

Линейная алгебра. Векторы

Содержание:

1. Что такое линейная алгебра?
2. Что такое вектор?
3. Какие основные операции с вектором?
4. Линейная зависимость векторов.

Что такое линейная алгебра?

Линейная алгебра — это раздел математики, изучающий векторы, векторные пространства, линейные преобразования и системы линейных уравнений. То есть линейная алгебра — это язык и инструментарий для работы с векторами.

Что такое вектор?

Часто объекты, процессы из материального мира описываются какими-то наборами их числовых характеристик.

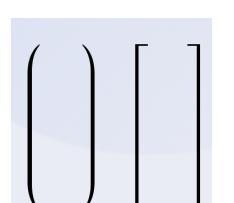
Вектор — упорядоченный конечный список чисел. Это любой элемент в пространстве, который мы получаем в виде данных. Например каждый элемент таблицы удобно представить в виде вектора.

Запись векторов.

Обычно вектор записывается в виде последовательности чисел, через запятую и все это берется в круглые скобки. Часто, когда вектор обозначается одной буквой, то над этой буквой ставится черточка.

$$\bar{x} = (x_1, \dots, x_n)$$

При этом сами числа x_1, \dots, x_n называются **компонентами** или **координатами вектора**. В различных математических статьях можно увидеть запись в виде так называемого вектора-столбца, когда компоненты вектора записываются по вертикали, и все это берется либо в круглые, либо в квадратные скобки.



$$\begin{matrix} x_1 & x_1 \\ x_2 & x_2 \\ \vdots & \vdots \\ x_n & x_n \end{matrix}$$

Вектора обычно записываются как вертикальный список, например:

$$\begin{bmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{bmatrix} \quad \begin{pmatrix} -1.1 \\ 0.0 \\ 3.6 \\ -7.2 \end{pmatrix}$$

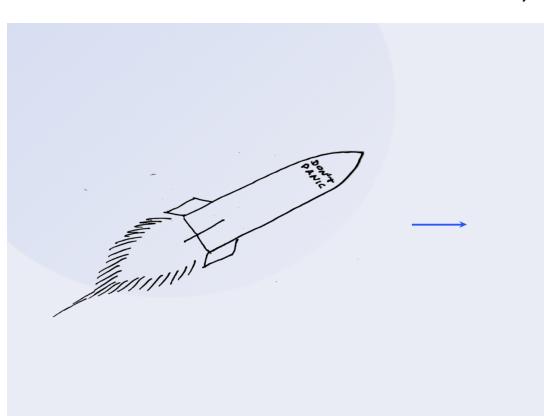
Вектор может быть записан также в следующем виде:

$$(-1.1, 0.0, 3.6, -7.2)$$

Векторы бывают разных размеров, от одномерного, представленного одним числом до n -мерного.

Пример вектора. Ракета, летящая в космосе.

Ракета, летящая в космосе, описывается своими координатами в пространстве x, y, z и компонентами ее скорости v_x, v_y, v_z . То есть шесть чисел описывают то, как устроена жизнь ракеты сейчас и что с ней будет дальше.

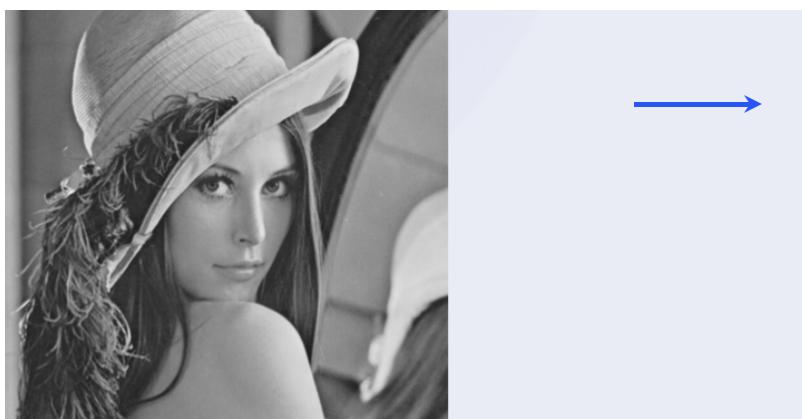


$$(x, y, z, v_x, v_y, v_z)$$

Ракета, летящая в космосе

Пример вектора. Цифровое изображение.

Цифровое изображение состоит из отдельных пикселей, а каждый пиксель задается числом — яркостью в данной точке.

