02 적분

기본 개념 적분은 어디에서 시작하는가

적분이라는 단어에서 '쌓아 올리는 것'을 뜻함을 알 수 있다. 하지만 적분의 시작은 우리가 흔히 생각하는 넓이를 구하는 방법으로서가 아닌, 단순히 어떤 함수의 부정적분, 또는 원시함수를 구하는 것에서부터 시작한다. 미분의 역연산이라고 할 수 있는 부정적분에서 시작하면 넓이, 그리고 미분과도 관련 있는 정적분까지 이해할 수 있다.

02-1. 부정적분의 정의

함수 $F(x)$가 $f(x)$의 부정적분 $\Leftrightarrow$ $F'(x)=f(x)$ $\Leftrightarrow$ $F(x)$의 도함수가 $f(x)$

ex) $f(x)=2x+5$

$F(x)=x^2+5x$ 라고만 생각했다면 다시 생각해보자. 가령 $F(x)=x^2+5x+17$ 이나 $F(x)=x^2+5x+\pi$ 등에 대해서도 $F'(x)=2x+5=f(x)$ 이므로 모두 맞다. 즉 일반적으로 $f(x)=2x+5$ 의 부정적분은 $F(x)=x^2+5x+C$ (C는 상수)의 형태로 나타낼 수 있다. 이를 다음과 같이 나타낸다.

부정적분과 미분의 관계

$$\int f(x)dx=F(x)+C$$

이때 $\int$를 $integral$, $f(x)$를 피적분함수, x를 적분변수라 하고 C 는 임의의 적분변수이다.

추가 여러 가지 함수의 부정적분

피적분함수	부정적분	피적분함수	부정적분
삼각함수		$y=x^r$ (r은 실수) 형태	
$\cos x$		x	
$\sin x$		x^2	
$\sec^2 x$		x^3	
$\csc x \cot x$		x^r ($r \neq -1$)	
$\sec x \tan x$		$\frac{1}{x}$	
$\csc^2 x$			
지수함수			
e^x			
a^x ($a>0$, $a \neq 1$)			

추가 부분적분법과 치환적분법

부분적분법과 치환적분법은 적분을 계산하는 방법이다. 한 번에, 또는 쉽게 피적분함수의 부정적분을 구하지 못할 때 피적분함수의 일부를 치환하거나, 피적분함수를 두 함수의 곱으로 생각하고 곱의 미분법의 역연산으로 생각하여 부정적분을 구할 수 있다. 물론 이것이 모든 피적분함수의 부정적분을 구할 수 있게 하지는 않는다.

-치환적분법

미분가능한 함수 g에 대하여 $x=g(t)$로 놓으면

$$\int f(x)dx=\int f(g(t))g'(t)dt \qquad \left(\frac{dx}{dt}=g'(t),\ dx=g'(t)dt\right)$$

ex) $\int x\sqrt{x^2+1}\,dx=\int \frac{1}{2}\sqrt{t}\,dt=\frac{1}{3}t^{\frac{3}{2}}+C=\frac{1}{3}(x^2+1)^{\frac{3}{2}}+C$

$(\because x^2+1=t,\ 2xdx=dt)$

-부분적분법

미분가능한 두 함수 f, g에 대하여

$$\{f(x)g(x)\}'=f'(x)g(x)=f(x)g'(x)$$

$$\int \{f'(x)g(x)+f(x)g'(x)\}dx=f(x)g(x)+C$$

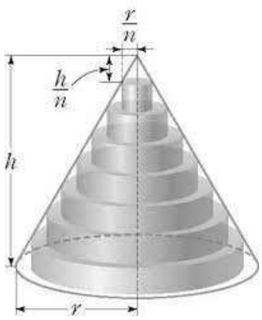
$$\therefore \int f(x)g'(x)dx=f(x)g(x)-\int f'(x)g(x)dx$$

