**고급소프트웨어실습1 [5주차과제]**

특정 확률 사건의 생성

학과 : 컴퓨터공학과

분반 : 2

학번 : 20181593

이름: 계인혜

보고서에 들어가기 전에 히스토그램의 구간은 200개로 나누었으며 각 구간의 길이는 1/200 = 0.005이다.

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

double linearEquation(double\* fx, double\* fy, int idx, double x1) {

// 선형보간 : (xm, pxm), (xm+1, pxm+1)을 지나는 직선의 방정식

return (((fy[idx + 1] - fy[idx]) / (fx[idx + 1] - fx[idx])) \* (x1 - fx[idx])) + fy[idx];

}

double bisectionMethod(double\* fx, double\* fy, int n, double h, double u) {

double x0 = 0.0, x1 = 1.0;

double a0 = 0.0, b0 = 1.0;

int i, a, s;// a : 직전의 샘플링 지점

x1 = (a0 + b0) / 2;

for (i = 0; ; i++) {

for (a = 0; a < n; a++) {

if (a \* h <= a0 && a0 < (a + 1) \* h) break;

}

for (s = 0; s < n; s++) {

if (s \* h <= x1 && x1 < (s + 1) \* h) break;

}

double tmp = linearEquation(fx, fy, s, x1);

if (fabs(tmp - u) < DELTA || i >= Nmax || fabs(x1 - x0) < EPSILON) {

break;

}

double t1, t2;

t1 = linearEquation(fx, fy, a, a0) - u;

t2 = linearEquation(fx, fy, s, x1) - u;

if (t1 \* t2 < 0) b0 = x1;

else a0 = x1;

tmp = (a0 + b0) / 2;

x0 = x1;

x1 = tmp;

}

return x1;

}

double trapezoidal(double\* ax, double\* ay, int n, double h, double start, double end) {

double area = 0;

int e;

for (e = 0; e < n; e++) {

if (e \* h <= end && end < (e + 1) \* h) break;

}

for (int i = 0; i <= e; i++) {

if (i == 0 || i == e) area += ay[i];

else area += 2 \* ay[i];

}

area = area / 2 \* h;

return area;

}

void program2\_2()

{

FILE\* fp\_r, \* fp\_w;

fp\_r = fopen("pdf\_table.txt", "r");

fp\_w = fopen("random\_event\_table.txt", "w");

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int n, userN;

double h, u;

double x = 0, tmp = 0;

scanf("%d", &userN);

fscanf(fp\_r, "%d %lf", &n, &h);

double\* ax = (double\*)malloc(sizeof(double) \* n);

double\* ay = (double\*)malloc(sizeof(double) \* n);

double\* fx = (double\*)malloc(sizeof(double) \* n);

double\* fy = (double\*)malloc(sizeof(double) \* n);

for (int i = 0; i < n; i++) {

fscanf(fp\_r, "%lf %lf", &ax[i], &ay[i]);

fx[i] = ax[i];

fy[i] = trapezoidal(ax, ay, n, h, 0, fx[i]);

}

for (int i = userN; i > 0; i--) {

u = (double)rand() / RAND\_MAX;

x = bisectionMethod(fx, fy, n, h, u);

printf("%.15lf\n", x);

fprintf(fp\_w, "%.15lf\n", x);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