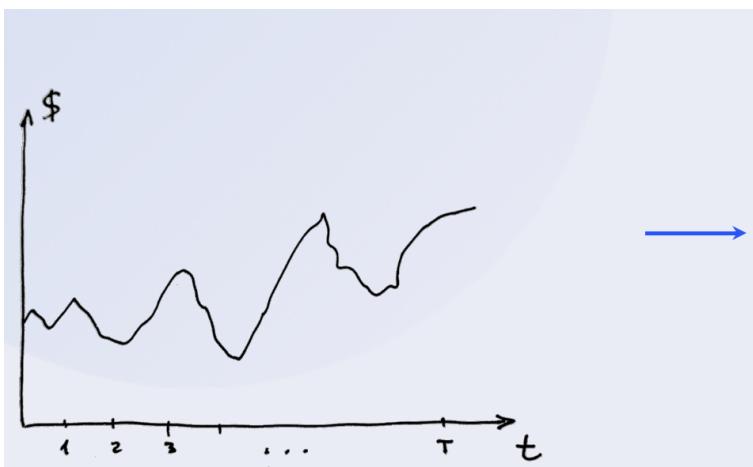


$(x_1, x_2, \dots, x_{512 \times 512})$
Изображение 512x512

Все пиксели можем записать просто в строчку. Их будет довольно много. Если у нас изображение 512x512, то будет ровно 512 умножить на 512 чисел. Они полностью описывают изображение.

Пример вектора. Различные курсы валют, котировки акций и другие временные ряды.

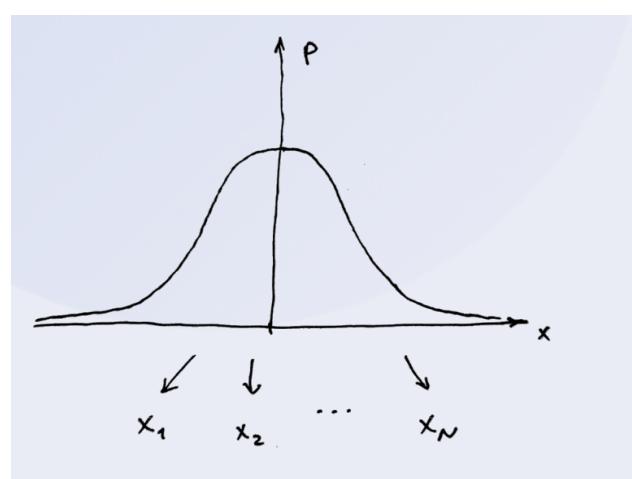
К временным рядам относится также запись звука или от части видео. Все эти вещи превращаются в последовательность чисел, так что мы смотрим на их значение в отдельные отсчеты времени — один, два и так далее, T большое. И это становится тоже упорядоченным набором чисел.



(x_1, x_2, \dots, x_T)
Временной ряд

Пример вектора. Выборка реализации случайной величины.

Изначально есть какая-то случайная величина, мы делаем какие-то испытания и смотрим на то, что вышло в итоге. Первое испытание, второе испытание и так далее, N -ое испытание. Чтобы сделать какой-то статистический анализ этих чисел, мы их тоже упаковываем в компактный набор, который называется вектор этих чисел.



(x_1, x_2, \dots, x_N)
выборка реализации случайной величины

Можно ли представить вектор в виде слов?

Да, для этого нужно понять, как распарсить слово по буквам и представить вектор в виде букв.

Буквы алфавита занимают определенное место и могут быть обозначены в виде порядкового номера (А-1, Б-2, Я-33). Любое слово можно представить как вектор в 33-мерном пространстве.

Но т.к. встречаются строчные и прописные буквы, то это будет вектор в 66-мерном пространстве.

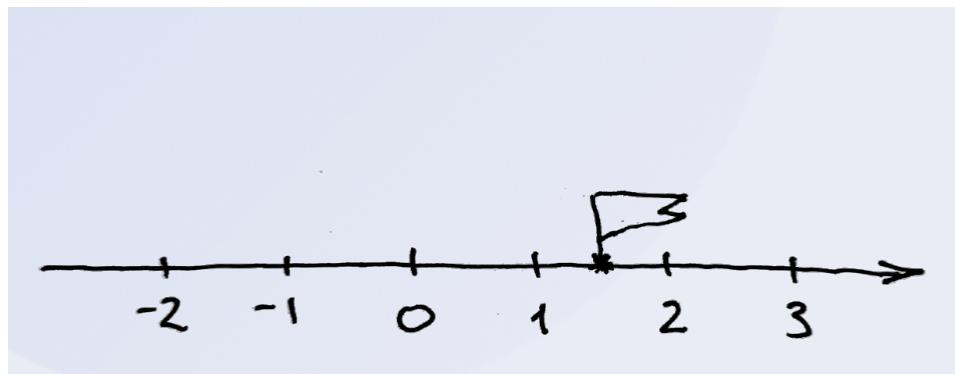
Геометрическая интерпретация векторов.

