**고급소프트웨어실습1 5주차 과제**

컴퓨터공학 20172141 김미소

**[프로그램 소개]**

이 프로그램은 CurveSampling을 통해 만들어진 임의의 함수를 샘플링한 결과를 통해 pdf를 만들어내고 이 pdf 값(분포)에 따라 난수를 생성해내는 시뮬레이션을 수행하는 프로그램이다.

**[프로그램 구동 방법]**

1. 먼저 CurveSampling.exe를 실행하여 곡선을 그리고 100개로 샘플링하여 결과를 sampling.txt로저장한다.

2. sampling.txt를 프로젝트 폴더 내에 sampling\_table.txt로 복사하여 사용한다.

3. ctrl+F5를 눌러 프로그램을 실행하면 구간 별 적분 결과를 알 수 있고 샘플링 지점에 대하여 정규화 된 pdf 값은 pdf\_table.txt에 저장되므로 이 파일을 열람하여 확인할 수 있다.

4. 난수를 생성할 개수를 입력한다. 입력 값은 25 이상을 추천한다.

5. 생성된 난수 결과는 random\_event\_table.txt에 저장되므로 이 파일을 열람하여 확인할 수 있다.

6. 숙제 2\_1번에 대하여 λ 값은 직접 입력하여 사용할 수 있도록 구현하였다. 3번 입력할 수 있다. 해당 λ 값에 대하여 임의로 지정한 개수인 100개의 난수를 확인할 수 있고 그에 대한 평균, 분산 값을 알 수 있다.

7. 숙제 2\_2에 대하여 2\_2\_a 부분이 먼저 실행되는데, 이는 실습 2-2에서 사용했던 방법을 조금 개선한 것으로 사용법은 동일하다. 생성할 난수 개수를 입력하면 된다. 8번의 히스토그램을 확인하기 위하여 생성 개수는 1000개 이상을 추천한다.

8. 7번을 수행하고 나면 프로젝트 폴더 내에 histogram.txt가 나온다. 이 파일을 열람하여 난수 생성이 분포에 맞게 제대로 되었는지 확인할 수 있다.

9. 숙제 2\_2에 대하여 2\_2\_b 부분이 실행되게 되는데 2\_2\_a와 마찬가지로 생성할 난수 개수를 입력하면 된다. 이는 Bisection 방법을 Secant 혹은 Newton-Raphson 방법으로 대치시킨 것으로 둘 다 구현하였으니 원하는 방법으로 골라서 사용하면 된다. 역시 난수 생성 개수는 1000개 이상을 추천한다.

**[실습 2-1]**

**확률 밀도 함수 형태의 곡선 설계**

CurveSampling.exe 파일을 통해 곡선을 그리고 샘플링 개수를 100개로 지정하여 샘플링 한 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

위와 같이 간격이 5.484848인 형태로 100개의 샘플링 지점을 얻어낼 수 있었다.

이 샘플링 지점을 가지고 합성 사다리꼴 공식을 사용하여 수치적으로 적분하면 1이 아닌 매우 큰 숫자가 나오게 된다. 그러나 pdf의 적분 결과는 1이 나와야 하므로 각 함수 값을 합성 사다리꼴 공식을 통해 구한 수치 적분 값으로 나누어 pdf를 적분하였을 때 1이 나오도록 정규화 하였다.

뿐만 아니라 x 범위도 [0.0, 1.0]이 되도록 정규화해야 하였으므로 최소-최대 정규화를 통해 x값을 조정하고 이에 따른 pdf 값도 다시 조정하였다.

