

# [CSE 4152] 고급 소프트웨어 실습 I

## 『GPS 수신기 위치 계산 문제』

### 수치 컴퓨팅 실험 2: 특정 확률 사건의 생성

담당교수: 컴퓨터공학과 임인성 (AS-905, 02-705-8493, ihm@sogang.ac.kr)

담당조교: 컴퓨터공학과 안재풍 (AS-914, 02-711-5278, ajp5050@sogang.ac.kr)

### Why randomisation and probabilistic techniques?

*Probability theory is used widely in such areas of study as mathematics, statistics, finance, gambling, science (in particular physics), artificial intelligence/machine learning, computer science, game theory, and philosophy to, for example, draw inferences about the expected frequency of events. Probability theory is also used to describe the underlying mechanics and regularities of complex systems. (from Wikipedia).*

Some computer science and engineering applications:

- Computer architecture, operating systems, distributed systems
- Network protocols, analysis of Web traffic
- Computer security
- Software engineering
- Data mining
- Bioinformatics
- Machine learning
- Other applications

## 1 확률 분포와 확률 사건

다음은 여러분들이 중고등학교 시절부터 배워온 확률에 관한 글이다.

확률론 (*probability theory*)은 확률에 대해 연구하는 수학의 한 분야이다. 확률론은 비결정론적 (*nondeterministic*) 현상을 수학적으로 기술하는 것을 목적으로 하며, 주요 연구 대상으로는 확률 변수 (*random variable*), 확률 과정 (*stochastic process*), 사건 (*event*) 등이 있다. 확률론은 통계학의 수학적 기초이다. 또한 인간은 살아가기 위해 매 순간 칙칙적으로 예측할 수 없는 방법으로 변화하는 환경에 대처하여 결정을 내릴 필요가 있으며, 이는 의식적으로나 무의식적으로나 확률론에 기반하게 된다. 통계역학 등에서 완전한 정보가 알려지지 않은 복잡계를 기술하는 데에도 확률론적 방법론은 큰 역할을 한다. 이에 더해 20세기 초에 등장한 물리학 이론인 양자역학은 미시계의 물리적 현상이 근본적으로 확률적인 본질을 갖고 있음을 알려주었다.<sup>1)</sup>

우리가 고급 소프트웨어를 제작할 때 특정 확률 분포 (*probability distribution*)를 따르는 확률 사건 (*stochastic event*)을 수치적으로 충실히 모사 즉 시뮬레이션 해야하는 경우가 종종 발생한다. 예를 들어, 어떤 컴퓨팅 시스템을 설계하였을 때, 그 설계가 적절한지를 확률적인 방법을 사용하여 시스템 성능을 미리 시뮬레이션 해보거나, 컴퓨터 게임 소프트웨어를 제작할 때 컴퓨터가 모사해주는 상대방의 행동이 분명히 어떤 특징을 가지지만 예측하기 힘든 형태로 행위가 이뤄지게 하려면 확률적인 기법을 적용해야 한다.

이번주에는 지난 주에 학습한 비선형 방정식의 풀이 기법의 한 응용으로서, 사용자가 GUI (Graphics User Interface) 소프트웨어를 사용하여 설계한 특정 확률 분포를 가지는 확률 사건을 시뮬레이션 해주는 방법에 대하여 학습하도록 한다. 이를 위하여 먼저 기본적인 확률 이론에 대해 기억을 되살려 보자.

## 2 확률 변수, 그리고 확률 밀도 함수와 누적 분포 함수

확률 변수 (*random variable*)는 주어진 범위 내에서 임의의 값을 가질 수 있는 변수로서, 각 값을 가질 확률이 정확히 정의되어 있다. 예를 들어, 주사위를 던질 때 나오는 눈의 수는 확률변수가 되며, 이때 그 값이 1일 확률은  $1/6$ 이 됨은 잘 알고 있을 것이다. 이와 같이 확률 변수가 한 개씩 나열할 수 있는 값을 가질 때, 이산 확률 변수 (*discrete random variable*)라고 하는데, 여기서는 실수 공간처럼 연속 공간에서 정의된 연속 확률 변수 (*continuous random variable*)를 고려하자.

1) 출처: <http://ko.wikipedia.org/wiki/확률론>

임의의 실수 값을 가질 수 있는 확률 변수  $X$ 가 어떠한 확률로 특정 값을 가질지를 결정하는 함수가 바로 확률 밀도 함수 (probability density function, pdf)이다.  $p_X(x)$ 를 실수 공간에서 정의된 확률 변수  $X$ 의 확률 밀도 함수라 할 때, 이 함수는 다음과 같은 조건을 만족해야 한다.

- (a) 모든 실수 값  $x$ 에 대해,  $p_X(x) \geq 0$
- (b)  $\int_{-\infty}^{\infty} p_X(x) dx = 1$

예를 들어,  $[0, 1]$  구간에 대해 정의된 균등 분포 (uniform distribution)를 따르는 균등 확률 변수 (uniform random variable)의 확률 밀도 함수는 다음과 같다.

$$p_X(x) = \begin{cases} 1, & \text{if } 0 \leq x \leq 1, \\ 0, & \text{otherwise.} \end{cases}$$

