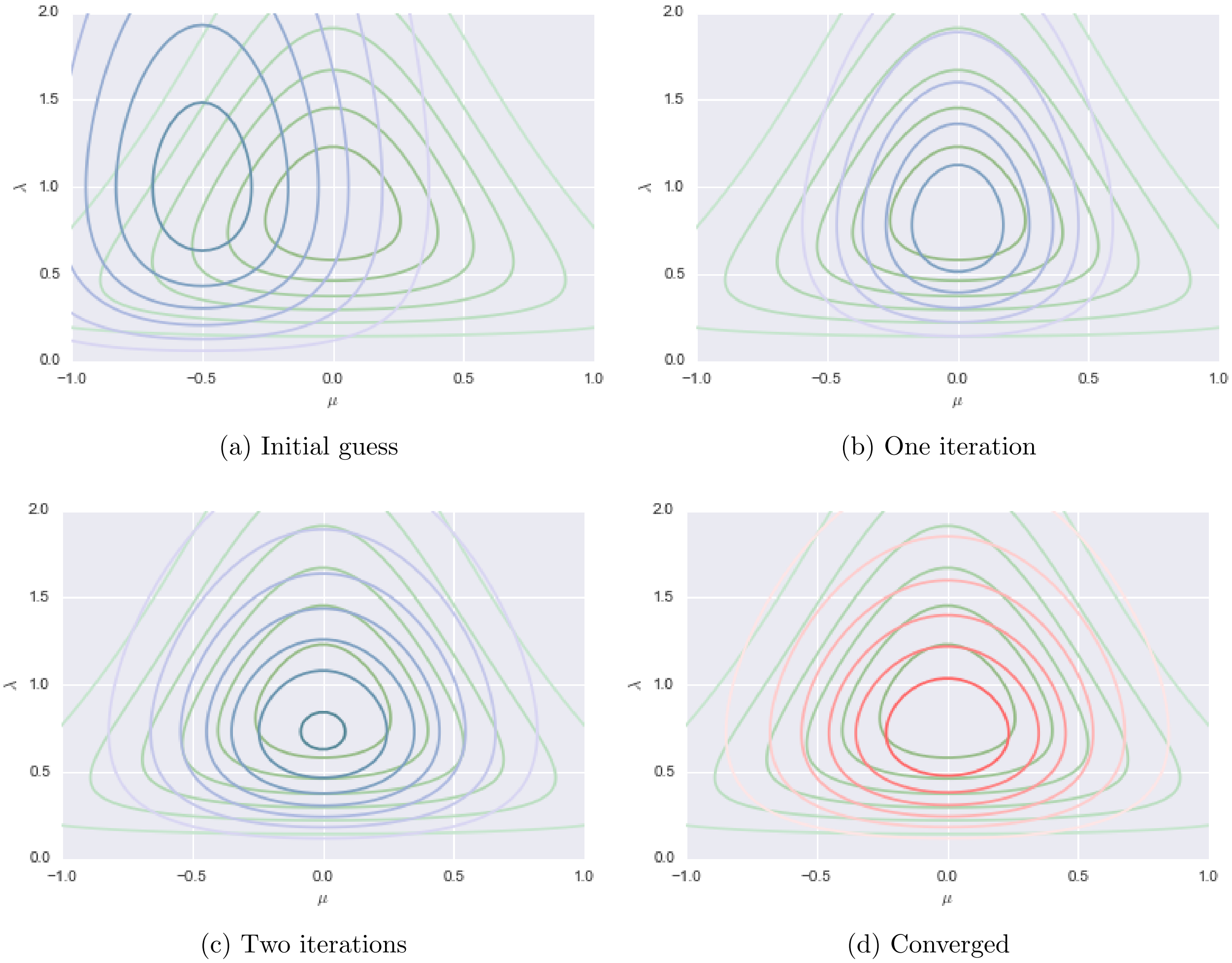
Interestingly, the approximation converged quickly despite various extreme priors and even with only a few datapoints. With more data, the distribution becomes more compact and our approximation also fits the data better. This makes sense as, in general, it's common for factorized variational approximation to be more compact (Bishop).