텍스트이(가) 표시된 사진

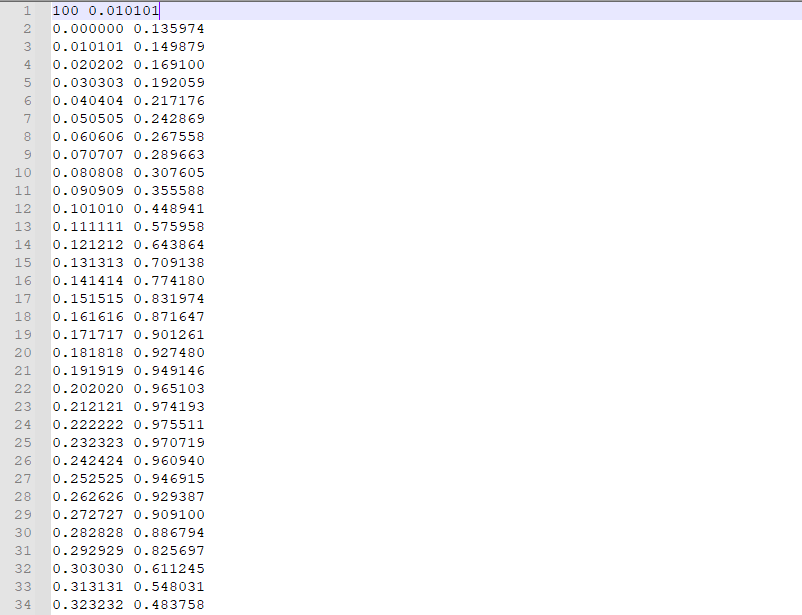
자동 생성된 설명

X 값을 (maxX - minX)로 나누었으니 y값은 곱해주어 다시 수치적분을 했을 때 1이 성공적으로 나올 수 있도록 하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

구간 별 적분 결과는 위와 같다.



그리고 정규화를 통하여 pdf\_table.txt에 저장한 값은 위와 같다.

**[실습 2-2]**

Pdf\_table.txt에 저장된 값을 기반으로 cdf를 구하여야 한다. cdf는 누적 분포 함수이므로 적분 값을 누적으로 저장한 값이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Cdf 값은 위처럼 이전에 구한 cdf값에 다음 구간에 대한 적분 값을 더하는 방식으로 구할 수 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

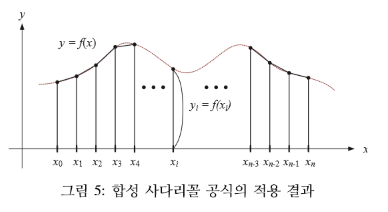
난수 생성 방법은 위와 같은 함수를 사용하였다.



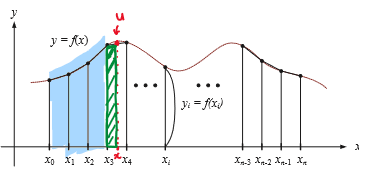
그리고 난수가 0과 1 사이의 값이 나올 수 있도록 생성한 난수를 MY\_RAND\_MAX로 나누어 사용하였다.

이번 실습은 입력 확률 밀도 함수를 시뮬레이션 해주는 Nr개의 난수 X1, X2, … XNr-1을 발생시켜 random\_event\_table.txt에 저장하는 것이다. 생성되는 임의의 값 u를 통해 해당 X값을 찾는 문제이다. 따라서 F(x) = u를 만족시키는 x 값을 찾는 문제이고 이는 F(x)-u = 0 방정식을 풀어 x를 찾을 수 있다. 방정식을 푸는 방법은 Bisection Method를 사용한다.

우선 cdf를 구하는 F 함수에 대하여 먼저 정의해야 하는데



위와 같은 곡선이 있다고 가정했을 때 우리가 뽑아낸 난수 u가 px(x3)과 px(x4) 사이에 있는 수라면 이 u에 대한 x값은 x3과 x4사이에 있을 것이다. 이 x에 대한 F 값은 F(x3)에 x3과 x 사이의 면적을 더해주면 된다.



즉 파란색 부분 F(x3)과 초록색 부분(x3과 x사이의 면적)을 더하면 된다.



초록색 부분의 면적은 위의 공식을 이용하여 구할 수 있고 결국 x에 대한 F 값을 찾는 코드는 아래와 같다.