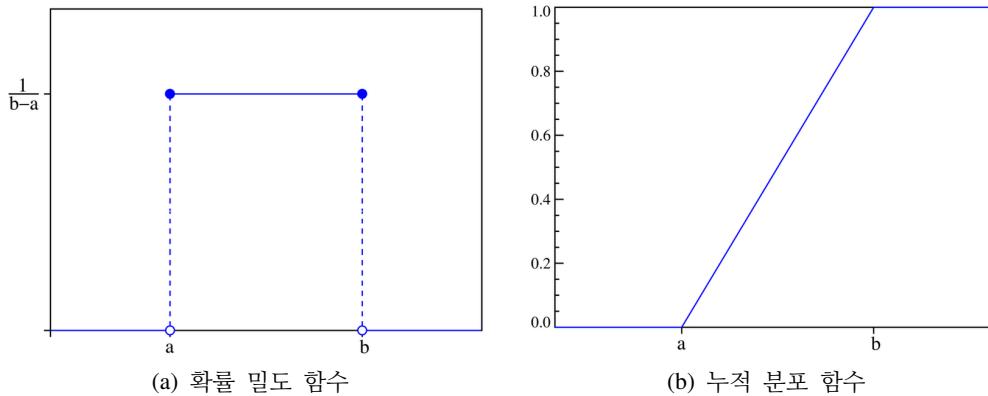


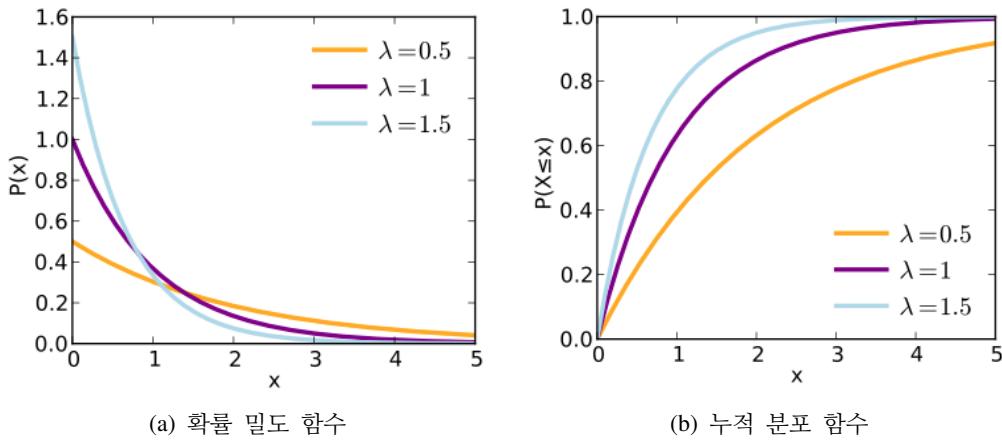
그림 1: 균등 분포의 정의 (출처: <http://en.wikipedia.org>).

또한 확률 과정 (stochastic process)의 시뮬레이션에 흔히 쓰이는 지수 분포 (exponential distribution)는 다음과 같은 확률 밀도 함수로 정의된다.

$$p_X(x) = \begin{cases} \lambda e^{-\lambda x}, & \text{if } x \geq 0, \\ 0, & \text{otherwise.} \end{cases}$$

확률 밀도 함수는 정확히 말해서 확률값을 나타내는 함수가 아니라 확률 변수  $X$ 가 특정 값을 가질 정도를 나타내주는 함수이기 때문에, 0과 1 사이의 값을 가져야 하는 확률값과는 달리 확률 밀도 함수값은 1보다 큰 값을 가질 수도 있다. 그러면 실제 확률값과 확률 밀도 함수값은 어떤 관계가 있을까? 이는 다음과 같은 조건에 의해 정의된다.

$P(x \leq X \leq x + \varepsilon) \approx p_X(x) \cdot \varepsilon$

그림 2: 지수 분포의 정의 (출처: <http://en.wikipedia.org>).

이는 아주 작은 값인  $\varepsilon$ 에 대해 확률 변수  $X$ 가  $[x, x+\varepsilon]$ 이라는 작은 구간에서의 값을 가질 확률이 근사적으로  $p_X(x) \cdot \varepsilon$  정도가 된다는 뜻으로서, 확률 밀도 함수값  $p_X(x)$ 은  $X$ 가  $x$  부근의 값을 가질 실제 확률값과 밀접한 연관이 있음을 알 수가 있다.

누적 분포 함수 (cumulative distribution function, cdf)는 실제로 어떤 확률 변수가 특정 값보다 작거나 같을 확률을 기술해주는 함수이다. 즉,  $F_X(x)$ 를 연속 확률 변수  $X$ 의 누적 분포 함수라 하면,  $F_X(x) = P(X \leq x)$ 가 된다. 따라서 위에서 언급한 사실을 고려하면 쉽게 다음과 같은 관계를 얻을 수가 있다.

$$F_X(x) = \int_{-\infty}^x p_X(t) dt$$

이 식은 확률 변수의 확률 밀도 함수가 주어져 있을 때, 누적 분포 함수를 구하는 방법을 기술하고 있다. 예를 들어, 위에서 언급한 균등 분포와 지수 분포의 경우 누적 분포 함수는 각각 다음과 같다.

$$F_X(x) = \int_0^x 1 dt = x, \quad 0 \leq x \leq 1$$

$$F_X(x) = \int_0^x \lambda e^{-\lambda t} dt = 1 - e^{-\lambda x}, \quad x \geq 0$$

반면에, 고등학교 시절의 미적분 시간에 배운 바에 의하면 다음과 같은 관계도 유도할 수 있는데,

$$p_X(x) = \frac{dF_X}{dx}(x)$$

이는 누적 분포 함수의 순간 변화율, 즉  $x$ 가 순간적으로 증가함에 따라  $F_X(x)$ , 즉  $X$ 가  $x$ 보다 같거나 작을 확률이 증가하는 정도가 바로  $p_X(x)$ 라는 것을 의미한다.

이때 0과 1 사이의 값을 가지는 누적 분포 함수의 중요한 특징 중의 하나는 바로 이 함수가 단조 비감소 함수 (monotonically nondecreasing function)라는 점으로서, 다음과 같이 기술할 수가 있다.

만약  $x_0 \leq x_1$ 이라면,  $F_X(x_0) \leq F_X(x_1)$

아래의 그림 3은 주어진 확률 밀도 함수에 대한 누적 분포 함수로서 그 의미와 둘 간의 관계를 잘 생각해보자.