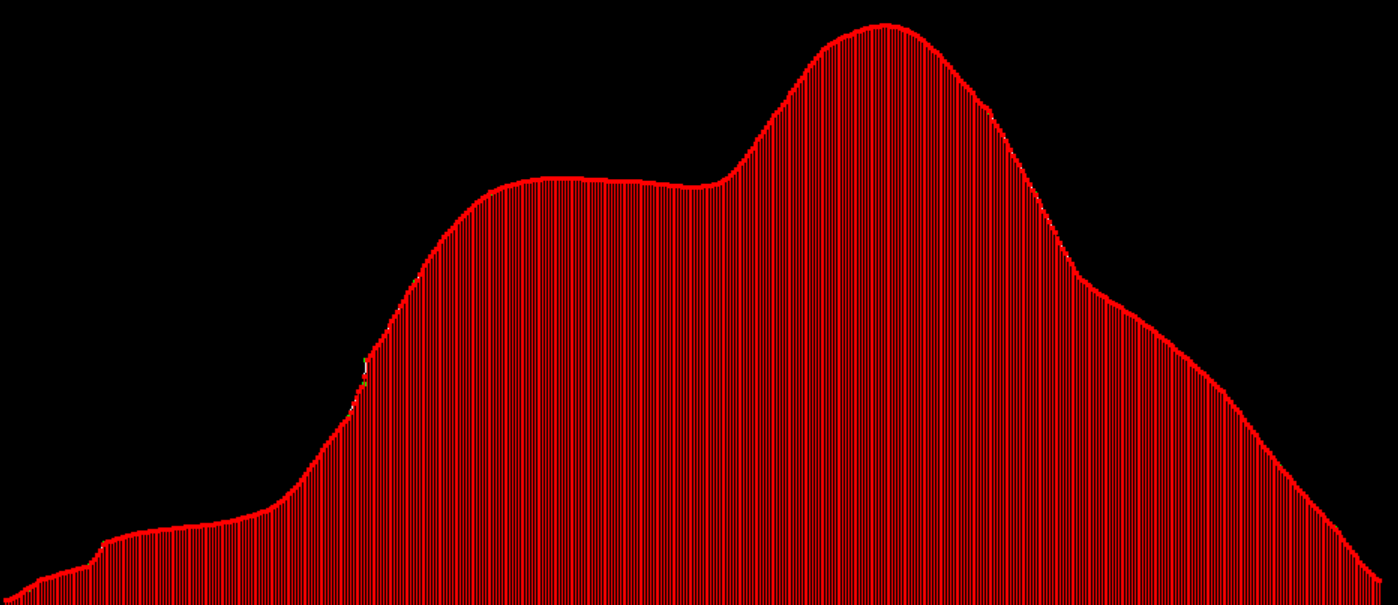
if (fp\_r != NULL) fclose(fp\_r);

if (fp\_w != NULL) fclose(fp\_w);

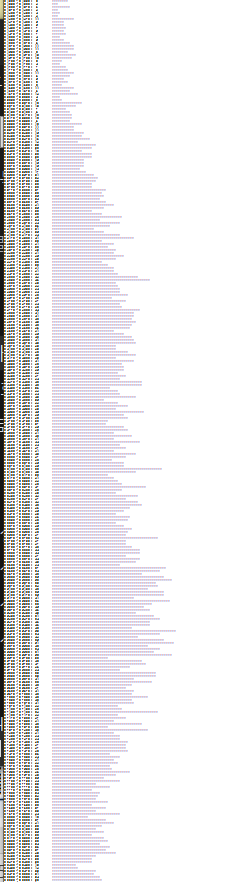
}

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

숙제 코드로 구한 난수 배열이 확률 밀도 함수(pdf)를 따르는지 확인해보자. [0,1]까지의 정의역을 길이가 0.005인 부분 구간 200개로 나누었다. 난수의 개수는 10000개로 설정하였다.