In figure 3 we see the true posterior compared with an approximation from 20 samples of a univariate Gaussian distribution. In figures 3b and 3c we see how the approximation converges to the true posterior and the converged approximation in 3d.

Using fewer (5) samples, the approximation still converges as seen in figure 4. The shape will be less compact and will have larger variance around the mean.

By increasing the prior rate of the distribution, the shape becomes flat as seen in figure 5 and takes a bit longer to converge, which makes sense as we are being more informative in our guess.

Task 2.4



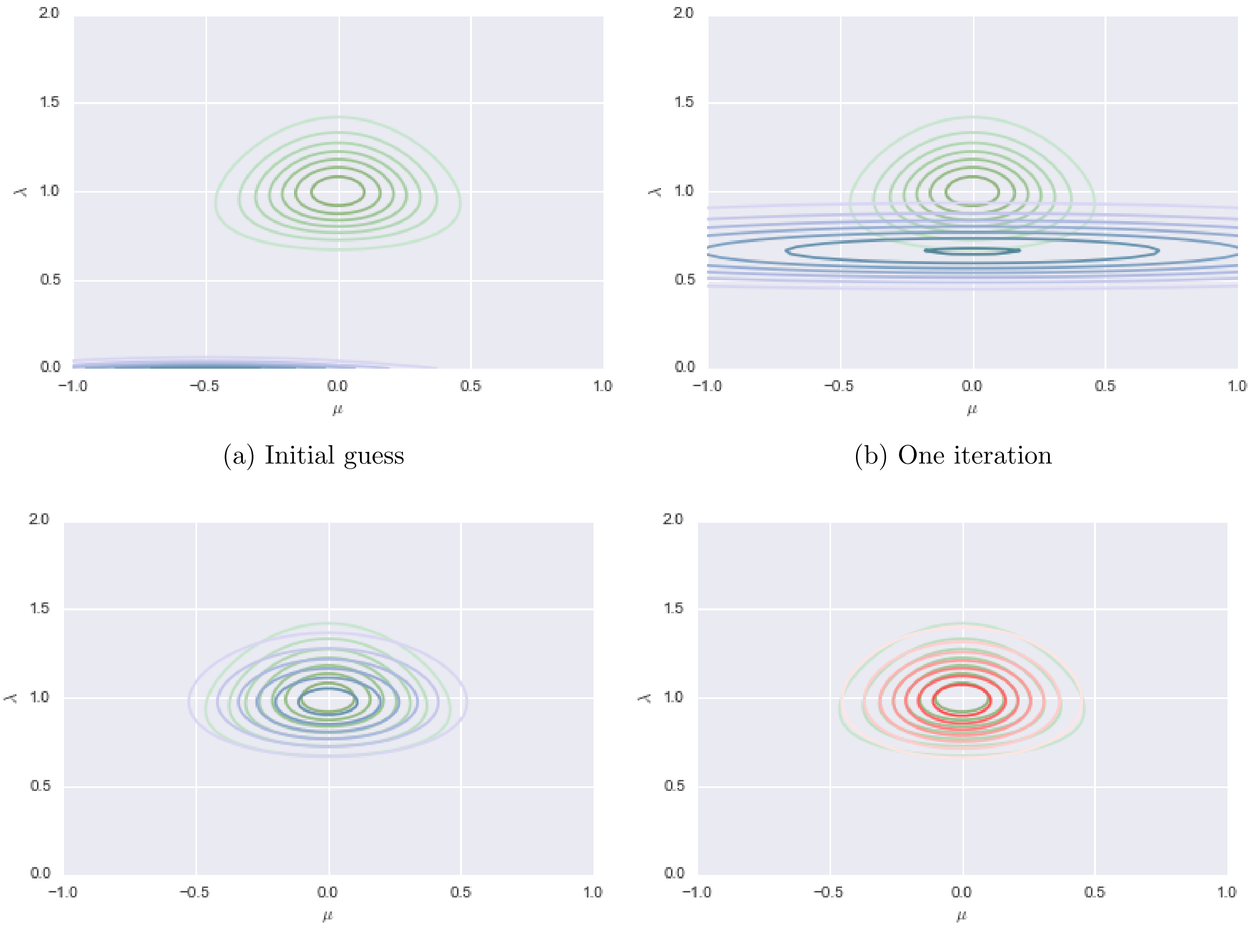
(a) Initial guess

(b) One iteration

(c) Two iterations

(d) Converged

Figure 4: Variational inference of only 5 samples.