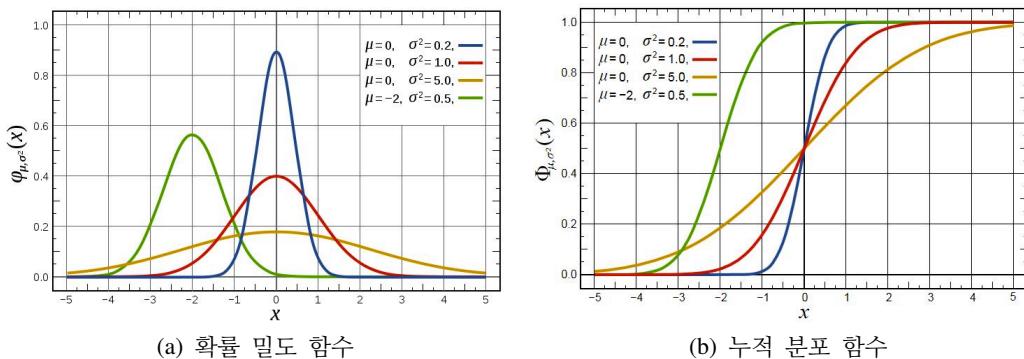


그림 3: 정규 분포 (normal distribution)의 정의 (출처: <http://en.wikipedia.org>).

## 2.1 Inversion 방법

우선 다음과 같은 문제를 생각하자.

사용자가 임의로 정의한 확률 밀도 함수  $p_X(x)$ 를 따르는 확률 변수  $X$ 를  
모사해주는 난수 수열  $X_0, X_1, X_2, X_3, \dots$ 를 발생시켜라.

여기서  $p_X(x)$ 를 따르는 난수란 이를 무한히 많이 발생시켜  $x$ 값에 대한 분포를 계산할 경우, 그 분포가  $p_X(x)$ 에 수렴함을 의미한다.

이제 컴퓨터를 사용하여 위 문제에서 요구하는 난수를 수치적으로 계산해주는 방법에 대하여 살펴보자. 특히 여기서는 inversion 방법이라는 기법을 사용할 것인데, 이 방법은 다음과 같은 사실에 기반을 두고 있다.

$U_0, U_1, U_2, U_3, \dots$ 을  $[0, 1]$  구간의 값을 가지는 균등 확률 변수에 대하여 생성한 난수 수열이라고 하자.  $F_X(x)$ 를 주어진 확률 밀도 함수  $p_X(x)$ 에 대한 누적 분포 함수라 하면, 이 확률 분포를 따르는 난수 수열 값  $X_i$  ( $i = 0, 1, 2, 3, \dots$ )는  $X_i = F_X^{-1}(U_i)$ 와 같이 구할 수 있다.

이러한 사실은 다음과 같은 관계로부터 확인할 수 있는데,

$$P(X_i \leq x) = P(F_X^{-1}(U_i) \leq x) = P(U_i \leq F_X(x)) = F_X(x)$$

이를 다르게 표현하면 주어진 균등 확률 변수 값  $U_i$ 에 대해  $U_i = F_X^{-1}(X_i)$ 를 만족시켜주는  $X_i$ 를 찾으면 된다는 것을 의미한다.

원하는 난수를 생성하기 위하여 주어진 누적 분포 함수  $u = F_X(x)$ 의 역함수  $x = F_X^{-1}(u)$ 를 구해야 한다. 예를 들어, 위에서 기술한 지수 분포를 따르는 난수  $X_i$ 를 생각하면,  $U_i = F_X(X_i) = 1 - e^{-\lambda X_i}$  관계에서  $X_i = -\frac{\ln(1-U_i)}{\lambda}$ 와 같이 원하는 난수를 계산할 수 있게 된다. 즉 0과 1 사이의 균등 확률 변수를 반복적으로 발생시켜 위에서와 같이  $X_i$ 를 계산하면 지수 분포를 따르는 확률 사건을 발생시킬 수 있으며, 이러한 수열 데이터를 원하는 확률 과정을 시뮬레이션하는데 사용하면 된다.

다만 잘 알다시피 일반적으로 복잡한 형태의 누적 분포 함수  $u = F_X(x)$ 에 대한 역함수  $x = F^{-1}(u)$ 의 수식을 찾는 것이 매우 어렵거나 불가능하기 때문에, 이 역함수 문제를 수치적으로 (numerically) 해결해야 하는데, 바로 이 과정에서 지난 추에 배운 비선형 방정식의 근을 구하는 방법을 적용하게 된다.

### 3 수치 적분 문제의 정의 및 합성 사다리꼴 공식

실습에 들어가기 전에 한 가지 더 알아야 할 것이 있는데, 바로 누적 분포 함수값  $F_X(x)$ 의 계산에 관한 것이다. 앞에서 기술한 바와 같이 이 값은  $F_X(x) = \int_{-\infty}^x p_X(t) dt$ 와 같이 확률 밀도 함수  $p_X(x)$ 를 적분한 값이다.  $p_X(x)$ 의 형태가 단순할 경우 고등학교에서 배운 적분 기법을 사용하여  $F_X(x)$ 에 대한 수식을 얻을 수 있으나, 그렇지 못할 경우에는 컴퓨터를 사용하여 수치적인 방법 (numerical method)을 사용하여 이 값을 계산해주어야 한다.

#### 3.1 수치 적분 문제 정의

주어진 구간  $[a,b]$ 에 대해 임의의 함수  $y = f(x)$ 를 적분하는 문제, 즉  $\int_a^b f(x) dx$  값을 구하는 문제를 고려하자. 일반적으로 수치 적분 방법 (numerical integration method) 들은  $y = f(x)$  함수를  $[a,b]$  구간에서 유한개 지점에 대하여 샘플링 (sampling)한 후, 그러한 샘플 데이터를 사용하여 적분값을 근사적으로 계산하는 방식을 취한다.

다음 테이블은 함수  $y = f(x)$ 를 다음과 같이  $[a,b]$  구간 안에 존재하는  $n+1$ 개의 지점에 대해 함수값을 샘플링한 데이터를 기술하고 있다.