텍스트이(가) 표시된 사진

자동 생성된 설명

우리는 x를 Bisection 방법을 통하여 구하기로 하였으므로

텍스트, 검은색, 화면, 스크린샷이(가) 표시된 사진

자동 생성된 설명

반복문을 계속 돌면서 값을 찾고 종료조건에 다다르면 값 찾기를 종료한다. 여기서 a와 b의 초기값은 각각 0과 1이다. 0과 1의 구간 내에서 Bisection Method를 계속 진행하면서 정답 xn을 찾는 것이다.

이 방법으로 난수 X1, X2, …, XNr-1을 생성한 결과는 아래와 같다.

테이블이(가) 표시된 사진

자동 생성된 설명

**[숙제 2-1]**

**(i) 지수 분포의 인자 λ (> 0)에 대해 확률 변수 X의 기대값이 E[X] = 이고 분산이 Var[X] = 이라는 사실을 고려하여, 세 가지 서로 다른 λ 값을 설정하라.**



λ 값은 직접 입력하여 사용할 수 있도록 구현하였다. 3번 입력할 수 있다.

**(ii) 자신이 정한 λ에 해당하는 지수 분포 각각에 대하여, inversion 방법을 사용하여 자신이 정한 충분히 큰 개수만큼 이 지수 분포를 따르는 난수를 생성하라.**



지수 분포에 대한 역함수는 위와 같다. 따라서 이 식을 이용하여 X를 찾는다.

텍스트이(가) 표시된 사진

자동 생성된 설명

코드는 위와 같이 작성하였다. hw\_rand는 기존의 rand와 같은 함수이다. 난수는 100개를 생성하기로 하였고 그에 대한 난수 생성 결과는 콘솔에 출력된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

100개가 다 출력이 되지만 스크린샷은 일부만 첨부하겠다.

**(iii) 자신이 생성한 난수들을 사용하여 평균 값과 분산 값을 구한 후 이론적인 값들과 얼마나 일치하는지 분석하라.**

텍스트이(가) 표시된 사진

자동 생성된 설명

평균과 분산을 구하는 방법은 해를 계산하여 해의 합과 해의 제곱의 합을 임의의 변수에 더하여 저장한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그리고 이 해의 합들을 가지고 평균을 구하고, 해의 제곱의 합과 평균을 통하여 분산을 구하였다.

λ = 3에 대하여 100개의 난수를 생성한 결과,

텍스트이(가) 표시된 사진

자동 생성된 설명

오차가 평균에 대하여 0.02 정도로 나타나고 분산에 대하여 0.008 정도로 이론적인 값과 어느정도 가까운 값을 도출하고 있다.

λ = 20에 대하여 100개의 난수를 생성한 결과,

텍스트이(가) 표시된 사진

자동 생성된 설명

오차는 평균에 대하여 0.008 정도이고 분산에 대하여 0.0006 정도로 이론적인 값과 어느정도 가까운 값을 도출한다.

λ = 100에 대하여 100개의 난수를 생성한 결과,

텍스트이(가) 표시된 사진

자동 생성된 설명

오차는 평균에 대하여 0.00005정도이고 분산에 대하여 0.000004정도로 이론적인 값과 꽤 가까운 값을 도출한다.

**과연 얼마나 많은 난수를 생성해야만 통계적으로 구한 평균 값과 분산 값이 이론적인 평균 값과 분산 값과 충분히 일치하게 되는지 명확히 분석하여 보고서에 기술하라.**

|  |  |
| --- | --- |
|  | 10개의 난수를 생성했을 때  λ = 3에 대한 오차  - 평균: 0.058276…  - 분산: 0.073272…  λ = 20에 대한 오차  - 평균: 0.026665…  - 분산: 0.002202…  λ = 100에 대한 오차  - 평균: 0.001910…  - 분산: 0.000051… |
|  | 100개의 난수를 생성했을 때  λ = 3에 대한 오차  - 평균: 0.027962…  - 분산: 0.015429…  λ = 20에 대한 오차  - 평균: 0.000307…  - 분산: 0.000847…  λ = 100에 대한 오차  - 평균: 0.000685…  - 분산: 0.000022… |
|  | 10000개의 난수를 생성했을 때  λ = 3에 대한 오차  - 평균: 0.001795…  - 분산: 0.005162…  λ = 20에 대한 오차  - 평균: 0.000471…  - 분산: 0.000089…  λ = 100에 대한 오차  - 평균: 0.000033…  - 분산: 0.000000044… |