(a) Initial guess

(b) One iteration

(c) Two iterations

(d) Converged

Figure 5: In this experiment the prior rate, b, was set to be high which converged somewhat slower.

Question 10. Describe an algorithm that, given (1) the parameters Θ of the full casino model of Task 2.2 (so, Θ is all the categorical distributions corresponding to all the dice), (2) a sequence of tables $r_1, ..., r_K$ (that is, r_i is t_i or t_i') , and (3) an observation of dice sums $s_1, ..., s_K$, outputs $p(r_1, ..., r_K|s_1, ..., s_K, \Theta)$.

$$\begin{aligned} p(r_1, ..., r_K|s_1, ..., s_K, \Theta) &= \big\{\text{Bayes' theorem}\big\} \\ &= \frac{p(s_1, ..., s_K|r_1, ..., r_K, \Theta)p(r_1, ..., r_K, \Theta)}{p(s_1, ..., s_K, \Theta)} \end{aligned} \tag{3}$$

Since Θ is fixed throughout, we can omit it.

The sum of the two dice rolls depend only on the table:

$$\begin{aligned} p(s_1, ..., s_K|r_1, ..., r_K) &= p(s_1, ..., s_{K-1}|s_K, r_1, ..., r_K)p(s_K|r_1, ..., r_K) \\ &= p(s_1, ..., s_{K-1}|r_1, ..., r_K)p(s_K|r_K) \\ &= p(s_1, ..., s_{K-2}|s_{K-1}, r_1, ..., r_K)p(s_{K-1}|r_1, ..., r_K)p(s_K|r_K) \\ &= p(s_1, ..., s_{K-2}|r_1, ..., r_K)p(s_{K-1}|r_{K-1})p(s_K|r_K) \\ &= \prod_{k=1}^{K} p(s_k|r_k) \end{aligned} \tag{4}$$

We can view the sum as the emission governed by the table and represent it by an emission matrix B (using popular HMM notation):

$$\prod_{k=1}^{K} B_{r_k,s_k} = \prod_{k=1}^{K} \sum_{x=1}^{6} \sum_{z=1}^{6} I(s_k = x + z)(X_k = x)(Z_k = z) \tag{5}$$

Knowing that the table's dice distribution only depends on the previous table's dice distribution, and as such has the first order Markov property, we can express this by:

$$\begin{aligned} p(r_1, ..., r_K) &= p(r_2, r_3, ..., r_K|r_1)p(r_1) \\ &= p(r_3, ..., r_K|r_2)p(r_2|r_1)p(r_1) \\ &= p(r_1) \prod_{k=2}^{K} p(r_k|r_{k-1}) \end{aligned} \tag{6}$$

We can take the first initial term of probabilities and call it π:

$$\pi_{r_1} = p(r_1) \tag{7}$$

And subsequent probabilities is our transition matrix A, such that:

$$\prod_{k=2}^{K} A_{r_k,r_k-1} = \prod_{k=2}^{K} p(r_k|r_{k-1}) \tag{8}$$

We can then write the prior using eq.7 and eq.8:

$$p(r_1,...,r_K) = \pi_{r_1}\prod_{k=2}^{K} A_{r_k,r_{k-1}} \tag{9}$$

Finally, the denominator:

$$\begin{aligned} p(s_1,...,s_K) &= \sum_i p(s_1,...,s_K,R_k=i) \\ &= \sum_i \alpha_K(i) \end{aligned} \tag{10}$$

where $\alpha_k(i)$ is the forward variable from the forward-backward algorithm.