$x$	$x_0$	$x_1$	$x_2$	$\dots$	$x_n$
$y$	$y_0$	$y_1$	$y_2$	$\dots$	$y_n$

편의상,<sup>2)</sup> 여기서는 구간  $[a, b]$ 를  $n$ 개의 균등한 구간으로 나눈 후, 각 구간의 경계점에 해당하는  $n+1$ 개의  $x$  지점에 대해  $f(x)$  값을 구한 샘플 데이터라고 가정하자. 이 경우 다음과 같은 조건을 만족한다 (그림 4 참조).

$$\Delta h = \frac{b-a}{n}, \quad x_i = a + i \cdot \Delta h \quad (i = 0, 1, 2, \dots, n), \quad y_i = f(x_i) \quad (i = 0, 1, 2, \dots, n)$$

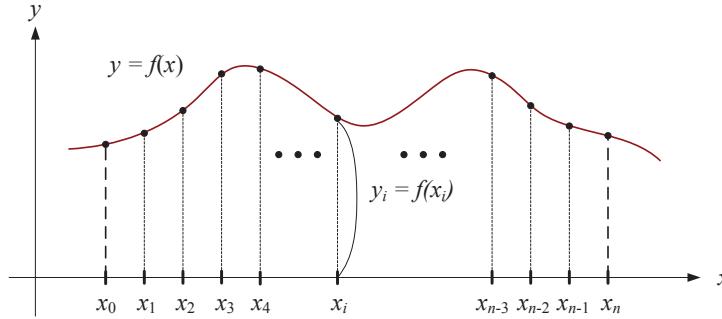


그림 4: 수치 적분을 위한 함수의 균등 샘플링

일반적인 수치 적분 방법들은 이러한 샘플 데이터로부터 다음과 같이 함수값에 대한 일종의 가중 평균과 같은 방식으로 적분값에 대한 근사값을 계산한다.

$$\int_a^b f(x) dx \approx \sum_{i=0}^n w_i \cdot y_i$$

이때 함수값  $y_i$ 에 곱해주는 가중치  $w_i$ 는 사용하고자 하는 수치 적분 방법에 따라 다른데, 재미있는 사실은 어떤 가중치 값을 사용하는가는 추정한 적분 근사값의 정확도에 크게 영향을 미친다는 점이다.

### 3.2 합성 사다리꼴 공식

본 실습에서는 여러 수치 적분 방법 중 합성 사다리꼴 공식 (composite trapezoidal rule)<sup>3)</sup>이라는 방법을 사용하자. 이 방법은 다음과 같은 공식을 사용한다.

$$\int_a^b f(x) dx \approx \frac{h}{2} \left\{ f(x_0) + 2 \sum_{i=1}^{n-1} f(x_i) + f(x_n) \right\}$$

즉 가중치 관점에서 보면,  $w_0 = w_n = \frac{h}{2}$ <sup>3)</sup>이고, 나머지에 대해서는  $w_i = h$ 인 값을 사용하고 있다. 이 방법의 이론적인 오차는  $E = -\frac{f''(\xi)(x_n-x_0)}{12}h^2$ ,  $\xi \in [x_0, x_n]$ 이 됨을 보일 수 있는데,

2) 반드시 그럴 필요는 없지만,

이에 대해서는 일단 넘어가도록 하자. 대신 왜 이 방법의 이름에 사다리꼴이라는 용어가 쓰였는지, 그림 5을 보면서 위 공식의 의미를 곰곰히 생각해보자.

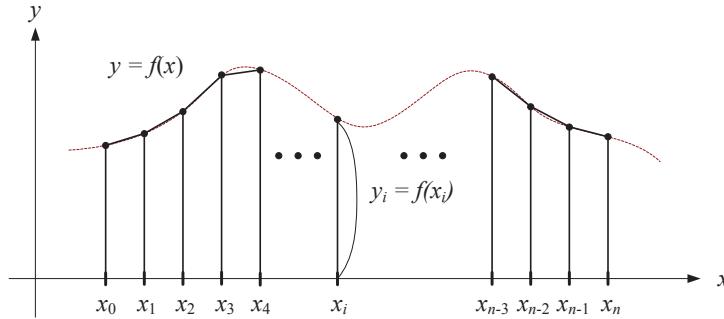


그림 5: 합성 사다리꼴 공식의 적용 결과

실습으로 넘어가지 전에 한 가지 더 생각을 해야할 것이 있다. 위에서는  $[a, b]$ , 즉  $[x_0, x_n]$  전체 구간에 대하여 수치 적분을 하는 방법을 기술하였는데, 본 실습의 문제를 해결하고자 할 때,  $[a, b]$  구간에 존재하는 임의의  $x$ 에 대해  $F_X(x) = \int_{x_0}^x p_X(t) dt$  값을 반복적으로 계산해주어야 한다. 이때 일반적으로 적분 구간의 오른쪽 경계  $x$ 가 샘플링 한 지점이 아닌데, 본 실습에서는  $x_0$ 에서  $x$  직전의 샘플링 지점, 예를 들어,  $x_m$ 까지의 적분값  $\int_{x_0}^{x_m} p_X(t) dt$ 은 합성 사다리꼴 공식을 사용하여 구하고, 나머지 구간에 대한 적분값  $\int_{x_m}^x p_X(t) dt \stackrel{?}{=} \frac{x-x_m}{x_{m+1}-x_m} \cdot \frac{\Delta h}{2} (p_{X_m} + p_{X_{m+1}})^3$ 과 같이 구한 후, 두 값을 더하여  $F_X(x)$ 에 대한 근사값을 계산하자.

3) 오류 수정:  $(p_{X_m} + \frac{p_{X_{m+1}} - p_{X_m}}{x_{m+1} - x_m} \cdot \frac{x - x_m}{2})(x - x_m)$

## 4 실습 문제

이번 실습 시간과 숙제를 통하여 제작할 소프트웨어의 요구 사항은 다음과 같이 세 가지 문제로 요약된다.

