Math as Code (Python version)

이 문서는 Python 코드와의 비교를 통해 개발자가 수학 표기법을 더 쉽게 이해할 수 있도록 돕는 참고 자료입니다.

서문

수학 기호는 저자, 문맥, 그리고 연구 분야(선형대수학, 집합론 등)에 따라 다른 의미를 가질 수 있습니다. 이 가이드는 기호의 모든 용도를 다루지 않을 수 있습니다. 경우에 따라, 기호가 실제 어떻게 사용되는지 보여주기 위해 실제 참고 자료(블로그 포스트, 출판물 등)를 인용할 것입니다.

더 완전한 목록은 위키피디아 - 수학 기호 목록을 참조하세요.

단순화를 위해, 여기의 많은 코드 예제는 부동 소수점 값으로 작동하며 수치적으로 안정적이지 않을 수 있습니다. 이것이 왜 문제가 될 수 있는지에 대한 자세한 내용은 Mikola Lysenko의 Robust Arithmetic Notes를 참조하세요.

목차

변수명 규칙

문맥과 연구 분야에 따라 다양한 명명 규칙이 있으며, 항상 일관되지는 않습니다. 하지만 일부 문헌에서는 다음과 같은 패턴을 따르는 변수명을 찾을 수 있습니다:

- s - 스칼라(예: 숫자)를 위한 이탤릭체 소문자
- **x** - 벡터(예: 2D 점)를 위한 굵은 소문자
- **A** - 행렬(예: 3D 변환)을 위한 굵은 대문자
- θ - 상수 및 특수 변수(예: 극좌표 각도 θ, 세타)를 위한 이탤릭체 소문자 그리스 문자

이 가이드도 이 형식을 따를 것입니다.

Numpy

Numpy는 강력한 **배열 프로그래밍** 라이브러리로, 파이썬에서는 **도메인 특화 언어**(DSL)로 해석될 수 있습니다. 때로는 파이썬의 수학을 두 언어가 네임스페이스를 공유하고, 특별한 문법 설탕으로 서로 접근하는 것으로 생각하는 것이 도움이 됩니다. 이는 벡터와 행렬 섹션에서 중요해질 것입니다. 왜냐하면 약간 다른 파이썬 문법이 큰 입력에서 다른 속도를 의미하기 때문입니다. 관례는 `import numpy as np`이므로, `np.something`을 보면 numpy로 작업하고 있음을 알 수 있습니다.

등호 기호

등호 =와 유사한 여러 기호가 있습니다. 다음은 몇 가지 일반적인 예입니다:

- = : 동등 (값이 같음)
- ≠ : 부등 (값이 다름)
- ≈ : 거의 같음 (π ≈ 3.14159)
- := : 정의 (A는 B로 정의됨)

파이썬에서:

```
## 동등
2 == 3

## 부등
2 != 3

## 거의 같음
import math
math.isclose(math.pi, 3.14159) # isclose는 허용 오차 인수가 없어 False

from numpy.testing import assert_almost_equal
assert_almost_equal(math.pi, 3.14159, 1e-5) # 허용 오차 5자리 지정

def almost_equal(x, y, epsilon=7):
  """직접 만들 수도 있습니다!
  numpy에서는 1e-7이 기본 엡실론입니다.
  """
  return abs(x - y) < 10 ** -epsilon
```

수학 표기법에서는 :=, =:, = 기호가 **정의**에 사용될 수 있습니다. 예를 들어, 다음은 x를 2kj의 다른 이름으로 정의합니다.

$x := 2kj$

파이썬에서는 =로 변수를 **정의**하고 별칭을 부여합니다.

```
x = 2 * k * j
```

반면에 다음은 동등성을 나타냅니다:

$x = 2kj$

중요: =와 ==의 차이점은 수학 문헌보다 코드에서 더 명확할 수 있습니다! 파이썬에서 =는 **명령**입니다. 반면 ==는 기계에게 "bool 값을 주시겠어요?"라고 묻는 것입니다. 수학에서는 전자의 경우가 := 또는 =로, 후자의 경우는 **보통** =로 표현되므로 문맥에 따라 구분해야 할 수 있습니다.

제곱근과 복소수

제곱근 연산은 다음과 같은 형태입니다:

$\sqrt{x^2} = x$

프로그래밍에서는 `sqrt` 함수를 사용합니다:

```
import math
print(math.sqrt(2))
# 결과: 1.4142135623730951
```

복소수는 $a + ib$ 형태의 표현식으로, a는 실수부, b는 허수부입니다. 허수 i는 다음과 같이 정의됩니다:

$i = \sqrt{-1}$

파이썬은 `complex` 생성자와 `cmath` 표준 모듈을 제공합니다.

```
complex(1, 1)  # (1+1j)

import cmath
cmath.sqrt(complex(-1, 0)) # (0+1j)
```

내적 & 외적

점 · 과 십자 × 기호는 문맥에 따라 다른 용도로 사용됩니다.

스칼라 곱셈

두 기호 모두 스칼라의 단순 곱셈을 나타낼 수 있습니다. 다음은 동일합니다:

$5 \cdot 4 = 5 \times 4$

프로그래밍 언어에서는 곱셈에 별표 *를 사용합니다.