Using eq. 5, 9, 10 we can compute the probability of a primed or non-primed table given the sums:

$$p(r_1,...,r_K|s_1,...,s_K) = \frac{\pi_{r_1}\prod_{k=2}^{K} A_{r_k,r_{k-1}}\prod_{k=1}^{K} B_{r_k,s_k}}{\sum_{i=1}\alpha_K(i)} \tag{11}$$

Question 11. You should also show how to sample $r_1,...,r_K$ from $p(R_1,...,R_K|s_1,...,s_K,\Theta)$ as well as implement and show test runs of this algorithm. In order to design this algorithm show first how to sample r_K from

$$p(R_K|s_1,...,s_K,\Theta) = p(R_K,s_1,...,s_K|\Theta)/p(s1,...,s_K|\Theta)$$

and then r_{K-1} from

$$p(R_{K-1}|r_K,s_1,...,s_K,\Theta) = p(R_{K-1},r_K,s_1,...,s_K|\Theta)/p(r_K,s_1,...,s_K|\Theta)$$

As in Question 10, Θ has been omitted for brevity. We want to sample r_k, given the model parameters Θ and some given sequence of dice sums S_K:

$$p(R_K|s_1,...,s_K) = p(R_K,s_1,...,s_K)/p(s_1,...,s_K|)$$

From equation 10 we have that:

$$p(R_K=r_K|s_1,...,s_K) = \frac{\alpha_K(r_K)}{\sum_i \alpha_K(i)} \tag{12}$$

For subsequent steps:

$$\begin{aligned} p(r_{K-1}|r_K,s_1,...,s_{K-1} &= \{\text{Bayes' theorem}\} \\ &= \frac{p(r_K|r_{K-1},s_1,...,s_{K-1})p(r_{K-1}|s_1,...,s_{K-1})}{p(r_K|s_1,...,.s_{K-1})} \\ &= \frac{A_{r_{k-1}r_k}\alpha_{K-1}(r_{K-1})}{\sum_i A_{ir_k}\alpha_{K-1}(i)} \end{aligned} \tag{13}$$

	Player a										Player b									
Sum	12	4	3	8	3	3	6	5	3	4	5	4	10	8	7	2	6	12	5	5
Table	0	0	1	0	1	0	1	1	0	1	0	1	1	1	1	1	1	0	1	1
Sample	0	0	0	0	0	1	1	1	0	1	0	1	0	1	1	1	1	0	0	0

	Player c										Player d									
Sum	4	7	10	9	3	10	7	5	5	7	9	6	8	10	3	4	10	5	9	6
Table	1	0	0	1	1	1	0	1	1	1	0	1	0	0	1	1	1	1	1	1
Sample	0	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Table 1: Sampling R given sums S. Results show whether the actual table was primed or not and our sample approximation for players a, b, c & d, each visiting 10 tables where primed tables have biased distributions.

This allows us to sample from the posterior given the sum S of the dice outcomes and the dice distributions Θ. To see that this makes sense, we let the players have uniform dice distributions, and we let some of the tables have a biased dice to prefer low outcomes $Cat(\frac{4}{10}, \frac{2}{10}, \frac{2}{10}, \frac{2}{10}, 0, 0)$. With the primed tables' dice, rolling a 6 is not possible, and this is reflected when we sample as seen in table 1 as we will always draw $r_k = 0$, ie non-primed table, given a sum of 12.

The sampling was implemented in Python and is about 80 lines of code.

Task 2.5

Question 12. Present the algorithm written down in a formal manner (using both text and mathematical notation, but not pseudo code).

With Expectation-Maximization, we want to maximize the expectation for a set of observations. In particular, we want to maximize the expectation of our dice sums $p(S|\Theta)$ given S, eg finding the dice distribution Θ.

We have:

$$p(\mathbf{S}, \mathbf{X}, \mathbf{Z}, \Theta) = p(\mathbf{S}|\mathbf{X}, \mathbf{Z}, \Theta)p(\mathbf{X}|\Theta)p(\mathbf{Z}|\Theta)$$

Let's think about what we are trying to achieve. We don't know Θ so we can't compute the complete log likelihood $p(S|X, Z)$ since the distribution of X and Z are latent variables. Thus we have only the posterior distribution $p(X, Z|S, \Theta)$.