문제 I: 본 실습에서 제공하는 GUI 소프트웨어를 사용하여 임의의 확률 밀도 함수  $p_X(x)$ 를 설계하고,

문제 II:  $p_X(x)$ 를 따르는 확률 변수  $X$ 를 시뮬레이션 해주는 난수 수열  $X_0, X_1, X_2, X_3, \dots$ 를 확률적으로 올바른 방법으로 발생시킨 후,

문제 III: 자신이 올바르게 난수를 생성하였는지 통계적으로 확인하라.

### 4.1 문제 I: 확률 밀도 함수 $p_X(x)$ 의 설계

#### 확률 밀도 함수 형태의 곡선 설계

아래의 그림 6은 본 실습에서 제공하는 GUI 소프트웨어를 사용하여 설계한 임의의 곡선을 도시하고 있다 (사용법은 조교가 설명). 여러분은 자신이 사용하고자 하는 확률 밀도 함수의 형태를 가지는 곡선을 가급적 많은 수의 점을 사용하여 부드럽게 설계한 후, 이를 100개 이상의 지점에서 샘플링을 하여 파일에 저장하도록 하라.

이 소프트웨어는 샘플링한 곡선 데이터를 이름이 `sampling_table.txt`인 ASCII 파일에 다음과 같은 형식으로 저장한다.

```

100 5.676768
20.000000 9.000000
25.676767 11.591169
31.353535 15.391957
37.030304 20.120052
42.707069 25.132595
:
564.969727 47.323036
570.646484 42.336571
576.323242 36.826424
582.000000 32.999992

```

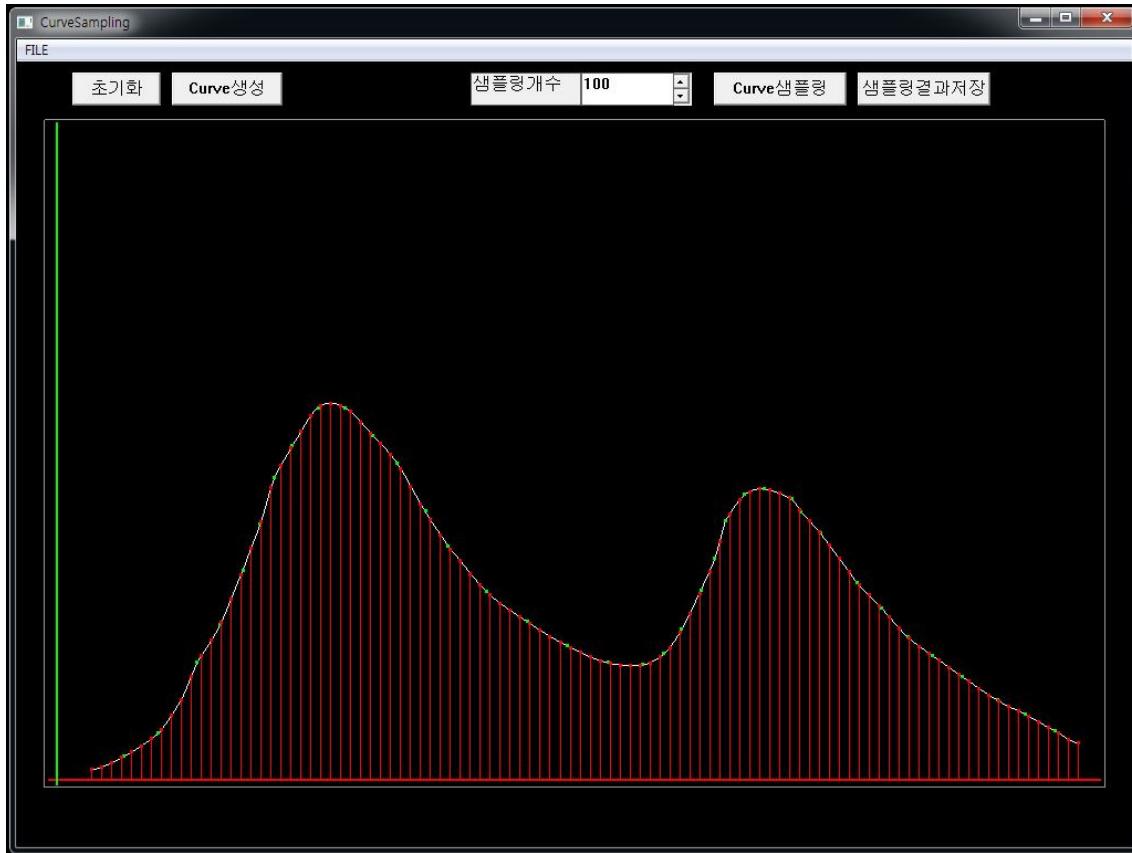


그림 6: GUI 소프트웨어를 사용한 곡선 설계 예

이 예에서는 양 끝점 포함 100개의 지점에서 곡선을 샘플링 한 경우인데 (즉  $n = 99$ ), 이 곡선에 대한 함수가  $y = f(x)$ 라 할 때 각 숫자의 의미는 다음과 같다.

- 100: 샘플링 지점의 개수  $n + 1$
- 5.676768: 샘플링 지점 간의 거리  $\Delta h = \frac{x_n - x_0}{n}$
- 20.000000, 25.676767, 31.353535, …, 582.000000: 각 샘플링 지점의  $x$  좌표  $x_i, i = 0, 1, 2, \dots, n$
- 9.000000, 11.591169, 15.391957, …, 32.999992: 각 샘플링 지점  $x_i$ 에서의 함수 값  $y_i = f(x_i), i = 0, 1, 2, \dots, n$

즉 여러분이 개념적으로 연속 공간인 실수 공간에서 설계한 곡선  $y = f(x)$ 를  $[x_0, x_n]$  구간에 대해 이산적으로 양 끝점 포함  $n + 1$ 개 샘플링한 다음과 같은 데이터를 출발점으로 본 실습 문제를 해결해보자.