난수 생성 개수가 늘어남에 따라 오차가 줄어드는 것을 확인했다. 그리고 λ 값도 커질수록 오차가 줄어드는 것을 확인했다. 오차 허용 범위를 얼마나 두는가에 따라 결과가 달라지겠지만 같은 λ값에 대하여 비교하여 보면

텍스트이(가) 표시된 사진

자동 생성된 설명

100만개 정도의 난수를 생성했을 때 이론적인 값과 충분히 일치하는 값을 얻을 수 있을 것이라 판단된다.

**[숙제 2-2]**

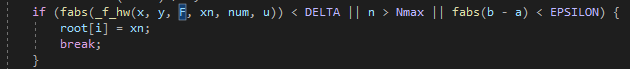
**(i) 여러분이 실습 시간에 작성한 코드 (프로그램 2-2)를 잘 가다듬어 문제없이 그리고 효율적으로 수행되도록 수정하라 (프로그램 2-2(a)).**

Bisection Method를 수행하면서 종료조건에 다다르면 즉시 출력하고 break를 통하여 반복문을 탈출하였는데 수행 시간에 입출력 부분을 제외하기 위하여 종료조건에 다다르면 임의의 배열에 저장해두고 반복문을 탈출하는 방식으로 변경하였다.

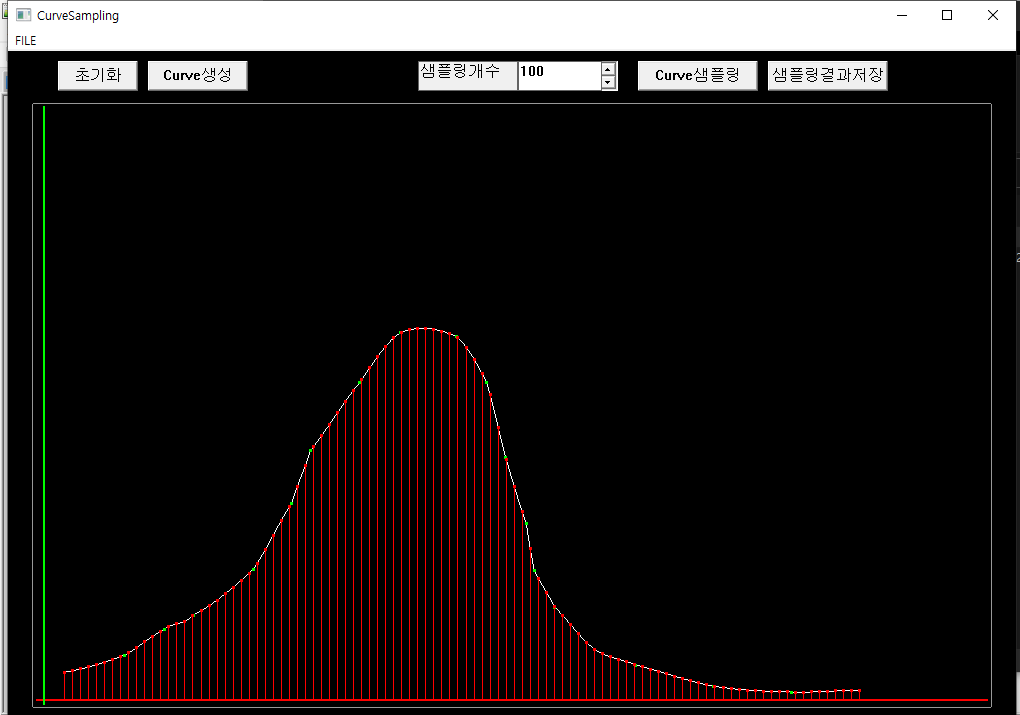
텍스트이(가) 표시된 사진

자동 생성된 설명

↓



**(ii) 주어진 확률 밀도 함수에 대하여 자신의 코드가 올바르게 난수를 생성하는지 통계적으로 확인하라. 이를 위하여 아래에서 설명하는 내용의 코드 (프로그램 2-3)을 작성한 후 이를 활용하라.**



해당 커브로 새로 샘플링하고 program2\_2\_a() 함수에서 구현한 방식으로 새로 난수를 생성해 보았다. 난수는 1000개를 생성하였고, 그 결과 히스토그램은