Instead we'll compute the expected log likelihood (eq.15), which is the E step of the EM algorithm. And in the M step we try to maximize this expectation.

$$\begin{aligned} \mathcal{L}(X, Z, S, \Theta) &= p(S, X, Z|\Theta) \\ &= \prod_{n=1}^{N}\prod_{k=1}^{K} p(s_k^n, x_k^n, z_k^n|\Theta) \\ &= \prod_{n=1}^{N}\prod_{k=1}^{K} \Big(p(x_k^n|\Theta)p(z_k^n|\Theta)\Big)^{I(s_k^n = x_k^n + z_k^n)} \end{aligned} \tag{14}$$

Apply the logarithm:

$$\log \mathcal{L}(S; X; Z; \Theta) = \sum_{n=1}^{N}\sum_{k=1}^{K} I(s_k^n = x_k^n + z_k^n)\Big(\log p(x_k^n|\Theta) + \log p(z_k^n|\Theta)\Big) \tag{15}$$

We'll denote $r_{k,n,a,b} = p(x_k^n = a, z_k^n = b|s_k^n, \Theta)$ and proceed:

$$\begin{aligned} \log \mathcal{L}(S; X; Z; \Theta) &= \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{a=1}^{6}\sum_{b=1}^{6} p(x_k^n = a, z_k^n = b|s_k^n, \Theta) I(s_k^n = x_k^n + z_k^n)\Big(\log p(x_k^n|\Theta) + \log p(z_k^n|\Theta)\Big) \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{a=1}^{6}\sum_{b=1}^{6} r_{k,n,a,b} I(s_k^n = x_k^n + z_k^n)\Big(\log p(x_k^n|\Theta) + \log p(z_k^n|\Theta)\Big) \end{aligned} \tag{16}$$

The algorithm follows:

1. Choose a setting for Θ^{old}.

2. **(E step)** Here we want to evaluate $r_{k,n,a,b}$:

$$\begin{aligned} r_{n,k,a,b} &= p(x_k^n = a, z_k^n = b|s_k^n, \Theta^{old}) \\ &= \frac{p(x_k^n = a|\Theta)p(z_k^n = b|\Theta)I(a + b = s_k^n)}{\sum_{i=1}^{6}\sum_{j=1}^{6} p(x_k^n = i|\Theta)p(z_k^n = j|\Theta)I(i + j = s_k^n)} \end{aligned} \tag{17}$$

3. **(M step)** Let $\Theta^{new} = argmax_\Theta \mathcal{Q}(\Theta, \Theta^{old})$ where $\mathcal{Q}$ is the expected log likelihood:

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{a=1}^{6}\sum_{b=1}^{6} r_{k,n,a,b} I(x_k^n + z_k^n = s_k^n)\Big(\log p(x_k^n|\Theta) + \log p(z_k^n|\Theta)\Big) \tag{18}$$

where

	Table distribution (X)		**Player distribution (Z)**	
	True	Approximated avg	True	Approximated avg
$p(x=1)$	$\frac{4}{10}$	0.3989917	$\frac{1}{6}$	0.17383351
$p(x=2)$	$\frac{2}{10}$	0.11961509	$\frac{1}{6}$	0.18215501
$p(x=3)$	$\frac{2}{10}$	0.17389909	$\frac{1}{6}$	0.22589532
$p(x=4)$	$\frac{2}{10}$	0.20343964	$\frac{1}{6}$	0.21460231
$p(x=5)$	$\frac{0}{10}$	0.09452868	$\frac{1}{6}$	0.12179775
$p(x=6)$	$\frac{0}{10}$	0.0095258	$\frac{1}{6}$	0.0817161

Table 2: True and averaged approximated dice distributions using the EM algorithm for 10 tables and 100 players.

$$p(x_k = a|\Theta) = \frac{\sum_{n=1}^{N} \sum_{b=1}^{6} r_{k,n,a,b}}{\sum_{n=1}^{N} \sum_{a=1}^{6} \sum_{j=b}^{6} r_{k,n,a,b}} \tag{19}$$

$$p(z^n = b|\Theta) = \frac{\sum_{k=1}^{K} \sum_{a=1}^{6} r_{k,n,a,b}}{\sum_{k=1}^{K} \sum_{a=1}^{6} \sum_{b=1}^{6} r_{k,n,a,b}} \tag{20}$$

4. Let $\Theta^{old} = \Theta^{new}$ if the algorithm has not converged according to some criterion, and go back to step 2.

Question 13. Implement it and test the implementation with data generated in Task 2.2, and provide graphs or tables of the results of testing it with the data.