$x$	$x_0$	$x_1$	$x_2$	…	$x_n$
$y$	$y_0$	$y_1$	$y_2$	…	$y_n$

### 확률 밀도 함수 형태의 곡선 설계

앞에서 설명한 GUI 소프트웨어를 사용하면 손쉽게 원하는 형태의 곡선을 설계할 수 있으나, 이를 바로 확률 밀도 함수로 사용하는데에는 문제가 있다. 앞에서 설명한 확률 밀도 함수의 기본 요건을 다시 한번 생각해보자.

$$(a) \text{ 모든 실수 } x \text{에 대해, } p_X(x) \geq 0$$

$$(b) \int_{-\infty}^{\infty} p_X(x) dx = 1$$

여러분이 설계한 곡선은 현재  $[x_0, x_n]$  구간에서 0보다 같거나 커야하는 첫 번째 조건을 만족하고 있다. 하지만, 전체 구간에서 여러분이 설계한 곡선  $y = f(x)$ 를 적분할 경우 1보다 큰 값이 나오기 때문에 두 번째 조건을 만족시키고 있지 못하고 있다. 따라서 이 곡선을 위 구간에 대해 정규화(normalization) 해주어야 하는데, 다음과 같은 계산을 통하여 우리가 원하는 확률 밀도 함수를 만들 수 있다.

$$p_X(x) = \frac{f(x)}{\int_{x_0}^{x_n} f(x) dx}, \quad x \in [x_0, x_n]$$

#### 실습 문제 2-1

`sampling_table.txt` 파일에 저장되어 있는 함수  $y = f(x)$ 의 샘플 데이터를 입력 받아, 그에 해당하는 확률 밀도 함수  $p_X(x)$ 에 대한 샘플 데이터를 동일한 형식으로 ASCII 파일 `pdf_table.txt`에 저장해주는 콘솔 프로그램 (프로그램 2-1)을 작성하라.

$x$	$x_0 = 0.0$	$x_1$	$x_2$	$\dots$	$x_n = 1.0$
$p_X(x)$	$p_{X_0}$	$p_{X_1}$	$p_{X_2}$	$\dots$	$p_{X_n}$

- 확률 밀도 함수를 구축할 때 `sampling.txt` 파일에 저장되어 있는 함수의 독립 변수, 즉  $x$ 의 구간을 0과 1 사이의 값으로 정규화 하라. 즉, 예를 들어, 앞에서의 예에서 정의되어 있는  $x \in [20.0, 582.0]$ 의 구간이  $x \in [0.0, 1.0]$  구간으로 선형적으로 매팅이 되도록 변환 후 `pdf_table.txt`에 저장하라.
- (프로그램 2-1)은 위의 파일에 확률 밀도 데이터를 저장한 후, 다시 이 데이터를 사용하여  $[x_0, x_n] = [0.0, 1.0]$  구간에서 여러분이 이산적으로 모델링 한 확률 밀도 함수  $p_X(x)$ 를 다시 수치적으로 적분하여 그 값이 1이 되는지, 아니면 최소한 1에 아주 가까운 값이 되는지를 확인해주는 내용을 다음과 같은 형식으로 콘솔 윈도우에 출력해주어야 한다. 또한  $[x_0, x_n] = [0.0, 0.25]$ ,

$[x_0, x_n] = [0.25, 0.5]$ ,  $[x_0, x_n] = [0.5, 0.75]$  그리고  $[x_0, x_n] = [0.75, 1.0]$  등 각 구간에 대하여 동일한 작업을 반복한 후 자신이 설계한 곡선의 내용과 직관적으로 일치하는지 확인하라.

```
*** Integrating the pdf from 0.0 to 1.0 = 0.999987
*** Integrating the pdf from 0.0 to 0.25 = 0.273152
*** Integrating the pdf from 0.25 to 0.5 = 0.214839
*** Integrating the pdf from 0.5 to 0.75 = 0.371293
*** Integrating the pdf from 0.75 to 1.0 = 0.123848
```

[참고] 부동 소수점 숫자의 저장 및 연산은 모두 `double` 타입의 변수 및 연산을 사용하라.

## 4.2 문제 II: 난수 생성

위의 과정을 통하여 우리가 원하는 확률 밀도 함수를 이산적으로 모델링하여 pdf\_table.txt 파일에 저장을 해놓았는데, 이제 이에 기반을 두어 난수 수열을 생성하여 보자. 이를 위해서 우선  $[0, 1]$  구간에 대한 균등 확률 변수를 여러분의 C 프로그램에서 생성해주어야 하는데, 엄밀히 말해서 수학적으로는 순수한 난수는 아니지만, 충분히 난수처럼 보이는 유사 난수 (pseudo-random number)는 다음과 같은 방법을 기반으로 생성할 수 있다 (자세한 내용은 조교가 설명할 예정이며 적절히 변형할 것).

```
#include <stdio.h>
#include <stdlib.h> // #define RAND_MAX ???
// int rand(void);
// void srand(unsigned int seed);
#include <time.h> // For the time function

void my_func(int m) {
    int i = 0, irand;
    unsigned int iseed = (unsigned int) time(NULL);
    srand(iseed);
    while (i < m) {
        :
        irand = rand();
        :
    }
}
```

또는 여러분 프로그램에 직접 다음과 같은 코드를 삽입하여 변형 후 사용해도 된다.

```
static unsigned long int next = ???; // ??? = a positive integer

#define MY_RAND_MAX 32767
int rand(void) {
    next = next * 1103515245 + 12345;
    return (unsigned int) (next/65536) % (MY_RAND_MAX+1);
}
```

```

void srand(unsigned int seed) {
    next = seed;
}

```

이제 앞에서 설명한 inversion 방법 설명 시 기술한 바와 같이,  $[0, 1]$  구간에 대한 균등 확률 변수 수열  $U_0, U_1, U_2, U_3, \dots$ 를 하나씩 생성하면서 우리가 원하는 난수  $X_i$ 를  $X_i = F_X^{-1}(U_i)$ 와 같이 계산하여 발생시키면 된다. 여기서 다시 기억을 되살려보면  $F_X(x)$  함수는 누적 분포 함수로 본 실습의 문제 환경에서는  $F_X(x) = \int_{x_0}^x p_X(t) dt$ 와 같이 정의된 함수이다.