ex) $\int x\sin x dx=x(-\cos x)-\int(-\cos x)dx=-x\cos x+\sin x+C$

02-2. 구분구적법

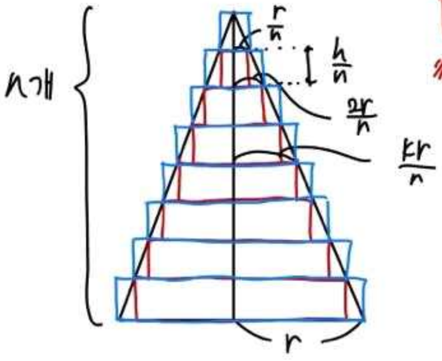
기본 개념 구분구적법

여러분이 원뿔의 부피를 구하는 공식($V=\frac{1}{3}\pi r^2 h$)을 모른다면 여러분은 그것을 어떻게 구하겠는가? 한 가지 방법은 원뿔을 밑면에 평행한 수많은 원기둥으로 나누고 그 부피를 모두 합하는 것이다. (물론 원뿔의 부피를 구하는 공식을 배우기 전에 구분구적법을 알고 있는 경우는 거의 없다.) 이렇게 주어진 도형을 쉽게 파악하고 있는 도형으로 나누어 이 도형들의 넓이 또는 부피의 합의 극한값으로 주어진 도형의 넓이 또는 부피를 구하는 방법을 구분구적법이라고 한다.



이것이 시뮬레이션의 핵심이 될 수도 있다. 자, 원뿔의 부피를 한 번 구해보자.

(다음장으로)



모든 자연수 n에 대하여
색 원기둥 부피의 합: S_n

$$S_n = \sum_{k=0}^{n-1} \pi\left(\frac{kr}{n}\right)^2 \cdot \frac{h}{n}$$
$$= \frac{\pi r^2 h}{n^3}\sum_{k=0}^{n-1} k^2$$
$$= \frac{\pi r^2 h}{n^3}\cdot\frac{(n-1)n(2n-1)}{6}$$
$$= \frac{\pi r^2 h\,(n-1)(2n-1)}{6n^2}$$

큰 원기둥 부피

큰 원기둥 부피 중 작은 원기둥 부피의 합이 차지하는 비율

$\left(\lim_{n\to\infty}\frac{(n-1)(2n-1)}{6n^2} = \frac{1}{3}\right)$

모든 자연수 n에 대하여
색 원기둥 부피의 합: T_n

$$T_n = \sum_{k=1}^{n} \pi\left(\frac{kr}{n}\right)^2 \cdot \frac{h}{n}$$
$$= \frac{\pi r^2 h}{n^3}\sum_{k=1}^{n} k^2$$
$$= \frac{\pi r^2 h}{n^3}\cdot\frac{n(n+1)(2n+1)}{6}$$
$$= \frac{\pi r^2 h\,(n+1)(2n+1)}{6n^2}$$

큰 원기둥 부피

큰 원기둥 부피 중 작은 원기둥 부피의 합이 차지하는 비율

$\left(\lim_{n\to\infty}\frac{(n+1)(2n+1)}{6n^2} = \frac{1}{3}\right)$

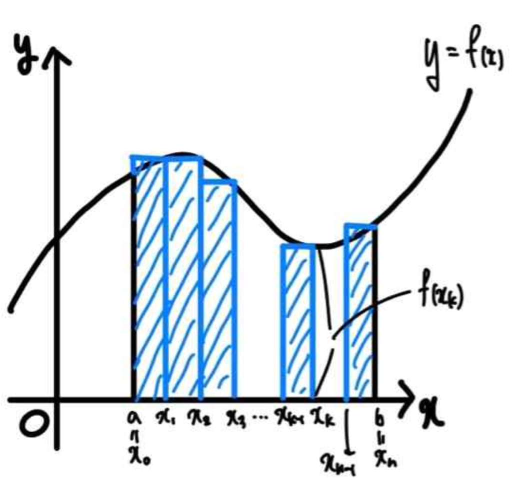
구하고자 하는 원뿔의 부피를 V라 하자.

