COL863: Special Topics in Theoretical Computer Science
Rapid Mixing in Markov Chains
II semester, 2016-17
Minor III

Total Marks 100

Due: On moodle at 11:55PM, 10th April 2017

Problem 3.1 (30 marks) *Ex 11.2 of LPW 2e. What upper and lower bounds does Matthews method give for the cycle $\mathbb{Z}_n$? Compare to the actual value computed in Example 11.1 and explain why Matthews method gives a poor result for this class of chains.*

From example 11.1, we know that

$$t_{cov} = \frac{n(n-1)}{2}$$

From theorem 11.2, we know that

$$t_{cov} \leq t_{hit}(\sum_{i=1}^{n} \frac{1}{i})$$

We recall that the hitting time for the cycle $\mathbb{Z}_n$ is the same as the time taken for the Gambler's ruin chain to abandon the game. Hence,

$$t_{hit} = \frac{n^2}{4}$$

Hence, matthew's method tells us that

$$t_{cov} \leq \frac{n^2}{4}(\sum_{i=1}^{n} \frac{1}{i})$$

but that is far from the value of $\frac{n(n-1)}{2}$.
The reason why matthew's method performs poorly is because of the choice of σ in the proof of theorem 11.2. Picking a uniform random permutation of states leads to a lot of unnecessary counting. This is especially true in the steps where expectations are taken over $\sigma(k)$

Problem 3.2 (30 marks) *Ex 12.5 of LPW 2e.*

By Lemma 12.2, we know that

$$\frac{P^t(x,y)}{\pi(y)} = \sum_{j=1}^{|\mathcal{X}|} f_j(x) f_j(y) \lambda_j^t$$

$$\frac{P^t(x,x)}{\pi(x)} = 1 + \sum_{j=2}^{|\mathcal{X}|} f_j^2(x) \lambda_j^t$$

$$\frac{P^{t+k}(x,x)}{\pi(x)} = 1 + \sum_{j=2}^{|\mathcal{X}|} f_j^2(x) \lambda_j^{t+k}$$

Since $|\lambda_j| \leq 1$, if $k \geq 0$, then,

$$\frac{P^{t+k}(x,x)}{\pi(x)} = 1 + \sum_{j=2}^{|\mathcal{X}|} f_j^2(x) \lambda_j^{t+k} \leq 1 + \sum_{j=2}^{|\mathcal{X}|} f_j^2(x) \lambda_j^t = \frac{P^t(x,x)}{\pi(x)}$$

Hence, with $t = 2t$ and $k = 2$, part (a) is proved. Similarly, with $t = t$ and $k = 1$, part (b) is shown.

Part(c): We know that $\frac{P^t(x,y)}{\pi(y)} - 1 = \sum_{j=2}^{|\mathcal{X}|} f_j(x) f_j(y) \lambda_j^t = \sum_{j=2}^{|\mathcal{X}|} (f_j(x) \lambda_j^{\lceil t/2 \rceil - 1})(f_j(y) \lambda_j^{\lceil t/2 \rceil + 1})$ Now we have to show that

$$\sum_{j=2}^{|\mathcal{X}|} (f_j(x) \lambda_j^{\lceil t/2 \rceil - 1})(f_j(y) \lambda_j^{\lceil t/2 \rceil + 1}) \leq \sqrt{(\sum_{j=2}^{|\mathcal{X}|} (f_j(x) \lambda_j^{\lceil t/2 \rceil - 1})^2)(\sum_{j=2}^{|\mathcal{X}|} (f_j(y) \lambda_j^{\lceil t/2 \rceil + 1})^2)}$$

This is clearly true by a simple application of cauchy-schwarz inequality.
Cauchy-Schwarz:

$$|\sum u_i v_i|^2 \leq \sum |u_i|^2 \sum |v_i|^2$$

Apply cauchy schwarz with $u_i = (f_j(x) \lambda_j^{\lceil t/2 \rceil - 1})$, $v_i = (f_j(y) \lambda_j^{\lceil t/2 \rceil + 1})$ and summation starting with $i = 2$.

Problem 3.3 (40 marks) *This problem deals with randomness amplification in randomised algorithms. Suppose we have a randomised algorithm that picks a random value from a set V for use in its running. If the random value is picked from set B then the algorithm gives the wrong answer and if it is picked from $V \setminus B$ then the algorithm gives the right answer. We repeat the algorithm t times, picking a random value from V independently each time. Suppose that if the right answer appears even once then we recognise it and return it as the right answer, i.e. we make a mistake only if all t random values are picked from B. If $|B| = \beta|V|$ then the probability of making an error is β^t and the number of random bits used to achieve this is $t \log |V|$ (since picking a random number from a space of size n takes $\log n$ random bits.)*

Now, we do something different. We assume that we have a $d-regular$ graph with vertices V where $d = \theta(1)$. We pick the first value from V uniformly at random as before and generate the next $t-1$ random values by running a random walk on the graph, i.e., we run a Markov chain $(X_i)_{i\geq 0}$ that is the random walk on this graph and we feed the values $X_0, X_1, \ldots, X_{t-1}$ to the t-repetitions of the randomised algorithm, where X_0 is picked uniformly at random from V. Clearly the algorithm makes an error only if all of $X_0, X_1, \ldots, X_{t-1}$ are B. Note that since $d = \theta(1)$ the total number of random bits used is $\log |V| + (t-1)\log d$ which is significantly lower than before. Give an upper bound on the probability that this version of the algorithm makes a mistake. Use the spectral techniques learnt in Chap 12 of LPW for this purpose.

Let A be the adjacency matrix of the graph. The transition matrix, $P = d^{-1}A$. We use the same notation used in Chapter 12 of LPW.
Notice that $Pf = \lambda f$ means that $\lambda f(x) = \sum_{y:y\ x} f(y)$ for any eigen function f with eigen value λ. Let n be the number of states $= |\mathcal{X}|$
Notice that π, the stationary distribution is uniform. To aid with the answer, let us define R_B, a nxn matrix. It restricts any vector to B. Hence, $R_B[i,j] = 1$ if $i = j \in B$ and 0 otherwise.
The probability that at time t, the walk remains in B is the L_1 norm of $(R_BP)^t(R_B)\pi$. Notice that R_BP is not normalized, and $R_B\pi$ is the probability of having started the walk in that state such that the walk stays in B.
To bound the probability, we find bounds on general function $f : \mathcal{X} \to \mathbb{R}$. We know that the constant constant is an eigen function of P. Hence, we write $f = \alpha_1 c + \sum_{j=2}^{n} \alpha_j f_j = c + v$. Notice that $< c, v >_\pi = 0$. By the triangle inequality, $||R_BPf|| \leq ||R_BPc|| + ||R_BPv||$
Now we only need to bound $||R_BPc||$ and $||R_BPv||$. First, we tackle $||R_BPc||$. Since c is $\alpha_1 f_1$, $Pc = c$, and R_B restricts the vector to B. We can see that $||R_BPc|| = \beta||R_Bc||$. Next, we look at $||R_BPv||$. Let λ_2 be the largest (absolute value) eigen value after 1. $||R_BPv|| \leq \lambda_2||R_Bv||$. Therefore, $||R_BPf|| \leq (\beta + \lambda_2)||f||$. Applying that inequality iteratively, we get that $||(R_BP)^tR_B\pi|| \leq (\beta + \lambda_2)^t\beta$
Therefore, the probability of the algorithm making a mistake is bounded by $\beta(\beta + \lambda_2)^t$
