Homework # 9

Reading

- Sections 13.2.2 discusses forward-backward HMM equations. Section 13.2.4 discusses scaling factors and 13.2.5 the Viterbi algorithm. (The other sections in 13.2 relate to EM, which we will cover next week.)
- See below regarding the SNE and t-SNE articles.

1. In this problem we will revisit the cheating casino model of Problem 1 in homework 8. Let $Y(0), Y(1), \ldots, Y(200)$ be the die rolls you generated in $1(a)$ of homework 8. (In the lecture I used $Z(t)$ as the hidden state, but here to match hw 8 I'll use $X(t)$ as before.

   (a) Derive the forward algorithms for computing,
   $$\alpha_t = P(X(t) \mid Y(0), \ldots, Y(t)). \tag{1}$$
   Write an R/Python function **Forward(Y)** that accepts a vector Y giving the values $Y(0), Y(1), \ldots, Y(200)$ and returns α_t for all values of t.

   (b) Derive the backward algorithm for computing,
   $$\beta_t = P(Y(t+1), Y(t+2), \ldots, Y(200) \mid X(t)). \tag{2}$$
   Write an R/Python function that computes β_t over all t given the die roll data.

   (c) Write an R/Python function that computes the smoothing probability,
   $$P(X(t) = C \mid Y(0), Y(1), \ldots, Y(T)) \tag{3}$$
   Plot the smoothing probability as a function of t and compare to your estimate in hw 8.

2. Consider the continuous state Markov chain $X(t+1) = rX(t) + \mathcal{N}(0, \sigma^2)$. Assume $X(0) = x_0 \in \mathbb{R}$.

   (a) Explain why $X(t)$ is always normally distributed. Then compute the mean and variance of $X(t)$. (Hint: you can express the mean and variances of $X(t)$ in terms of the mean and variances of $X(t-1)$.

(b) Using a), determine the distribution $\pi = \lim_{t\to\infty} X(t)$. Write down the kernel $K(s_0, s_1) = P(X(1) = s_1 \mid X(0) = s_0)$ and show,

$$\int_{-\infty}^{\infty} K(s_0, s_1)\pi(s_0)ds_0 = \pi(s_1). \tag{4}$$

In words, π is the stationary distribution of $X(t)$.

3. Attached you will find two papers describing the SNE and t-SNE dimensional reduction algorithms. The SNE algorithm was introduced in 2002 and the t-SNE algorithm was introduced as an improvement on SNE in 2008. Here we will implement the SNE paper, just for the sake of simplicity given the limited time. Both algorithms attempt to find a low-dimensional non-linear space that describes the data, as opposed to PCA which attempts to find a linear subspace. Your goal below will be to apply the SNE algorithm with steepest descent. Watch the first 25 minutes of this video on t-SNE, which essentially applies to SNE as well.

`https://www.youtube.com/watch?v=RJVL80Gg3lA&list=UUtXKDgv1AVoG88PLl8nGXmw`

(a) In the SNE paper, equation (4) gives the loss function the authors are trying to optimize. They call it a cost function. Explain the authors' motivations in using this cost function. What are the variables in the cost function? In other words, what are we optimizing?

(b) Derive equation (5) which gives a formula for the gradient of the cost function. As a start, you may want to assume that we are reducing to a single dimension, making each y a scalar, but then make sure to generalize to any dimension.

(c) Write R functions to compute the cost function and the gradient of the cost functions given data points x and the dimensional reduced points y.

(d) Using steepest descent, compute SNE to dimensionally reduce the dataset of hw 8, problem 2 to 1 and 2 dimensions. Plot the 1-d and 2-d reductions produced by SNE and compare to your results in hw 8.