Statistical Computing 2

숙제 6
2019년 가을학기
응용통계학과 석사과정 최석준

1. Using slice sampler, generate standard normal samples.

나중에 다른 케이스에 slice sampler 를 쓸 일이 있을까 싶어서 slice sampler 의 약간 추상적인 버전을 만들었다.

Pdf 가 multimodal 이어서 slice step 에서 뽑힌 특정 lower-bound 값에서 slice 가 여러 개 나올 경우를 대비해 놓았다. 조금 더 자세히 설명하면 slice 에서 sample 을 uniform 하게 뽑는 2 번째 step 을 두 단계로 나누어 구현했는데, 먼저 여러 개의 slice 중에서 각각의 길이를 weight 로 이용하여 하나를 선택하고 (r 의 sample 함수가, python standard library 에 random.choices 로 구현되어 있어 가져다 썼다.), 선택된 slice 에서 uniform 으로 뽑는다. 따라서 결과적으로는 전체 slice 들에서 uniform 하게 하나 draw 하는것과 같다.

[Python]

```python
#python 3 file created by Choi, Seokjun

from math import exp, log, sqrt
from random import uniform, choices, seed
from statistics import mean, variance
import time

import matplotlib.pyplot as plt


class SliceSampler:
    def __init__(self, target_pdf, get_interval_func, initial):
        self.target_pdf = target_pdf
        self.get_interval = get_interval_func
        self.samples = [initial]

    def sampler(self, last):
        #slicing
        prob_lower_bound = uniform(0, self.target_pdf(last))
        interval_list = self.get_interval(prob_lower_bound)

        #draw from slice
        interval_weight = [interval[1]-interval[0] for interval in interval_list]
        chosen_interval = choices(interval_list, weights=interval_weight)[0]
        # print(chosen_interval)
        new_sample = uniform(chosen_interval[0], chosen_interval[1])
        return new_sample

    def generate_samples(self, num_samples):
        start_time = time.time()
        for _ in range(num_samples):
            last = self.samples[-1]
            self.samples.append(self.sampler(last))
        elap_time = time.time()-start_time
```

```
            print("iteration", num_samples, "/", num_samples,
                  " done! (elapsed time for execution: ", elap_time//60,"min ", elap_time%60,"sec)")

        def get_samples(self):
            return self.samples

        def show_hist(self):
            plt.hist(self.samples, bins=100)
            plt.show()

        def get_moments(self):
            return (mean(self.samples), variance(self.samples))
```

이제 이번 숙제의 목적인, standard normal sample 을 뽑자. 구현은 수업시간에 본 것과 마찬가지로 상수 뗀 경우로 하였다. 20 만개 생성한다. 초기값은 임의로 0 으로 두었다.

```
[Python]
if __name__ == "__main__":
    seed(2019-311252)

    def standard_normal_pdf(x):
        return exp(-0.5*(x)**2)

    def standard_normal_get_interval(prob):
        upper_bound = sqrt(-2*log(prob))
        lower_bound = -upper_bound
        return [(lower_bound, upper_bound)]

    StdNormal_SliceSampler = SliceSampler(standard_normal_pdf, standard_normal_get_interval, 0)
    StdNormal_SliceSampler.generate_samples(200000)
    print("sample mean and variance:", StdNormal_SliceSampler.get_moments())
    StdNormal_SliceSampler.show_hist()
```

마지막 2 줄은 생성된 sample 의 평균/분산과 histogram 을 출력한다. 잘 만들어진 것을 확인할 수 있다.

```
[Console]
iteration 200000 / 200000   done! (elapsed time for execution:   0.0 min   0.885120153427124 sec)
sample mean and variance: (-0.0010359665909823285, 1.0095921573850073)
```