Когда есть длинный набор чисел, то с ним удобно делать разные арифметические операции, но не очень удобно представлять, как это выглядит. Для этого мы посмотрим как же устроена геометрическая интерпретация векторов.

Скаляр — это число. Обычно оно может быть натуральным либо действительным или десятичным. По сути вектор — набор из нескольких скаляров

$$s \in \mathbb{N}$$

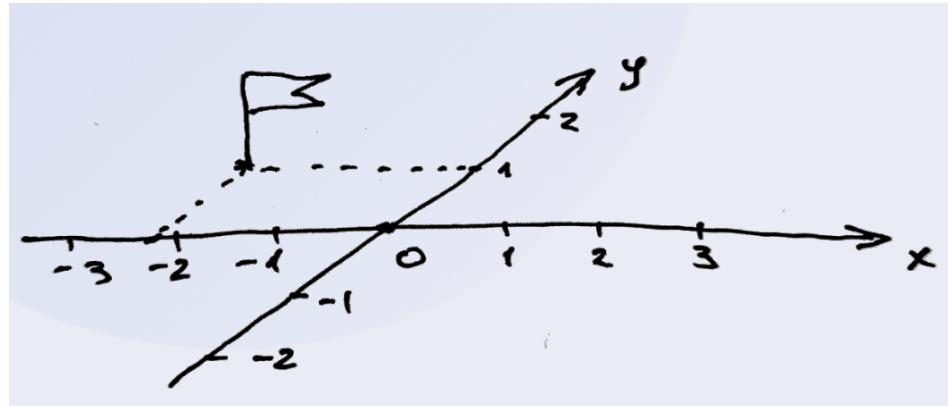
Вектор из одного числа ("скаляр") - точка на прямой.



Одномерный вектор, который состоит ровно из одного числа. Точка, равная 1.5.

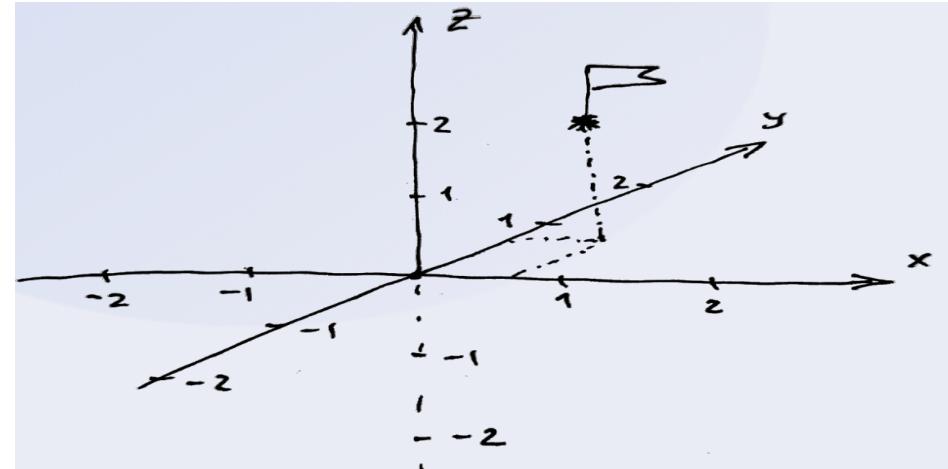
Вектор из двух чисел — точка на плоскости

Здесь первая компонента вектора — это координата точки по оси x, а вторая компонента вектора — это координата точки по оси y.



$$v = (-2.2, 1.0)$$

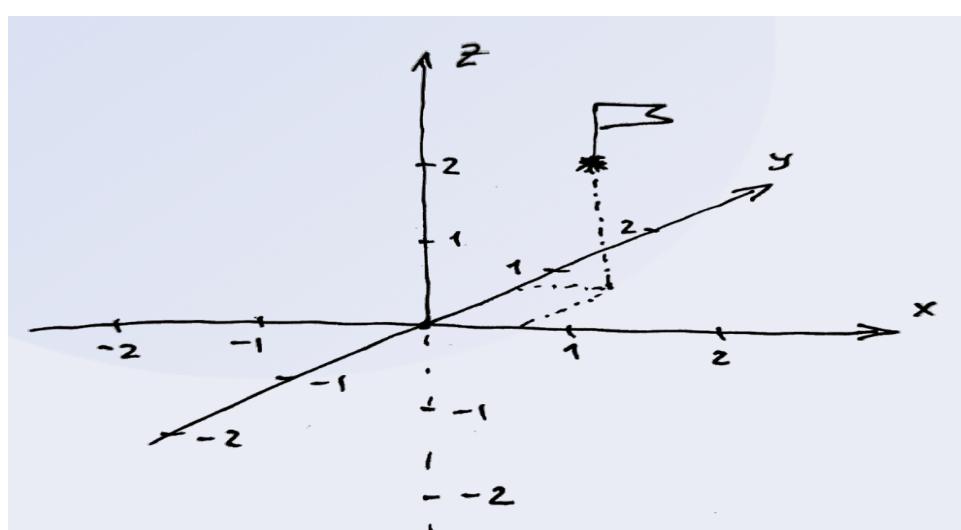
Вектор из трёх чисел — точка в трёхмерном пространстве