여기서 문제는  $p_X(x)$  함수가 수학적으로 명확한 공식으로 주어진 것이 아니라 이산적으로 샘플링한 정보만 가지고 있는 상황이기 때문에, 다음과 같은 기본 문제를 수치적으로 (numerically), 즉 프로시저를 사용하여 (procedurally) 해결하여  $X_i$  값을 계산해주어야 한다.

- (a) 임의의  $x$ 값에 대하여  $F_X(x)$ 을 구한다.
- (b) 임의의  $u$ 값에 대하여  $F_X(x)$ 의 역함수 값  $x = F^{-1}(u)$ 을 구한다.

첫 번째 문제를 어떻게 처리할 지에 대해서는 앞에서 설명하였다. 문제는 두 번째 요구 사항인데, 수식도 모르면서 수치적으로 밖에 계산할 수 없는 함수  $u = F_X(x)$ 의 역함수를 계산해주어야 한다는 점이다. 일면 불가능해보이는 것 같기도 하지만, 이 문제는 지난 주에 학습한 비선형 방정식의 풀이 방법을 사용하면 쉽게 해결할 수가 있다. 즉, 구간  $[0, 1]$ 에 존재하는 임의의 값  $U_i$ 에 대하여, 구간  $[x_0, x_n]$ 에 존재하는 것이 보장되는  $X_i = F_X^{-1}(U_i)$  값을, 새로운 함수  $f(x)$ 를 다음과 같이 정의하면,

$$f(x) = F_X(x) - U_i, \quad x \in [x_0, x_n]$$

역함수를 구하는 문제는 바로 비선형 방정식  $f(x) = 0$ 의 근을 구하는 문제로 귀결된다. 한 가지 명심해야 할 점은 앞에서 언급한 바와 같이  $[x_0, x_n]$  구간에서  $F_X(x)$ 는 0과 1 사이의 값을 가지며, 또한 단조 비감소라는 사실이다. 따라서 0과 1 사이의 값  $U_i$ 에 대해  $[x_0, x_n]$  구간에서  $f(x)$ 는  $-U_i$  (음수)와  $1 - U_i$  (양수) 사이의 값을 가지며 역시 단조 비감소이므로 유일한 (unique) 근을 갖게 된다 (최소한 개념적으로는 그렇지만 실제로 수치적으로 문제를 풀 때에는 여러 예상치 못한 문제가 발생할 수 있음을 명심할 것). 본 실습에서는 지난 주에 자신이 숙제로 작성한 Bisection 방법을 사용하여 이 문제를 풀어보자. 이때 초기 구간은  $[x_0, x_n]$ 으로 하면 됨을 명심하라.

**[참고]** 앞에서 잠깐 언급한 바와 같이 이 문제를 풀다보면 예상치 못한 수치 문제가 발생할 수 있으니, 이러한 문제를 신중하게 고려하라. 예를 들어, 만약 우연히도  $U_i$ 가 1이라면 단순히 코딩을 할 경우 프로그램 수행 시 문제가 발생할 소지가 높다. 이를 위하여  $U_i \in (0, 1)$ 인

$U_i$ 를 생성하는 것도 한 가지 방법이다.

### 실습 문제 2-2

pdf\_table.txt 파일에 저장되어 있는 확률 밀도 함수  $p_X(x)$ 에 대한 샘플 데이터를 입력 받아,

$x$	$x_0 = 0.0$	$x_1$	$x_2$	$\cdots$	$x_n = 1.0$
$p_X(x)$	$p_{X_0}$	$p_{X_1}$	$p_{X_2}$	$\cdots$	$p_{X_n}$

콘솔 윈도우에서 생성할 난수 개수  $n_r$ 을 입력 받아, 입력 확률 밀도 함수를 시뮬레이션 해주는  $n_r$ 개의 난수  $X_0, X_1, X_2, \dots, X_{n_r-1}$ 를 발생시켜, 그 결과를 이름이 random\_event\_table.txt인 ASCII 파일에 저장해주는 콘솔 프로그램 (프로그램 2-2)를 작성하라. 이 결과 파일에는 각 줄에 한 개씩 다음과 같은 내용의 계산 결과를 출력하도록 하라.

$n_r$

$X_0$

$X_1$

$X_2$

$\vdots$

$X_{n_r-1}$

[참고] 부동 소수점 숫자를 다룰 때에는 모두 double 타입의 변수 및 연산을 사용하고, 파일에 저장 시 유효숫자 15자리까지 출력할 것.

## 5 숙제

제출 마감: ?월 ?일 오후 ?시 정각

제출물 및 방법: 조교가 실습 시간에 공지

이번 주의 숙제는 다음과 같다.

### 숙제 2-1

지금 개발 중인 컴퓨팅 시스템의 성능 평가를 위한 시스템 모델링 작업을 수행 하려 한다. 이 시스템을 구성하는 한 컴포넌트가 작동중 고장이 나는 상황을 통 계적으로 모델링하여 보자. 시스템 가동 시점부터 이 컴포넌트가 처음 고장이 날 때까지의 시간  $X (\geq 0)$ 를 강의 시간에 설명한 지수 분포로 모사하려 한다.

- (i) 지수 분포의 인자  $\lambda (> 0)$ 에 대해 확률 변수  $X$ 의 기대값이  $E[X] = \frac{1}{\lambda}$ 이고 분산이  $Var[X] = \frac{1}{\lambda^2}$ 이라는 사실을 고려하여, 세 가지 서로 다른  $\lambda$  값을 설정하라.
- (ii) 자신이 정한  $\lambda$ 에 해당하는 지수 분포 각각에 대하여, inversion 방법을 사용하여 자신이 정한 충분히 큰 개수만큼 이 지수 분포를 따르는 난수를 생성하라.
- (iii) 자신이 생성한 난수들을 사용하여 평균값과 분산값을 구한 후 이론적인 값들과 얼마나 일치하는지 분석하라.