The EM algorithm was implemented in Python and is around 60 lines of code. The algorithm worked very well given a small sized sample such as $K = 10$ and $N = 100$. For simplicity and ease of interpretation, all the tables in the casino had the same biased dice distribution as seen in table 2 and all the players' dice distribution were uniform. These distributions are also illustrated by figures 6 and 7.

A convergence graph can be seen in figure 8.

Task 2.6

Question 14. Present the algorithm written down in a formal manner (using both text and mathematical notation, but not pseudo code).

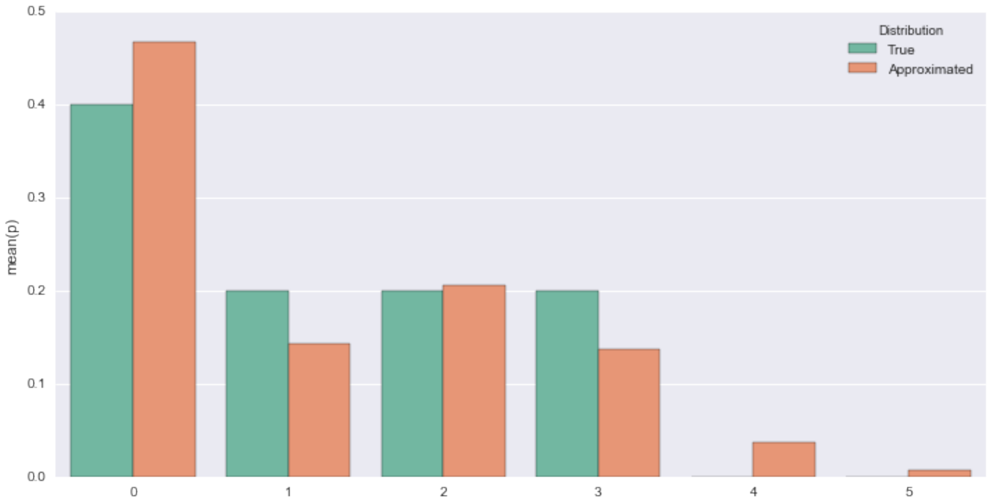


Figure 6: True vs approximated distributions for the tables' distributions.

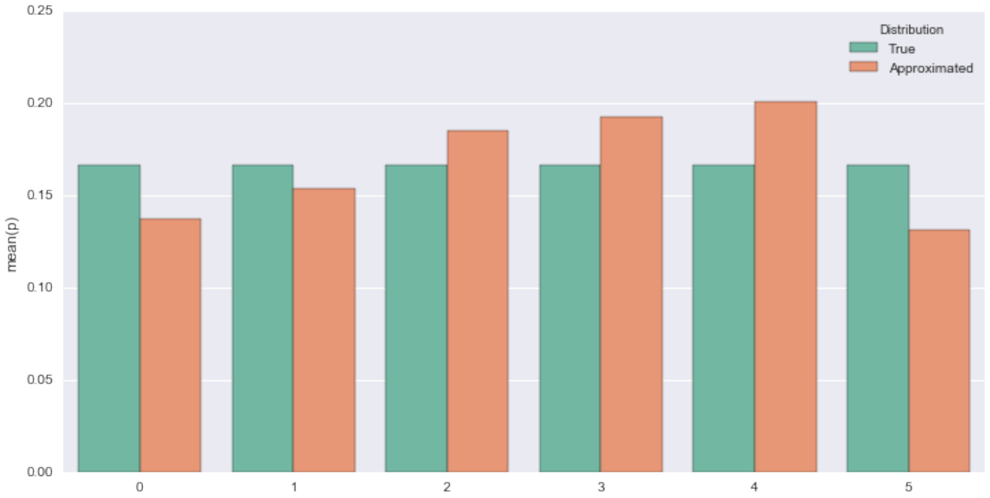


Figure 7: True vs approximated distributions for the players' distributions.