$$v = (0.7, 0.7, 1.5)$$

Вектор из N чисел — точка в N-мерном пространстве

Что такое N-мерное пространство уже не так легко представить. Обычно это легко себе представить просто по аналогии с трехмерным пространством, просто как будто координат стало больше.



$$v = (0.7, 0.7, 1.5, \dots)$$

Векторы на Python.

[Пример: Создание и вывод 4-мерного вектора](#)

Для работы с векторами на Python мы используем библиотеку `pintur`. Это довольно стандартная большая библиотека для работы с векторами, матрицами и еще очень многое.

В первой строчке мы ее импортируем. И дальше мы создаем вектор из четырех компонент с помощью команды np.array. Это некоторый объект класса array, что переводится как «массив». В третьей строке мы просим Python вывести нам то, что находится в переменной v. Мы видим, что у нас получился массив из тех самых чисел, которые мы туда передали. При создании вектора пишем в квадратных скобках, то есть в виде питоновского списка те числа, из которых мы хотим сделать вектор. Для разнообразия здесь включено стандартное обозначение для числа Пи (np.pi). При выводе результата мы получим число. В четвертой строке мы вызываем очень полезный метод shape, который показывает нам какого размера наш вектор v. Мы видим, что не случилось ничего неожиданного и получился вектор размерности 4.

```
[1]: import numpy as np
[2]: v = np.array([0., 1., 10., -np.pi])
[3]: v
[3]: array([ 0.        ,  1.        , 10.        , -3.14159265])
[4]: v.shape
[4]: (4,)
```

Пример: Создание трех векторов размерности 2 и их рисование с помощью библиотеки matplotlib.

В первой ячейке создаем векторы v1, v2 и v3, соответственно с числами 5 2, -3 4 и -2 e, где e — то самое число — основание натурального логарифма. Дальше с помощью библиотеки matplotlib (сначала мы ее импортируем). Дальше в нижней строке мы изображаем наш вектор на плоскости. Обратите внимание как мы обращаемся к компонентам отдельных векторов. Сначала мы обращаемся к нулевой компоненте вектора, потом к первой v1[0], v1[1]. В Python компоненты вектора нумеруются с нуля, соответственно в двумерном векторе есть компоненты номер 0 и номер 1.

```
v1 = np.array([5., 2.])
v2 = np.array([-3., 4.])
v3 = np.array([-2., -np.e])

import matplotlib.pyplot as plt

plt.figure()
plt.plot(v1[0], v1[1], 'o')
plt.plot(v2[0], v2[1], 'o')
plt.plot(v3[0], v3[1], 'o')
plt.grid()
```

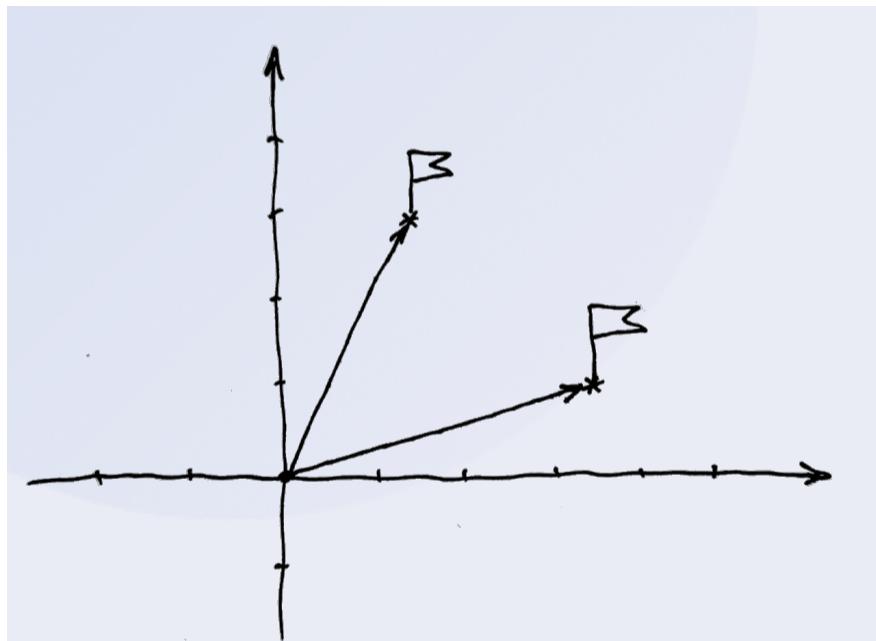
Какие основные операции с вектором?