[참고] 부동 소수점 숫자의 저장 및 연산은 모두 `double` 타입의 변수 및 연산을 사용하라. 과연 얼마나 많은 난수를 생성해야만 통계적으로 구한 평균값과 분산값이 이론적인 평균값과 분산값과 충분히 일치하게 되는지 명확히 분석하여 보고서에 기술하라.

**숙제 2-2**

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)).

숙제 2-2(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.

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 2-2(iv) 충분히 큰 난수의 개수  $n_r$ 에 대해 프로그램 2-2(a)와 프로그램 2-2(b)를 수행시킨 후, 각 방법이 난수를 생성하는데 걸린 시간을 비교하라.

**[참고]** 부동 소수점 숫자의 저장 및 연산은 모두 `double` 타입의 변수 및 연산을 사용하라.

**5.1 숙제 2-2(ii) 주의 사항 (프로그램 2-3 요구 사항 포함)**

자신의 프로그램이 올바르게 작동한다는 사실을 증명하는 것은 자신의 책임이므로, 논리적으로 합당한 방법을 사용하여, 예를 들어, 통계적 처리 및 그래프 등을 사용하거나 기타 방법을 사용하여, 보고서를 읽는 사람이 프로그램의 유효성에 대한 확신을 갖을 수 있도록 효과적으로 보고서를 작성하라.

이때 최소한 다음과 같은 내용을 포함하라.

- (가) 자신이 출력한 `pdf_table.txt`와 `random_event_table.txt` 파일에서 확률 밀도 함수와 생성한 난수 정보를 읽어들인다.
- (나)  $[x_0, x_n] = [0.0, 1.0]$  구간을 적절히 많은 개수의 구간으로 나눈 후, 각 구간에서 몇 개의 난수가 생성이 되었는지 세어 히스토그램 데이터를 구축하라.
- (다) 자신의 상상력을 동원하여 이 히스토그램 정보로 부터 난수가 올바르게 생성되었다는 사실을 증명하기 위한 정보를 이름이 `histogram.txt`인 파일에 저장한 후, 이를 활용하여 보고서를 작성하라.

**5.2 숙제 2-2(iii) 주의 사항**

비선형 방정식의 근을 구하기 위하여 Newton 방법을 사용할 경우, 방정식의 함수  $f(x)$ 에 대한 함수값뿐만 아니라 미분값  $f'(x)$ 도 계산을 할 수 있어야 한다. 먼저 함수값  $f(x) = F_X(x) - U_i$ 은 앞에서 설명한 수치 방법을 통하여 계산할 수 있다.

또한 미분값의 경우 다음과 같은 사실을 고려하면,

$$f'(x) = \frac{df(x)}{dx} = \frac{dF_X(x)}{dx} = p_X(x)$$

$[x_0, x_n]$  구간의 임의의  $x$ 에 대한 미분값  $f'(x)$ , 즉  $p_X(x)$ 는 이산적으로 기술되어 있는 확률 밀도 함수 정보로부터 추정할 수 있다. 이를 위하여 여러 수치적인 방법이 존재하지만, 가장 쉬운 방법은 선형 보간 (linear interpolation)을 적용하는 것으로서, 그 공식은 다음과 같으니 그 의미를 생각해보자.

$x \in [x_i, x_{i+1}]$  일 때,  $s = \frac{x-x_i}{x_{i+1}-x_i}$ 라고 하자. 그러면,

$$f'(x) = p_X(x) \approx (1-s) \cdot P_{X_i} + s \cdot P_{X_{i+1}}$$

이렇게 미분값을 구하는 방법을 알 경우 Newton 방법을 적용할 수 있는데, 초기값으로  $[x_0, x_n]$  구간 안에서 적절한  $x$ 값을 선택하여 사용하라. 초기값이 방정식의 근에 가까울 수록 수렴하는 속도, 즉 반복문이 수행되는 회수가 감소한다는 점을 명심하고 수행 속도가 빠른 프로그램을 작성하기 위하여 어떠한 방식으로 초기값을 정할지 생각해보고, 자신이 사용한 방법을 보고서에 상세히 기술하라.

**[참고]** 한 가지 가능한 방법은  $[x_0, x_n]$  구간을 초기 구간으로 하여 Bisection 방법을 몇 회 반복하여 구간의 폭을 줄인 후, 그 중점을 Newton 방법의 초기값을 사용하는 것인데, 이 방법이 실제로 비용, 즉 계산 시간을 줄이는데 효과적인지 확인하고, 그 결과를 보고서에 상세히 기술하라.

비선형 방정식의 근을 구하기 위하여 Secant 방법을 사용할 경우, 두 개의 초기값을 어떻게 구할지 잘 생각해보기 바란다. 한 가지 방법으로 위의 Newton 방법에서와 같이 Bisection 방법을 활용하는 것도 고려해볼 것.

### 5.3 숙제 2-2(iv) 주의 사항

이 항목의 주목적은 Bisection 방법과 Newton 또는 Secant 방법의 비용을 분석하는 것이다. 즉 각 방법을 특정 응용 문제에 적용할 때 고려할 수 있는 유용성/정확도/비용 등의 분석 요인 중 비용에 관한 내용을 살펴보는 것이다.

의미있는 비교 결과를 산출하기에 충분한 개수의 난수를 생성한 후, 시간 측정을 통하여 비용 분석을 하라. 시간을 측정하는 방법은 조교가 실습 시간에 설명할 예정이며, 프로그램 2-2(a)와 프로그램 2-2(b) 각 프로그램에서 입력 데이터를 읽어 들인 직후 및 생성 결과 출력 직전 지점에 시간을 측정하는 코드를 삽입하라 (즉 입출력 시간은 시간 측정에서 제외할 것).

여러 상황에 대하여 실습을 수행한 후, 의미있는 비용 비교 결과를 보고서에 기술하라. 물론 실습 방법의 적절성에 따라 보고서를 읽는 사람이 여러분의 분석 결과에 대한 신뢰 정도가 크게 달라질 수 있음을 명심하라.