$$S_n < V < T_n$$

$$\pi r^2 h\cdot\frac{1}{3} = \lim_{n\to\infty} S_n \le \underbrace{\lim_{n\to\infty} V}_{=V} \le \lim_{n\to\infty} T_n = \pi r^2 h\cdot\frac{1}{3}$$

$$\therefore V = \frac{1}{3}\pi r^2 h$$

(모든 부분을 이해하지 않아도 되지만 모르는 부분을 알고 싶어 미치겠다면 편하게 질문해도 됩니다.) 어쨌든 돌아와서, 위 과정에서 원뿔의 부피 V보다 작은 S_n과 큰 T_n을 구하여 샌드위치 정리로 원뿔의 부피가 $\frac{1}{3}\pi r^2 h$ 임을 보였다. 그런데 위 예시처럼 원뿔을 작은 원기둥으로 무수히 나누었을 때 원뿔의 부피가 결국 될 것 같다면 S_n과 T_n중 하나만의 극한을 구해도 부피는 구할 수 있지 않을까? 맞다. 그리고 사실 구분구적법은 함수에도 적용할 수 있다.



$f(x)$가 $[a, b]$에서 연속, $f(x)\ge 0$ 일 때
$y=f(x)$, x축, $x=a$, $x=b$로 둘러싸인 도형의 넓이: S

의 넓이를 S_n이라 하자.

$$S = \lim_{n\to\infty} S_n = \lim_{n\to\infty}\sum_{k=1}^{n} f(x_k)\cdot\Delta x = \int_a^b f(x)\,dx$$
$$\left(\text{단, } \Delta x = \frac{b-a}{n},\ x_k = a + k\Delta x\right)$$
$$= \lim_{n\to\infty}\sum_{k=0}^{n-1} f(x_{k-1})\Delta x$$

이때 $f(x)$의 a에서 b까지의 정적분을 $\int_a^b f(x)dx = \lim_{n\to\infty} f(x_k)\Delta x$, $\left(\Delta x = \frac{b-a}{n},\ x_k = a + k\Delta x\right)$ 이라고 한다.

실습 구분구적법으로 $[a, b]$에서 $x = a$, $x = b$, $y = f(x)$, x축으로 둘러싸인 넓이 구하기

위의 수학적 지식은 중요하다. 위를 이해한 후 시뮬레이션을 제작하는 것이 중요하다. 여기서 다루는 시뮬레이션은 복잡하지 않으니 가벼운 마음으로 시작해보자.

먼저 필요한 라이브러리(모듈)를 import 해보자.

```
import numpy as np
# numpy: 리스트를 활용한다면 항상 import하는 게 좋을 것 같아요
# 근데 사실 리스트는 거의 항상 활용합니다. 실용적인 코드를 작성한다면 그럴 때마다
# numpy를 첫 번째 줄에 쓰고 있는 자기 자신을 발견하게 될 겁니다.
import matplotlib.pyplot as plt
# 그래프 그리는 기능을 포함한 라이브러리 matplotlib
# 에서 pyplot 만을 plt로 import
from matplotlib.widgets import Slider, Button
# matplotlib.widgets 에서 Slider와 Button이라는 class를 import
from sympy import Integral, Symbol     # 정적분 수행을 위한 import
```