Сложение и вычитание векторов

Когда у нас есть два вектора одной и той же размерности, то есть одной и той же длины, мы можем их складывать и вычитать поэлементно. Это аналогично тому, как мы в школе учились складывать и вычитать в столбик. Мы как будто бы пишем один вектор под другим, и тогда первая компонента нового вектора $a_1 + b_1$, вторая компонента данного вектора $a_2 + b_2$ и так далее, $a_n + b_n$.

$$\begin{aligned} &+ (a_1, \dots, a_n) \\ &(b_1, \dots, b_n) \\ &(a_1 + b_1, \dots, a_n + b_n) \end{aligned}$$

Геометрический смысл сложения векторов



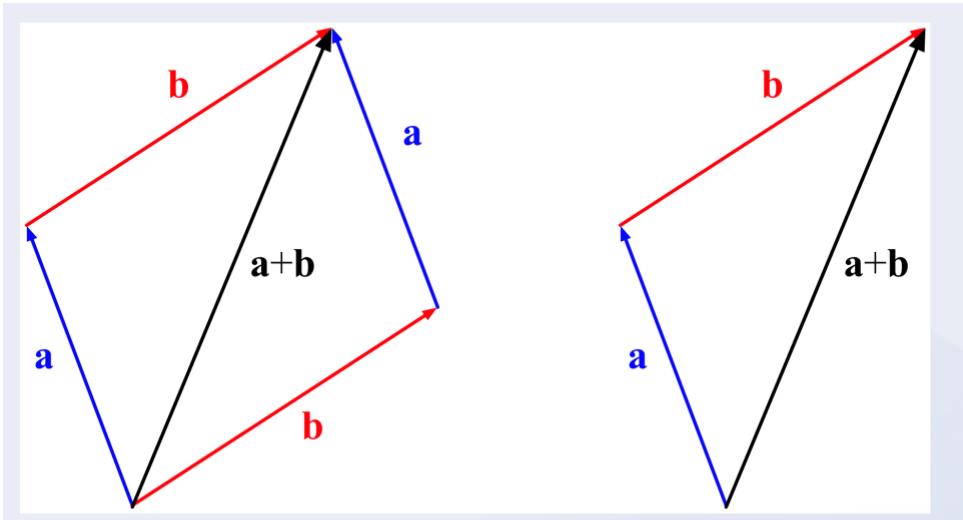
$$u_2 = (1.5, 2.0)$$

$$u_1 = (3.5, 1.0)$$

Чтобы его понять, мы вместо точек на плоскости будем представлять вектор как такую стрелочку, которая тянется из начала координат в нашу точку. Тогда сложение векторов описывается так: мы берем наш первый вектор a и к нему приставляем второй вектор b , и тогда третья сторона треугольника — это и будет вектор $a + b$.

Картина слева: мы видим еще, что вместо треугольника можно нарисовать еще параллелограмм, достроить его. Получится то же самое. Соответственно диагональ нашего параллелограмма — это и будет сумма векторов a и b .

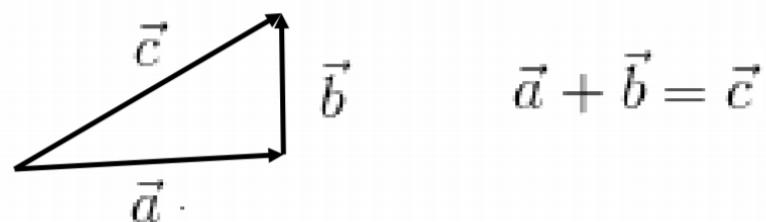
Два треугольника — слева и справа: мы видим, что $a + b = b + a$



Сложение на примере 2x векторов в 2x мерном пространстве

Вспомним, как производятся действия с векторами. Сложнее будет вектор, начало которого идет от начала вектора a и конец которого идет в конец вектора b . Вектор c . Вектор a и вектор b . Если мы складываем математически, то мы складываем координаты точек A , B и C .