테이블이(가) 표시된 사진

자동 생성된 설명

위와 같은 형태로 도출되었다. 완벽히 똑같은 굴곡을 가진 곡선은 아니지만 실제 곡선과 상당히 비슷한 결과이다.

테이블이(가) 표시된 사진

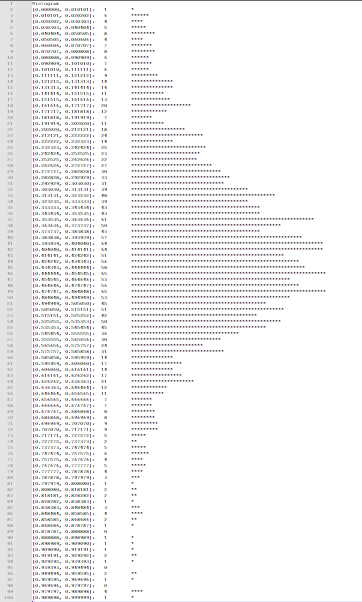
자동 생성된 설명

히스토그램은 이러한 양식으로 출력하였다.

**[시작구간, 끝 구간]: 구간 별 난수 개수**이다.

**(iii) 비선형 방정식의 근을 구하는 방법 구현 시 앞에서 사용한 Bisection 방법을 Secant 방법으로 대치한 난수 생성 프로그램을 작성하라 (프로그램 2-2(b)). 물론 자신의 프로그램이 제대로 작동하는지 실험적으로 확인하라.**

Bisection Method, Secant Method, Newton Method 방법에 대한 결과를 모두 histogram.txt 파일로 출력해 결과를 확인하였고 히스토그램 출력 결과가 곡선과 대략적으로 일치함을 확인할 수 있었다.



왼쪽 그림은 Bisection 방법으로 출력한 히스토그램이고, 오른쪽 그림은 Secant 방법으로 출력한 히스토그램이다.

테이블이(가) 표시된 사진

자동 생성된 설명

왼쪽 그림은 Bisection 방법으로 출력한 히스토그램이고, 오른쪽 그림은 Newton-Raphson 방법으로 출력한 히스토그램이다.

Secant 방법과 Newton-Raphson 방법을 수행하기 전 Bisection 방법을 몇 번 수행하여 각 방법의 초기값을 세팅하였다.

텍스트, 화면이(가) 표시된 사진

자동 생성된 설명

Bisection 방법을 delta와 epsilon이 0.00001정도로 정하여 종료조건을 지정하였다. 그 결과로 나온 a(시작 구간)과 b(끝 구간)에 대하여 Secant 방법에서 x0, x1으로 설정하였고 Newton-Raphson 방법에서는 a와 b의 중점을 초기값으로 설정하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트, 실내이(가) 표시된 사진

자동 생성된 설명

**(iiii) 충분히 큰 난수의 개수 nr에 대해 프로그램 2-2(a)와 프로그램 2-2(b)를 수행시킨 후, 각 방법이 난수를 생성하는데 걸린 시간을 비교하라.**

2000개의 난수를 생성한 결과,

텍스트이(가) 표시된 사진

자동 생성된 설명

Bisection 방법과 Secant 방법을 비교하였을 때, Bisection 방법을 몇 번 수행한 후 Secant 방법을 수행하는 것이 수행시간이 훨씬 줄어드는 것을 확인할 수 있었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이는 Bisection 방법과 Newton-Raphson 방법을 비교한 것인데, Bisection 방법을 몇 번 수행한 후 Newton-Raphson 방법을 수행하는 것이 수행시간이 훨씬 줄어드는 것을 확인할 수 있었다