함수들을 정의해보자.

```
# 함수 정의 (그릴 함수를 직접 쓰면 됩니다.)
def f(x):
    return (pow(x, 3) -9 *x)

# 작은 직사각형들 그리는 함수
def draw_rects(n):
    S_left =0     # 구분구적법으로 구할 넓이(0~n-1)
    S_right =0     # 구분구적법으로 구할 넓이(1~n)
    dx = (b - a) / n
    for k in range(0, n):
        x_k = a +k *dx
        ax.hlines(f(x_k+dx), x_k, x_k +dx, colors ='blue')
        ax.vlines(x_k, min(f(x_k+dx), 0), max(f(x_k +dx), 0), colors ='blue', linewidth =2)
        ax.vlines(x_k+dx, min(f(x_k +dx), 0), max(f(x_k +dx), 0), colors ='blue', linewidth =2)
        S_right += f(x_k +dx)*dx
    for k in range(0, n):
        x_k = a +k *dx
        ax.hlines(f(x_k), x_k, x_k+dx, colors ='red')
        ax.vlines(x_k, min(f(x_k), 0), max(f(x_k), 0), colors ='red')
        ax.vlines(x_k+dx, min(f(x_k), 0), max(f(x_k), 0), colors ='red')
        S_left += f(x_k) * dx
    return S_right, S_left

# 구분구적법으로 구한 넓이(S_left, S_right)와 정적분 값 출력하는 함수
def print_Sum(S_left, S_right):
    text ='\n'.join((
        f'S_right={S_right:.2f}',
        f'S_left={S_left:.2f}',
        f'Integral={integ:.2f}'))
    props = dict(boxstyle ='round', facecolor ='white', alpha =0.5)
    ax.text(0.05, 0.95, text, transform =ax.transAxes, fontsize =10,
            verticalalignment='top', bbox =props)
```

변수 설정(초기화)

```
# 구간([a, b])과 구간을 나누는 정도(n) 정의
a =-0.0000001
b =4.0
n =100
```

그래프 그리기

```
# a부터 0.00005씩 더한 수들로 리스트 생성
t = np.arange(a, b, 0.00005)

fig, ax = plt.subplots()      # 1개의 subplot 생성
line, = plt.plot(t, f(t), 'g')      # 리스트 t 각 원소에 대한 함숫값 리스트 f(t)로 그래프 그리기
S_right, S_left = draw_rects(n)      # 작은 직사각형들 그리기
plt.hlines(0, a, b, colors ='black')      # x축 표시
```

정적분 값 구하기

```
# 실제 정적분 값 구하기
x = Symbol('x')
g = (pow(x, 3) -9 *x)
integ = Integral(g, (x, a, b)).doit()
print_Sum(S_left, S_right)
```

슬라이더 만들기(위해 subplot 위치도 재설정)

```
plt.subplots_adjust(left=0.25, bottom =0.25)      # suplot 위치 설정
# 슬라이더 만들기 (Slider 클래스 객체 생성)
n_slider = Slider(
    ax=plt.axes([0.25, 0.1, 0.65, 0.03]),
    label='n',
    valmin=50,
    valmax=250,
    valinit=n,
    valstep=1
)
```

슬라이더 입력 바탕으로 업데이트하는 기능 구현

```
# subplot의 축 ax를 이용하여 그래프 다시 그리는 함수
# + print_Sum()
def update(val):
    ax.clear()
    ax.plot(t, f(t), 'g')
    ax.hlines(0, a, b, colors ='black')
    S_right, S_left = draw_rects(n_slider.val)
    print_Sum(S_left, S_right)

# on_changed: slider 값이 바뀌면 지정한 함수 실행

n_slider.on_changed(update)
```

리셋 기능 만들기

```
# 리셋 버튼 설정(위치: reset_ax, 표시 텍스트, 마우스 포인터가 올려졌을 시 색)
reset_ax = plt.axes([0.8, 0.025, 0.1, 0.04])
button = Button(reset_ax, 'Reset', hovercolor ='0.975')


# 슬라이더 객체의 각 멤버 변수의 값을 초기값으로 설정
def reset(event):
    n_slider.reset()


button.on_clicked(reset)
```