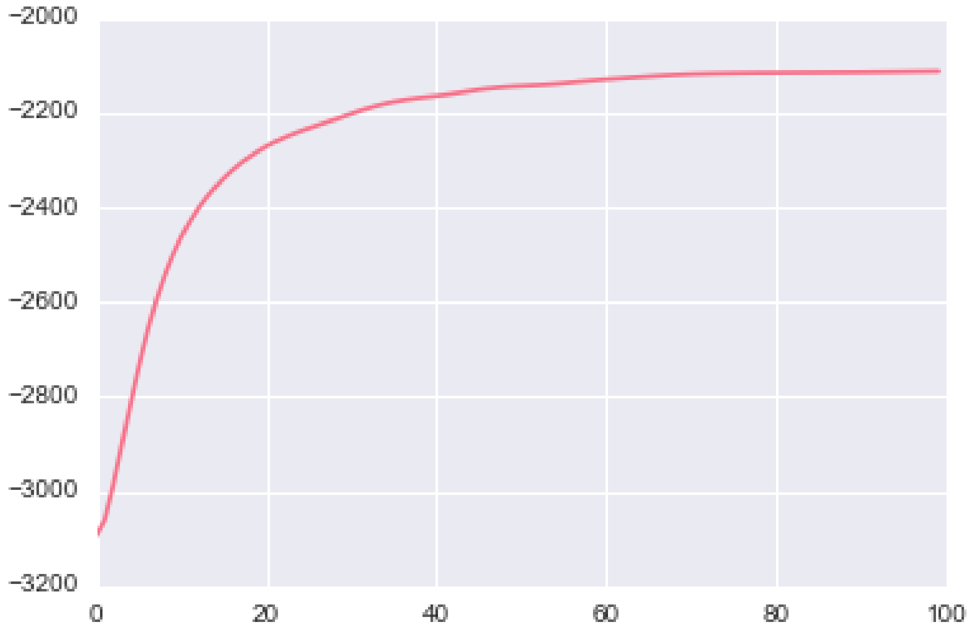


Figure 8: Convergence of the EM algorithm.

We know that the sum of two gaussian distributions is also gaussian, eg:

$$X \sim \mathcal{N}(\mu, \lambda^{-1})$$
$$Z \sim \mathcal{N}(\xi, \iota^{-1})$$
$$X + Z = S \sim \mathcal{N}(\mu + \xi, \lambda^{-1} + \iota^{-1})$$

Since the players and tables are independent of each other, we can describe the joint data as:

$$\begin{aligned} p(D, \mu_1, ..., \mu_k, \xi_1, ..., \xi_n) &= p(D|\mu_1, ..., \mu_k, \xi_1, ..., \xi_n)p(\mu_1, ..., \mu_k)p(\xi_1, ..., \xi_n) \\ &= \prod_{k=1}^{K}\prod_{n=1}^{N} p(S_k^n|\mu_k + \xi_n, \lambda_k^{-1} + \iota_n^{-1}) \prod_{k=1}^{K} p(\mu_k|\mu, \lambda^{-1}) \prod_{n=1}^{N} p(\xi_n|\xi, \iota^{-1}) \end{aligned} \tag{21}$$

Using Bishop 10.9, a general expression for the optimal solution for $q_j^*(S, X, Z|\Theta)$ is given by:

$$\ln q_k^*(\mu_k) = \mathbb{E}_{\xi_n}\Big[\ln p(D, \mu_{1:K}, \xi_{1:N})\Big] + c$$

So we take the log of eq. 21 to estimate the form of $q(\mu_1)$, and only keeping the terms that depend on μ_1 and completing the squares:

