Functions

March 29, 2021

0.1 Notes

1. The notebooks are largely self-contained, i.e, if you see a symbol there will be an explanation about it at some point in the notebook.
   - Most often there will be links to the cell where the symbols are explained
   - If the symbols are not explained in this notebook, a reference to the appropriate notebook will be provided
2. **Github does a poor job of rendering this notebook**. The online render of this notebook is missing links, symbols, and notations are badly formatted. It is advised that you clone a local copy (or download the notebook) and open it locally.
3. **See the Collections notebook before this notebook to gain familiarity with set notations**

1 Contents

1.1 Importing Libraries

[1]:
```python
import math
import random
import typing
```

Fundamentals

Set map

If A and B are any sets, a function f that maps values from set A to values in set B is denoted as:

$$f : A \rightarrow B$$

This denotes that the function f takes as input, values from set A and returns values that belong to set B.

In sciences and engineering, most functions take real numbers as input and output real numbers. This can be denoted as: $f : \mathbb{R} \rightarrow \mathbb{R}$

Note: Not all values of B needs to be realized. For example, cosine function takes any real number as input and outputs only real numbers BUT the real numbers remain in the bound $[-1, 1]$. Related concept: see Image of a function

[2]:
```python
def inverse(a):
    return 1/a

# inverse: A -> B

A = set([2, 4, 8])
B = set([1, 0.75, 0.5, 0.25, 0.125])

for A_i in A:
    print( inverse(A_i) in B )
```

```
True
True
True
```

Image of a function

For a function $f : X \rightarrow Y$, the image of the function f is denoted as: $\text{Im}(f)$

Essentially, the image of the function is the subset of B, where each value is the output of the application of f to elements in A. Formally:

$$\text{Im}(f) = \{y \in Y |\, \exists x \in X, f(x) = y\}$$

For more info on the there-exists notations, please see the Logic notebook

[3]:
```python
def inverse(a):
    return 1/a

# inverse: A -> B

A = set([2, 4, 8])
B = set([1, 0.75, 0.5, 0.25, 0.125])

image_of_inverse_A = []

for A_i in A:
    image_of_inverse_A.append(inverse(A_i))

print('A: ',A)
print('B: ',B)
print('Im(A): ',set(image_of_inverse_A))
print('Im(A) is subset of B: ', set(image_of_inverse_A).issubset(B))
```

```
A:  {8, 2, 4}
B:  {0.75, 1, 0.125, 0.5, 0.25}
Im(A):  {0.125, 0.5, 0.25}
Im(A) is subset of B:  True
```

Kernel of a function

In general the definition of the kernel may be complicated but an important special case of interest is the kernel of a linear map. For a linear mapping function $f : A \to B$, the kernel of f is denoted and defined as:

$$\ker f = \{\mathbf{x} \in A |\ f(\mathbf{x}) = 0\}$$

Note 1: The kernel of a matrix is also called the null space

Note 2: This concept is different from the kernel functions as used in the kernel trick in SVM.

For more information on the Kernel of a Matrix, please see the Linear Algebra Notebook

---

Generic value notation

The value y returned by a function f when the value x is passed to it is generically denoted as:

$$y = f(x)$$

Here x is called the argument to function f which returns y. Functions that accept multiple arguments can be generally denoted as:

$$y = f(x_1, x_2, ..., x_n)$$

The terms variable, arguments, and parameters can have more indepth and rigorous definitions, but here it is shown based on common usage and understanding. In common usage, a parameter is used as a term to define a quantity that influences the output or behavior of a mathematical object but is viewed as being held constant. For example, a function with argument x and parameter a (a is constant):

$$y = f(x, a) = a\sin(x)$$

```
[4]: func = math.sin

     # y = f(x)
     x = random.random()
     y = func(x)

     y, x
```

```
[4]: (0.4818035028875088, 0.5027116877317647)
```

```
[5]: def func(x_1, a):
         return  a * math.sin(x_1)

     # y = f(x1, a)

     x_1 = random.random()
     a = 2
     y = func(x_1, a)

     y, x_1, a
```

```
[5]: (0.3899580723353795, 0.19623607606244153, 2)
```

Note 1:

Sometimes an alternative notation is used to denote function evaluation especially if the function itself is specified using a formula. The general form is

$$\text{Expression involving variable } x \,\Big|_{x=a}$$

This means to evaluate the expression by substituting the value a for the variable x. This is used in Leibniz notations especially in evaluating Taylor's expansions: $\frac{df}{dx}\Big|_{x=a}$

Dot notation

Sometimes, the function needs to be referred to in a non-letter context or without using a dummy variable. Blank spaces are generally not preferrable. In such cases we can use a dot showing that an argument is expected, such as: $f(\cdot)$

An example of this is discussing the absolute value of a number, generally it is denoted as: $|x|$ but if the letter needs to be omitted and the function needs to be applied to a generic object, it can be shown using the dot notation as: $|\cdot|$