팝업

```
plt.show()    # 출력
```

[전체 코드]

```
import numpy as np
# numpy: 리스트를 활용한다면 항상 import하는 게 좋을 것 같아요
# 근데 사실 리스트는 거의 항상 활용합니다. 실용적인 코드를 작성한다면 그럴 때마다
# numpy를 첫 번째 줄에 쓰고 있는 자기 자신을 발견하게 될 겁니다.
import matplotlib.pyplot as plt
# 그래프 그리는 기능을 포함한 라이브러리 matplotlib
# 에서 pyplot 만을 plt로 import
from matplotlib.widgets import Slider, Button
# matplotlib.widgets 에서 Slider와 Button이라는 class를 import
from sympy import Integral, Symbol     # 정적분 수행을 위한 import


# ----------------------- 함수 정의 -----------------------


# 함수 정의 (그릴 함수를 직접 쓰면 됩니다.)
def f(x):
    return (pow(x, 3) -9 *x)


# 작은 직사각형들 그리는 함수
def draw_rects(n):
    S_left =0     # 구분구적법으로 구할 넓이(0~n-1)
    S_right =0     # 구분구적법으로 구할 넓이(1~n)
    dx = (b - a) / n
    for k in range(0, n):
        x_k = a +k *dx
        ax.hlines(f(x_k+dx), x_k, x_k +dx, colors ='blue')
        ax.vlines(x_k, min(f(x_k+dx), 0), max(f(x_k +dx), 0), colors ='blue', linewidth =2)
        ax.vlines(x_k+dx, min(f(x_k +dx), 0), max(f(x_k +dx), 0), colors ='blue', linewidth =2)
        S_right += f(x_k +dx)*dx
    for k in range(0, n):
        x_k = a +k *dx
        ax.hlines(f(x_k), x_k, x_k+dx, colors ='red')
        ax.vlines(x_k, min(f(x_k), 0), max(f(x_k), 0), colors ='red')
        ax.vlines(x_k+dx, min(f(x_k), 0), max(f(x_k), 0), colors ='red')
        S_left += f(x_k) * dx
    return S_right, S_left


# 구분구적법으로 구한 넓이(S_left, S_right)와 정적분 값 출력하는 함수
def print_Sum(S_left, S_right):
    text ='\n'.join((
        f'S_right={S_right:.2f}',
        f'S_left={S_left:.2f}',
        f'Integral={integ:.2f}'))
    props = dict(boxstyle ='round', facecolor ='white', alpha =0.5)
    ax.text(0.05, 0.95, text, transform =ax.transAxes, fontsize =10,
            verticalalignment='top', bbox =props)


# 구간([a, b])과 구간을 나누는 정도(n) 정의
a =-0.0000001
b =4.0
n =100
```

```
# a부터 0.00005씩 더한 수들로 리스트 생성
t = np.arange(a, b, 0.00005)

fig, ax = plt.subplots()     # 1개의 subplot 생성
line, = plt.plot(t, f(t), 'g')     # 리스트 t 각 원소에 대한 함숫값 리스트 f(t)로 그래프 그리기
S_right, S_left = draw_rects(n)     # 작은 직사각형들 그리기
plt.hlines(0, a, b, colors ='black')     # x축 표시

# 실제 정적분 값 구하기
x = Symbol('x')
g = (pow(x, 3) -9 *x)
integ = Integral(g, (x, a, b)).doit()
print_Sum(S_left, S_right)

plt.subplots_adjust(left=0.25, bottom =0.25)     # suplot 위치 설정
# 슬라이더 만들기 (Slider 클래스 객체 생성)
n_slider = Slider(
    ax=plt.axes([0.25, 0.1, 0.65, 0.03]),
    label='n',
    valmin=50,
    valmax=250,
    valinit=n,
    valstep=1
)

# subplot의 축 ax를 이용하여 그래프 다시 그리는 함수
# + print_Sum()
def update(val):
    ax.clear()
    ax.plot(t, f(t), 'g')
    ax.hlines(0, a, b, colors ='black')
    S_right, S_left = draw_rects(n_slider.val)
    print_Sum(S_left, S_right)
# on_changed: slider 값이 바뀌면 지정한 함수 실행

n_slider.on_changed(update)

# 리셋 버튼 설정(위치: reset_ax, 표시 텍스트, 마우스 포인터가 올려졌을 시 색)
reset_ax = plt.axes([0.8, 0.025, 0.1, 0.04])
button = Button(reset_ax, 'Reset', hovercolor ='0.975')


# 슬라이더 객체의 각 멤버 변수의 값을 초기값으로 설정
def reset(event):
    n_slider.reset()


button.on_clicked(reset)

plt.show()     # 출력
```