# 몬테카를로 시뮬레이션을 이용한

## 자연상수 $e$(Napier's constant)값 추정

## 시뮬레이션(101) HW01

부경대학교 시스템경영공학부 산업경영공학전공
201730262 이상주

## 1. 결과요약

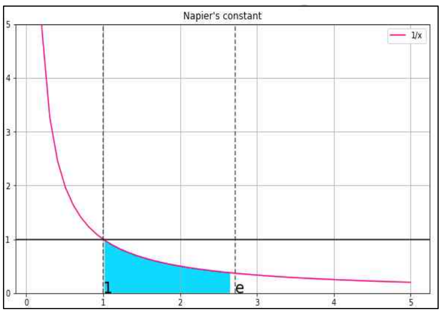
- 반복횟수: 10000번
- $\int_1^{?} \frac{1}{x}\, dx = 1$
- $e$ = 2.718281828...
- $\hat{e}$ = 2.7198

```
총반복횟수: 10000
자연상수 e 추정값: 2.7198
```

## 2. 서 론

몬테카를로 시뮬레이션을 이용해 자연상수 $e$값을 추정하는 과정으로 자연상수 $e$값(2.7182818...)을 모른다는 가정하에 자연상수 $e$는 곡선 y=1/x와 x축, x=1, x=a로 둘러싸인 영역의 넓이가 1이 되는 수 a>1를 자연상수 $e$라는 정의를 이용한다.

<자료1> 적분 범위를 표시한 그래프




## 3. 개발환경 및 수리적 모형

### 3.1. 개발환경(SW 환경)

- 언어: Python3
- 개발환경: Jupyter, colab

### 3.2. 수리적 모형

$$\int_a^b \frac{1}{x}\, dx = C(b-a)\frac{n}{w} = 1$$

- a = 1
- b = $e$
- c = y의 최대값(1)
- n = 1/x 곡선과 1~$e$ 범위 안에 타점된 점의 수
- w = 전체 타점된 점의 수(반복횟수)

## 4. 본 론

### 4.1. 초기 리스트 산출($e$값 후보 선정)

$e$값을 만족하는 조건은 $\int_1^e \frac{1}{x}\, dx = 1$ 이다.

따라서 1.001부터 5미만의 수를 0.001 단위씩

늘려가면서 반복문을 이용해 해당 수 모두 각각의 정적분을 진행 하고 반올림을 하여 그 값이 1과 같다는 조건을 만족하는 수를 미리 만들었던 빈 리스트인 'check_ok' 리스트에 담는다. 이렇게 담긴 check_ok 리스트에는 1.648,..,4.599까지 담겨 있으며 이 리스트 값은 $e$의 1차 후보가 된다.

<자료2> 초기 리스트 산출(1차 후보 선정)

```
########################################
# Monte Carlo integration_초기 리스트 산출#
########################################

import numpy as np
import math
from sympy import Integral, Symbol


check_list = np.arange(1, 5, 0.001)
x = Symbol('x')
f = 1/x

check_ok = []
for i in check_list:
    integral_value = Integral(f, (x, (1, i))).doit()
    if round(integral_value) == 1.0:
        check_ok.append(i)

print(check_ok)

[1.6489999999999285, 1.6499999999999284, 1.650999999999
```

## 4.2. 변수 및 기타 설정

<자료3> 변수 및 기타설정

```
########################################
# Monte Carlo integration_변수 및 기타설정#
########################################

import random as rd
import os


fun = lambda x:1/x
xA = 1. # x의 구간 시작
xB = 0. # x의 구간 끝
yA = 0. # f(x)의 최소값
yB = 1.0 # f(x)의 최대값보다 큰 임의의 값. 클수

rd.seed(os.times())

#기타변수
it = 10000 #반복횟수
count = 0
optimal_sum_counted = 0
```

초기 핵심 변수로는 1/x를 함수로 구현, x축의 xA(시작값) 1, xB(끝값)는 0으로 초기화 하고 f(x)의 최대값(yB)을 1, 최소값(yA)을 0으로 하는 변수를 선언한다. 기타 변수로는 반복횟수는 충분히 큰 10000으로 고정시키고 최적의 $e$값을 구하는데 사용되는 optimal_sum_counted, count 변수를 0으로 초기화와 동시에 선언한다.

## 4.3. 몬테카를로 알고리즘 및 연산과정

<자료4> 몬테카를로 알고리즘 및 연산과정

```
##############################
# Monte Carlo integration_연산#
##############################

for xB in check_ok:
    AREA = (xB-xA)*(yB-yA) # 구간으로 둘러싸인 영역의 넓이.
    counted = 0

    for i in range(it):
        if fun(rd.uniform(xA, xB)) >= rd.uniform(yA, yB): counted+=1

    monte_integral = AREA*(float(counted)/float(it))
    if round(monte_integral, 3) == 1.0:
        optimal_xB = round(xB, 4)
        optimal_sum_counted += counted
        count +=1

```

4.1.에서 만들었던 check_ok 리스트 안의 값들은 반복문을 이용해 순차적으로 순회하면서 $\hat{e}$(xB)에 대입 하여 구간을 둘러 싸고있는 영역의 넓이를 의미하는 변수인 AREA 값을 각각 구한다. 두 번째 루프는 반복횟수(10000)만큼 순회하면서 랜덤으로 뽑은 xA와 xB 사이의 값을 함수에 대입한 값과 랜덤으로 뽑은 yA와 yB 사이값을 비교한다. 즉, $y_0 \le f(x_B)$의 조건을 만족한다면 미리 선언했던 counted 변수의 값에 1을 더해줌으로 함수와 범위 내의 타점의 개수를 기록한다.

$C(b-a)\dfrac{n}{w}$를 이용해 $e$후보의 적분 값(면적)을 monte_integral 변수에 담아 그 값이 소수점 셋째자리까지 반올림 했을 때 1과 같다면 최종 후보로 인정하며 1개 이상의 최종 후보 값들의 함수와 범위 내 총 타점 값을 모두 더해 optimal_sum_counted 변수에 담고 총 몇 개의 최종 후보가 있는지 미리 선언했던 count 변수에 기록한다.

### 4.4. 결과 출력

위 선언 했던 optimal_sum_counted 변수와 count 변수를 이용해 최종 후보들의 평균 타점 값을 구해 $\hat{e}$의 신뢰도를 높이고 optimal_counted 변수에 담는다.

$e = \frac{w}{n} + 1$을 이용해 최종 최적의 $e$값을 의미하는 euler 변수에 값을 담아 출력한다.

<자료5> 결과 출력

```
#####################################
# Monte Carlo integration_결과 출력#
#####################################

optimal_counted = optimal_sum_counted / count
euler = (it/optimal_counted) + 1 # 자연상수 e 값 = (총반복횟수/함수 안의 총타점) + 1

print("총반복횟수:", it) # 총반복 횟수
print("자연상수 e 추정값:", round(euler, 4)) # 자연상수 e값 추정
```

```
총반복횟수: 10000
자연상수 e 추정값: 2.7198
```

## 5. 결 론

난수를 이용한 몬테카를로 시뮬레이션을 이용해 $\hat{e}$ 값을 구한 결과 2.7189 값을 확인할 수 있었으며 이는

- $e$ = 2.71828... ≃ $\hat{e}$ = 2.7198

라는 결과를 도출할 수 있다.

### 5.1. 추가 실행 후 도출 값

(1)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7203
```

(2)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7244
```

(3)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7239
```

(4)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7213
```

(5)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7198
```

(6)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7212
```

(7)

```
총반복횟수: 10000
자연상수 e 추정값: 2.7211
```