This notation can be very useful to show functions with multiple arguments where one or more arguments can vary while others are constant (parameters or arguments). For example:

$$f(\cdot, b)$$

represents a function f where the first argument can vary over it's domain, but the second argument is held constant at value b

[6]:
```python
# f(. , b)

def func(x, b = 5):
    return b*x**2

x = [0.5, 2, 10]
print('x: ', x, '\n')

for i, x_i in enumerate(x):
    print(f'y_{i} = {func(x_i)}')
```

```
x:  [0.5, 2, 10]

y_0 = 1.25
y_1 = 20
y_2 = 500
```

Map notation

The notation $y = f(x)$ can also be abbreviated generically through:

$$x \mapsto y$$

This is read as 'x maps to y' and it is to be noted that f is missing in the expression. In general this is used to communicate that there is a need to map x to y. Either that specific function that does the mapping is as of yet unknown, or is unambiguously known. This is upto the author to clarify.

Alternatively, $y = f(x)$ can be unambiguously abbreviated using:

$$x \overset{f}{\mapsto} y$$

[7]:
```python
func = math.cos

# x maps to y
x = random.random()
y = func(x)

y, x
```

[7]:
```
(0.9525168584064242, 0.3093986049633054)
```

Operator/Transform

A function whose inputs and outputs are themselves functions is often called an *operator* or *transform*. Generally, we would like to distinguish an operator from function of numbers by capitalizing the operator name and omitting the paranthesis.

For example, the evaluation of operator L (notice the capitalization) on function f can be denoted as:

$$Lf$$

A specific example of this is the Laplace transform using the Laplace tranform operator $\mathcal{L}$. This operator is often used on a temporal function $f(t)$ whichs maps it to a function in the complex space $F(s)$, such that $\mathcal{L}f = F$

Piecewise notation

Sometimes it is not possible to define a function as a single algebraic formula. This is very true when the function is defined over various unconnected domains, or if it is defined differently over different domains. In such cases, the function can be defined piece-wise where the function is defined explicitely over each respective domain.

$$f(x) = \begin{cases} e^{-x}, & \text{if } x > 0 \\ 1, & \text{if } x = 0 \\ 0, & \text{otherwise} \end{cases}$$

There aren't many standards on how piece-wise functions need to be defined. Some authors will use text such as 'otherwise', other authors will explicitely define all domains using notations. Some authors do not define the function over some domains with the understanding that the function is undefined over those domains. Context may be important in such cases.

[8]:
```python
def piece_func(x):
    if x > 0:
        return math.exp(-x)
    elif x == 0.0:
        return 1
    else:
        return 0


x = random.random()*random.randint(-10,10)
y = piece_func(x)

y, x
```

[8]:
```
(0.4056142603500078, 0.9023526686693493)
```

Composition of functions

Consider two functions, $f : A \to B$ and $g : B \to C$, then the *composition* of g and f is a new function created by the application of g on the values returned by f. The composition of functions is denoted by:

$$(g \circ f) : A \to C$$

where $(g \circ f)(x) = g(f(x))$

Note: The order of the composition matters since $(f \circ g)(x) = f(g(x))$ which is rarely the same.

[9]:
```
def f(a):
    return 2*a + 5

def g(b):
    return b**2

composition_g_f = lambda a: g(f(a))
composition_f_g = lambda a: f(g(a))

a = 5

print('a:',a,', f(a):', f(a), ', gof(a):', composition_g_f(a), ', fog(a):',␣
 ↪composition_f_g(a))
```

```
a: 5 , f(a): 15 , gof(a): 225 , fog(a): 55
```

Inversion of functions

Consider a function $f(x) = y$, then the function that operates on y and returns x is called the inverse of function f and is denoted as: f^{-1}

$$f^{-1}(y) = x$$

If the function $f(x) = y$ is such that multiple values of x can have the same y (like $y = x^2$), then f^{-1} is not a function. However, some authors use the notation $f^{-1}(y)$ to stand for the set of all x values that map to y without actually having an explicit function in mind.

Soem authors also coerce f^{-1} to be a function by restricting the domain of x. For example the arcsin function is $\sin^{-1}$. There are infinitely many values of x that can produce the same $y = \sin(x)$ such as $\sin(x) = \frac{1}{2}$. However, if we restrict x to $(-\frac{\pi}{2}, \frac{\pi}{2}]$, arcsin becomes a valid function.