벡터 곱셈

벡터와 스칼라의 곱셈이나 벡터 간의 요소별 곱셈을 나타낼 때는 일반적으로 점 ·이나 십자 × 기호를 사용하지 않습니다. 요소별 벡터 곱셈에는 열린 점 ◦을 사용하여 아다마르 곱을 나타낼 수 있습니다.

$3k \circ j$

내적 (Dot Product)

점 기호 ·는 두 벡터의 내적을 나타내는 데 사용될 수 있습니다. 스칼라 값을 반환하므로 스칼라 곱이라고도 합니다.

$k \cdot j$

```
k = [0, 1, 0]
j = [1, 0, 0]
d = np.dot(k, j) # 결과: 0
```

외적 (Cross Product)

십자 기호 ×는 두 벡터의 외적을 나타내는 데 사용될 수 있습니다.

$k \times j$

```
k = [0, 1, 0]
j = [1, 0, 0]
result = np.cross(k, j) # 결과: [ 0, 0, -1 ]
```

시그마 Σ

큰 그리스 문자 Σ (시그마)는 합산을 의미합니다.

$\sum_{i=1}^{100} i$

여기서 i=1은 1에서 시작하여 시그마 위의 숫자인 100에서 끝남을 의미합니다. 코드에서는 다음과 같습니다:

```
sum([k for k in range(1, 101)]) # 5050
```

대문자 파이 Π

대문자 파이 또는 "빅 파이"는 시그마와 매우 유사하지만, 값의 수열의 곱을 찾는 데 곱셈을 사용합니다.

$\prod_{i=1}^{6} i$

```
from functools import reduce
def times(x, y): return x * y
reduce(times, range(1, 7)) # 720
```

파이프 ||

파이프 기호는 문맥에 따라 다른 의미를 가질 수 있습니다. 다음은 세 가지 일반적인 용도입니다: 절댓값, 유클리드 노름, 행렬식.

- **절댓값**: $|x|$
- **유클리드 노름**: $\| \boldsymbol{v} \|$
- **행렬식**: $\det(\boldsymbol{A})$

햇 â

기하학에서 문자 위의 "햇" 기호는 단위 벡터를 나타내는 데 사용됩니다.

$\hat{a}$

단위 벡터는 길이가 1인 벡터입니다. 3D 벡터를 정규화하는 예:

```
def normalize(vec):
  # ... 구현
  return vec
```

"원소" ∈, ∉

집합론에서 "원소" 기호 ∈는 어떤 것이 집합의 원소인지 여부를 설명하는 데 사용됩니다.

$A = \{3, 9, 14\}, 3 \in A$

파이썬에서는 `in` 키워드가 이 역할을 합니다.

```
A = {3, 9, 14}
3 in A # True
```

일반적인 수 집합

- $\mathbb{R}$: 실수 (Python의 `float`)
- $\mathbb{Q}$: 유리수
- $\mathbb{Z}$: 정수 (Python의 `int`)
- $\mathbb{N}$: 자연수 (0을 포함하는 양의 정수)
- $\mathbb{C}$: 복소수 (Python의 `complex`)

함수 ƒ

함수는 입력을 출력 값으로 변환합니다. $f(x) = x^2$는 다음과 같이 작성할 수 있습니다:

```
def square(x):
  return x**2
```

조각별 함수

입력 값에 따라 다른 관계를 사용하는 함수입니다.

$f(x) = \begin{cases} \frac{x^2 - x}{x} & \text{if } x \geqslant 1 \\ 0 & \text{otherwise} \end{cases}$

```
def f(x):
  if (x ⩾ 1):
    return (x**2 - x) / x
  else:
    return 0
```

소수 ′

소수 기호(′)는 변수 이름에서 비슷한 것들을 구별하거나, 변환 후의 "다음 값"을 설명하는 데 사용됩니다. 함수에서는 함수의 **도함수**를 나타냅니다.

$f(x) = x^2$ 의 도함수는 $f'(x) = 2x$ 입니다.

바닥 & 천장 ⌊, ⌉

- $\lfloor x \rfloor$: 내림
- $\lceil x \rceil$: 올림
- $\lfloor x \rceil$: 반올림

```
import math
math.floor(4.8) # 4
math.ceil(3.1)  # 4
round(4.5)      # 4 (Python 3의 round는 짝수에 가까운 쪽으로)
```

화살표

- 논리적 함의: $A \Rightarrow B$ (A가 참이면 B도 참)
- 부등식: <, >, ⩽, ⩾
- 논리곱과 논리합: ∧ (AND), ∨ (OR)

논리 부정 ¬, ~, !

논리적 `NOT`을 나타냅니다. $x \neq y \Leftrightarrow \neg(x = y)$

구간

특정 값의 범위를 나타냅니다.

- $(0, 1)$: 0과 1을 포함하지 않음
- $[0, 1)$: 0은 포함, 1은 포함하지 않음
- $(0, 1]$: 0은 미포함, 1은 포함
- $[0, 1]$: 0과 1을 모두 포함