Сложение векторов:



$$\vec{a} + \vec{b} = \vec{c}$$

Умножение вектора на число:



Сложить два вектора на Python: мы используем значок «+». В первой ячейке мы создали два вектора: a и b . Затем сложили и вычли их. Вычитаем также с помощью обычного «-». В конце показано, что если мы вычтем a из a , то мы получим вектор из одних нулей, так называемый «нулевой» вектор.

```
[2]: a = np.array([0., 1., 10., -np.pi])
b = np.array([-1., 5., -3., 0.])

[3]: a+b
[3]: array([-1.        ,  6.        ,  7.        , -3.14159265])

[4]: a-b
[4]: array([ 1.        , -4.        , 13.        , -3.14159265])

[5]: a-a
[5]: array([0., 0., 0., 0.]) <
```

Нулевой вектор

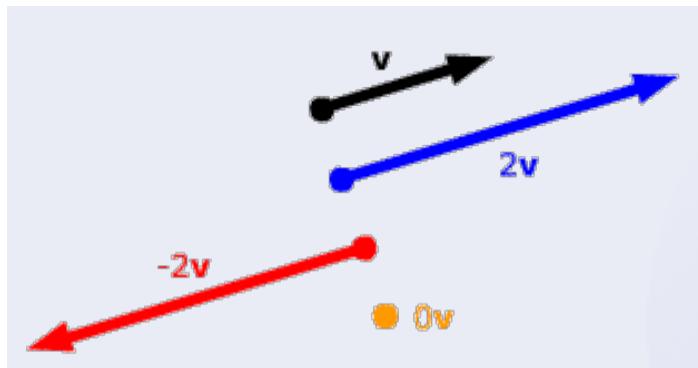
Вычитание 2x векторов в пространстве делается через сложение. Выполняем операцию умножение вектора на скаляр. Складываем вектор a и вектор $-b$. От начала вектора a строим вектор $-b'$. Далее у нас идет сложение 2x векторов. Из начала вектора a в конец вектора $-b'$ идет вектор c .

Умножение и деление векторов на числа.

Раз векторы можно складывать, то (по аналогии с обычными числами) мы можем умножить их на числа. Это происходит по тому же принципу, что на примере. У нас было одно яблоко, второе яблоко, третье яблоко — все это означает, что 3 умножить на яблоко.

$$\begin{aligned} a &= (a_1, \dots, a_n) \\ a + a + a &= 3 \cdot a = (3a_1, \dots, 3a_n) \end{aligned}$$

Соответственно когда у нас вектор a плюс еще вектор a плюс еще вектор a , это то же самое, что 3 умножить на a . Все это работает так, как интуитивно хочется, по аналогии с этим мы можем определить умножение и деление векторов на любые числа, в том числе отрицательные, в том числе числа меньше единицы. Например на картинке мы изначально берем черный вектор v , тогда вектор $2v$ — это стрелка вдоль того же направления, только в два раза длиннее, а вектор $-2v$ — это стрелочка, направленная в противоположную сторону и той же длины, что и $2v$. $0v$ — это уже встречавшийся нам нулевой вектор, это просто как точка в начале координат.



Если мы захотим два вектора умножить друг на друга, то питону разрешает нам такое сделать, с помощью операции умножения «*». Тогда наш вектор будет умножен поэлементно, также как бы в столбик. Даже можно разделить a на b поэлементно. Но если в векторе b у нас были нули, как на примере, то питону выдаст нам предупреждение, что пришлось разделить на ноль, в результате у нас получится странный символ в четвертой компоненте $-inf$, что значит «минус бесконечность».

```
[2]: a = np.array([0., 1., 10., -np.pi])
b = np.array([-1., 5., -3., 0.])

[3]: a*b
[3]: array([-0.,  5., -30., -0.])

[4]: a/b
[4]: array([-0.        ,  0.2        , -3.33333333, -inf])
```

Результат того, что у нас в векторе b был ноль.

У поэлементного умножения векторов нет геометрического смысла.

В numpy эта операция используется для ускорения вычислений: чтобы одновременно умножить/разделить много пар чисел. Есть у этого некоторые исключения, а именно для размерности 2, 3 и 4. Там есть математически стройные операции умножения. Для размерности два — это комплексные числа, для размерности три — это так называемые векторные умножения (произведения) в трехмерном пространстве и для размерности четыре — это алгебра кватернионов. Но эти все вещи скорее из области чистой математики и в Data Science особенно не нужны.

Длина вектора — это длина стрелки с начала уровня координат в ту точку в n -мерном пространстве, которая описывается компонентами вектора. Вычислить длину можно по теореме Пифагора вот по такой формуле:

$$\|a\| = \sqrt{a_1^2 + \dots + a_n^2}$$

Подсчет длины вектора

Длина вектора может быть подсчитана по формуле
евклидова расстояния:

$$\|x\| = \sqrt{\sum_i (|x_i|^2)}$$

где p — размерность вектора.