Warning: $f^{-1}(y)$ does NOT mean $\frac{1}{f(y)}$. See: Functions exponentiations

Note: In general, it can be a challenge to find an explicit form for f^{-1}. Mostly, computational methods are used to find an approximate form for f^{-1}

[10]:
```python
def f(x):
    return math.exp(x)

def f_inverse(x):
    return math.log(x)

x = 1.0
print('x: ', x)
y = f(x)
print('y = f(x): ', y)
print('x = f_inverse(y): ', f_inverse(y))
```

```
x:  1.0
y = f(x):  2.718281828459045
x = f_inverse(y):  1.0
```

The notation of f^{-1} can also be applied to a set of values. Given a function f operating on sets X, Y denoted as $f : X \to Y$ and if we have a set $B \subseteq Y$, then

$$f^{-1}(B) = \{x \in X |\ f(x) \in B\}$$

Functions exponentiations

In general, when a function is exponentiated, it is denoted as:

$$f^n(x)$$

This means straight forward exponentiation of the function, for example: \$ f^2(x) = [f(x)]^2\$ (Square brackets are a stylistic choice and don't hold any meaning here). This notation is used very commonly with trignometric functions, for example: $\sin^2\theta = (\sin\theta)^2$

Warning: However, this notation brings inconsistency because $f^{-1}(y)$ does NOT mean $[f(y)]^{-1}$. But in some contexts $f^2(x)$ may mean $(f \circ f)(x)$ which is $f(f(x))$ which is consistent with meaning of inverse. See: Composition of functions and Inversion of functions

[11]:
```
def f(x):
    return math.sin(x)

x = 1
print('x: ',x, ', f(x): ', f(x),', f^2(x): ', f(x)**2)
```

```
x:  1 , f(x):  0.8414709848078965 , f^2(x):  0.7080734182735712
```

Classes of functions

Continuous functions

This subtopic deals with notations used to decribe whole classes of functions. One of the most important of these classes of functions are functions that are continuous and hence differentiable. These are generally denoted by:

$$C^k$$

which describes a class of functions C with k continuous derivatives and whose k^{th} derivative is defined and continuous. Thus C^0 stands for class of continuous functions, C^1 stands for class of differentiable functions whose derivatives are continuous, and C^∞ contains those functions for which derivatives of all orders exist:

$$C^0 \supset C^1 \supset C^2 \supset ...$$

This may also be denoted as $C^k(\mathbb{R})$ which is equivalent to just C^k

---

Polynomial functions

For info on polynomials see: Polynomial functions

The set of *all* polynomials in the variable x with real coefficients is denoted by:

$$\mathbb{R}[x]$$

The notation for *all* polynomials in multiple variables $x_1, x_2, ..., x_n$ with real coefficients is denoted by:

$$\mathbb{R}[x_1, x_2, ..., x_n]$$

Polynomial functions

Introduction

A polynomial $p(x)$ is a function of the form:

$$p(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + ... + a_1 x + a_0$$

where the collection of a_i are called coefficients.

Some authors prefer the subscripts on the coefficients to run counter to the exponents:

$$p(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + ... + a_{n-1}x + a_n$$

Mostly the degree of the polynomial (See: Degree of polynomial) is known from context, but if the degree is not known, the polynomial function notation will include n as an argument as well: $p(x, n)$. Additionally, at times even the coeffiecients may be arguments and these may be specified as a vector of coefficients: $p(x, \mathbf{a})$ or $p(x, n, \mathbf{a})$ although n is implied from $\mathbf{a}$.

See the linear algebra notebook for more info on vector notations

[12]:
```
# p(x,A)

class Poly:

    def __init__(self, a: typing.List):
        """
        Initializes coefficients
        """
        self.a = a

    def p(self, x):
        """
        Evaluates p(x)
        """
        p = 0
        for i,a_i in enumerate(self.a):
            p += a_i*(x**i)
        return p

#Initialize p(x) = 5*x^2 + 10*x + 20
example = Poly([20, 10, 5])

#Evaluate p(2) = 5*(2^2) + 10*(2) + 20
example.p(2)
```

[12]:
```
60
```

Note 1: Polynomials may have more than one variables such as: $p(x, y) = x^3 + 5x^2y^2 + 6xy + 8y^3$

Degree of polynomial

The degree of a polynomial is the largest exponent on x associated with a non-zero coefficient. It is denoted as:

$$\deg p(x)$$

If $p(x) = 0$ (the zero polynomial) then the degree is either undefined or $-\infty$

```
[13]: class PolyWithDeg(Poly):

          def deg(self):
              """
              Returns degree of polynomial
              """
              return len(self.a)-1 if ((len(self.a) > 0) and self.a != [0])  else␣
       ↪'undefined'


      #Initialize p(x) = 5*x^2 + 10*x + 20
      example = PolyWithDeg([20, 10, 5])
      example.deg()
```

```
[13]: 2
```

Note 1: For polynomials in two or more variables, the degree of a term is the sum of the exponents of the variables in the term; the degree (sometimes called the total degree) of the polynomial is again the maximum of the degrees of all terms in the polynomial.

For example: Let $p(x, y) = x^2y^2 + xy^2 + y^3$, $\deg p(x, y) = 4$