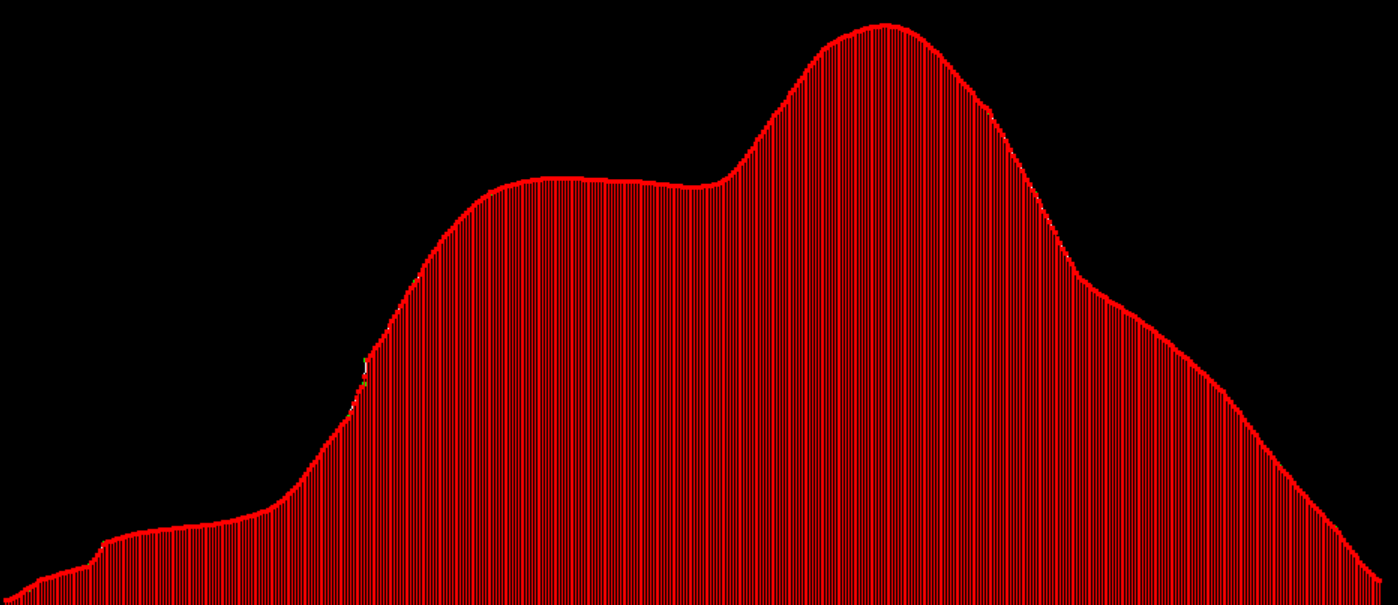
위와 같이 샘플링한 그래프를 이용하여 코드를 돌려본 결과 histogram.txt에 출력된 결과물은 아래와 같다.



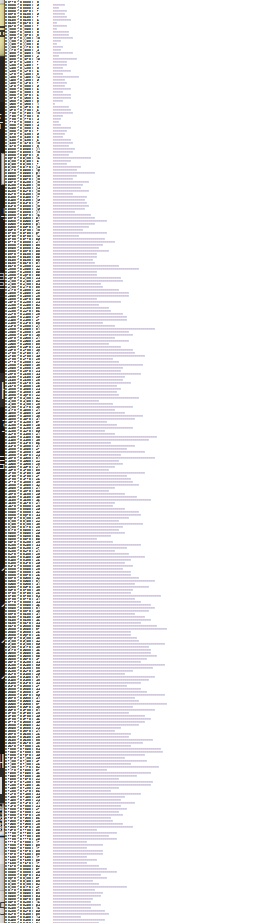
몇몇 구간에서 예외적으로 높거나 낮은 값을 보이기는 하지만 대체로 샘플링한 그래프와 개형이 비슷하다는 것을 확인할 수 있다.

**숙제 2-2(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.**

숙제 코드로 구한 난수 배열이 확률 밀도 함수(pdf)를 따르는지 확인해보자. [0,1]까지의 정의역을 길이가 0.005인 부분 구간 200개로 나누었다. 난수의 개수는 10000개로 설정하였다.

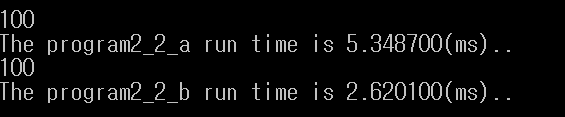
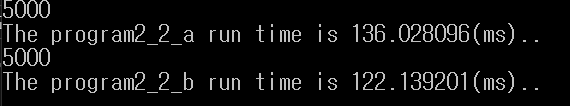
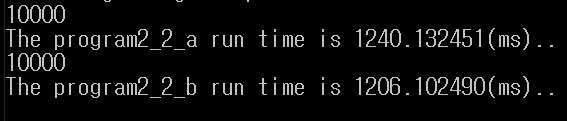


위와 같이 샘플링한 그래프를 이용하여 코드를 돌려본 결과 histogram.txt에 출력된 결과물은 아래와 같다.



몇몇 구간에서 예외적으로 높거나 낮은 값을 보이기는 하지만 대체로 샘플링한 그래프와 개형이 비슷하다는 것을 확인할 수 있다.

**숙제 2-2(iv)** **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.**

****

Program2\_2\_a는 bisection method를 사용한 것이고, program2\_2\_b는 secant method를 사용한 것이다. 결과로 보아 secant method를 사용한 방식이 bisection method를 사용한 것보다 시간이 적게 걸렸음을 확인할 수 있다.