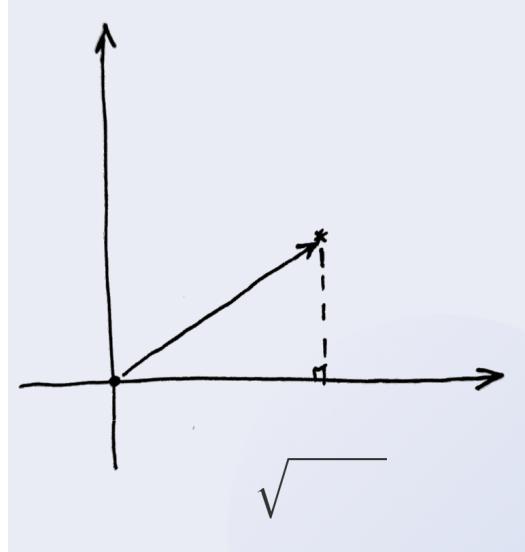
Для двумерного вектора данная формула становится следующего вида и называется L^2 нормой:

$$\|a\| = \sqrt{(x^2 + y^2)}$$

где a — вектор с координатами (x,y) .

Пример. Двумерный вектор

У нас есть вектор a , который состоит из двух компонент — a_1 и a_2 , тогда его длина — это отрезок из начала координат в точку (a_1, a_2) , а компоненты вектора — это соответственно длины двух катетов прямоугольного треугольника. И по формуле, знакомой нам со школы мы находим, что длина нашего вектора — это корень из квадратов этих двух катетов. Длина вектора еще называется Евклидовой нормой или L^2 нормой, это название особенно часто встречается в машинном обучении, когда речь идет о методах регуляризации.



$$\begin{aligned} a &= (a_1, a_2) \\ \|a\| &= \sqrt{a_1^2 + a_2^2} \end{aligned}$$

Скалярное произведение векторов.

Сначала кажется, что это довольно сложная операция, которая определяется так: нам нужно взять наши два вектора a и b , сначала их покомпонентно умножить $a_i * b_i$, а потом это все просуммировать.

π

$$(a, b) = \sum_{i=1}^n a_i \cdot b_i$$

Соответственно в итоге мы получаем некоторое одно число, которое обозначается как (a, b) , по английски еще иногда называется как dot product и обозначается либо как $a \cdot b$, либо как $\langle a, b \rangle$. На Python эта операция реализована с помощью функции dot. Загружаем библиотеку numpy, в которой как раз и есть эта функция, к векторам a и b мы вызываем эту функцию np.dot. Мы видим, что скалярное произведение наших двух векторов равно -25.

```
[1]: import numpy as np
[2]: a = np.array([0., 1., 10., -np.pi])
      b = np.array([-1., 5., -3., 0.])
      Скалярное произведение наших двух векторов равно -25.
[3]: np.dot(a, b)
[3]: -25.0
```

Зачем нужна эта сложная и с ходу какая-то не интуитивная операция? А оказывается очень полезно для самых различных расчетов с векторами. А дальше, когда у нас будут еще и матрицы, то мы увидим, что там они встречаются повсеместно.

Несколько полезных свойств скалярного произведения.

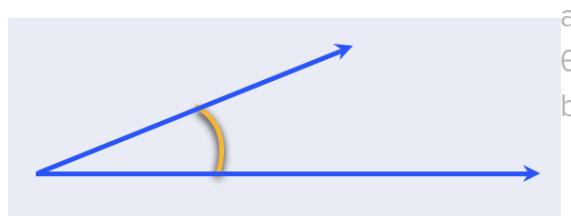
Скалярное произведение вектора на себя есть квадрат длины вектора.

$$(a, a) = \sum_{i=1}^n a_i \cdot a_i = |a|^2$$

Мы можем видеть, что если мы умножим скалярную на a , то это по определению сумма a_i , умноженная на a_i , то есть ровно квадрат его длины. Соответственно скалярное произведение мы можем использовать, чтобы вычислять длину вектора.

Для двух разных векторов a и b скалярное произведение можно использовать, чтобы измерить угол между ними. Для этого используется такое свойство скалярного произведения, что скалярное произведение любых a и b равно произведению их длин на косинус угла между ними:

$$(a, b) = |a| \cdot |b| \cdot \cos \theta$$



Соответственно, если мы вычислили скалярное произведение, мы знаем длины векторов a и b , а их мы тоже можем узнать с помощью скалярного произведения, то разделив одно на другое, мы находим косинус угла θ между нашими векторами. Полезное следствие из этого то, что векторы a и b перпендикулярны тогда и только тогда, когда их скалярное произведение равно нулю:

$$a \perp b \Leftrightarrow (a, b) = 0$$

Потому что косинус равен 0 тогда и только тогда, когда угол равен 90 градусов или -90, что то же самое.