$$\begin{aligned}
\ln q_1^*(\mu_1) &= \mathbb{E}_{\xi_n} \ln p(D, \mu_{1:K}, \xi_{1:N}) \\
&= -\frac{1}{2}\mathbb{E}_{\xi_n}\Big\{ \sum_{k=1}^{K}\sum_{n=1}^{N}\Big\{\frac{\big(S_k^n - (\mu_k + \xi_n)\big)^2}{\lambda_k^{-1} + \iota_n^{-1}}\Big\} + \frac{1}{\lambda^{-1}}\sum_{k=1}^{K}(\mu_k - \mu)^2 + \frac{1}{\iota^{-1}}\sum_{n=1}^{N}(\xi_n - \xi)^2\Big\} + c \\
&= -\frac{1}{2}\mathbb{E}_{\xi_n}\Big\{ \sum_{n=1}^{N}\frac{\cancel{S_1^{(n)2}} - 2S_1^{(n)}(\mu_1 + \cancel{\xi_n}) + (\mu_1^2 + 2\mu_1\xi_n + \cancel{\xi_n^2})}{(\lambda_1^{-1} + \iota_n^{-1})} + \frac{(\mu_1^2 - 2\mu_1\mu + \cancel{\mu^2})}{\lambda^{-1}} + \cancel{\frac{(\xi_n - \xi)^2}{\iota^{-1}}}\Big\} + c \\
&= \mathbb{E}_{\xi_n}\Big\{ -\frac{\mu_1^2}{2}\sum_{n=1}^{N}(\frac{1}{\lambda_1^{-1} + \iota_n^{-1}} + \lambda) - \mu_1\big(\sum_{n=1}^{N}\frac{1}{\lambda_1^{-1} + \iota_n^{-1}}(\xi_n - S_1^{(n)}) - \mu\lambda\big)\Big\} + c \\
&= \mathbb{E}_{\xi_n}\Big\{ -\frac{\mu_1^2}{2}\sum_{n=1}^{N}(\frac{\lambda_1\iota_n}{\lambda_1 + \iota_n} + \lambda) - \mu_1\big(\sum_{n=1}^{N}\frac{\lambda_1\iota_n}{\lambda_1 + \iota_n}(\xi_n - S_1^{(n)}) - \mu\lambda\big)\Big\} + c \\
&= -\frac{\mu_1^2}{2}\sum_{n=1}^{N}(\frac{\lambda_1\iota_n}{\lambda_1 + \iota_n} + \lambda) + \mu_1\mu\lambda - \mu_1\mathbb{E}_{\xi_n}\Big\{\sum_{n=1}^{N}\frac{\lambda_1\iota_n}{\lambda_1 + \iota_n}(\xi_n - S_1^{(n)})\Big\} + c
\end{aligned} \tag{22}$$

Looking at eq (22), we see that this expression is a quadratic function of μ_1, and we can identify it as a Gaussian distribution $q^*(\mu_1) \sim \mathcal{N}(\alpha_{\mu_1}, \beta_{\mu_1})$ where:

$$\beta_{\mu_1} = \Big(\sum_{n=1}^{N}\frac{\lambda_1\iota_n}{\lambda_1 + \iota_n} + \lambda\Big) \tag{23}$$

$$\alpha_{\mu_1} = \frac{\mu\lambda - \mathbb{E}_{\xi_{1:N}}\Big\{\sum_{n=1}^{N}\frac{\lambda_1\iota_n}{\lambda_1+\iota_n}(\xi_n - S_1^n)\Big\}}{\beta_{\mu_1}} \tag{24}$$

This can be generalized such that $q^*(\mu_i) \sim \mathcal{N}(\alpha_i, \beta_i)$, where:

$$\beta_{\mu_i} = \Big(\sum_{n=1}^{N}\frac{\lambda_i\iota_n}{\lambda_i + \iota_n} + \lambda\Big) \tag{25}$$

$$\alpha_{\mu_i} = \frac{\mu\lambda - \mathbb{E}_{\xi_{1:N}}\Big\{\sum_{n=1}^{N}\frac{\lambda_i\iota_n}{\lambda_i+\iota_n}(\xi_n - S_i^n)\Big\}}{\beta_{\mu_i}} \tag{26}$$

Similarly, due to symmetry, we have that $q^*(\xi_i) \sim \mathcal{N}(\alpha_{\xi_i}, \beta_{\xi_i})$, where:

$$\beta_{\xi_i} = \Big(\sum_{k=1}^{K}\frac{\lambda_k\iota_i}{\lambda_k + \iota_i} + \lambda\Big) \tag{27}$$

$$\alpha_{\xi_i} = \frac{\iota\xi - \mathbb{E}_{\mu_{1:K}}\Big\{\sum_{k=1}^{K}\frac{\lambda_k\iota_i}{\lambda_k+\iota_i}(\mu_k - S_k^i)\Big\}}{\beta_{\xi_i}} \tag{28}$$

And so we can conclude that in order to infer the mean of one distribution, we need the expected value of the other distribution and vice versa.