Векторные свойства, которые помогают нам вычислять скалярные произведения. Во-первых скалярное произведение симметрично. Мы можем a и b переставить, получается то же самое.

$$(a, b) = (b, a)$$

Также оно хорошо ведёт себя со сложением/вычитанием векторов и умножением/делением на скаляры. Когда мы скалярно умножаем вектор $\lambda a + \mu b$ на вектор c , то мы можем как бы раскрыть скобки и вынести скаляры наружу. У нас будет скаляр λ , умноженный на произведение a и c плюс скаляр μ , умноженный на произведение b и c .

$$(\lambda a + \mu b, c) = \lambda(a, c) + \mu(b, c)$$

Все это помогает достаточно гибко раскладывать разные сложные скалярные произведения на маленькие компоненты, которые по отдельности уже легко посчитать.

Линейная зависимость векторов.

Линейная комбинация векторов.

Вектор v есть линейная комбинация векторов u_1, \dots, u_k , если существуют такие числа a_1, \dots, a_k , что

$$v = a_1 u_1 + \dots + a_k u_k$$

То есть вектор v — это сумма векторов u_i с коэффициентами a_i . То есть мы можем как-то так умножить вектора u_i на числа и складывать их между собой, чтобы получить вектор v .

Геометрически, если вектор v есть линейная комбинация векторов u_1, \dots, u_k , это означает, что вектор v лежит в гиперплоскости, натянутой на векторы u_1, \dots, u_k . Это можно представить, глядя на картинки ниже:

Картина слева. Вектор v лежит в плоскости, натянутой на векторы u_1 и u_2 , из этого мы делаем вывод, что вектор v — это сумма u_1 и u_2 с какими-то числами, с какими-то численным коэффициентами.

Картина справа. Вектор v пропорционален вектору u_2 , то есть v есть какое-то число, умноженное на u_2 и соответственно его можно представить как сумму u_1 и u_2 с какими-то числами, при этом u_1 будет входить с нулевым коэффициентом, что тоже считается.



Линейная независимость.

Векторы u_1, \dots, u_k линейно независимы, если единственный способ сделать нулевую линейную комбинацию из них - это взять все числа a_1, \dots, a_k равными нулю.

$$a_1u_1 + \dots + a_ku_k = 0$$

То есть иными словами как только какое-то число от 0 отличается, то мы никогда не получим 0 справа, там будет какой-то ненулевой вектор.

Например возьмем векторы:

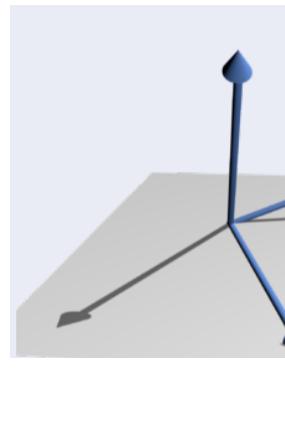
$$u_1 = (1, 0, 0)$$

$$u_2 = (0, 1, 0)$$

$$u_3 = (0, 0, 1)$$

Можно показать, что какие бы мы ни брали ненулевые коэффициенты a_1, a_2, a_3 и не складывали наши вектора с этими коэффициентами u_i , мы никогда не получим 0.

Геометрическое представление:



У нас есть векторы u_1, u_2, u_3 , и они все линейно независимы. Мы видим на этой картинке, что ни один из этих векторов не лежит в плоскости, натянутой на два других. Действительно, есть математическое утверждение, которое говорит нам о том, что

Векторы u_1, \dots, u_k линейно независимы тогда и только тогда, когда ни один из них не есть линейная комбинация других.

Базис - набор линейно независимых векторов, что к нему нельзя добавить ещё один вектор и остаться линейно независимыми. Иными словами это такой как бы максимальный набор линейно независимых векторов.

Вектора, записанные слева — это базис. Сами векторы u_1, u_2, u_3 линейно независимы. К ним нельзя добавить еще один, чтобы вся система осталась линейно независимой.

Справа у нас не базис. Хотя эти векторы тоже линейно независимы, но к ним можно добавить еще один вектор.

Это — базис:

$$u_1 = (1, 0, 0)$$

Это — не базис:

$$u_1 = (1, 0, 0, 0)$$

$$u_2 = (0, 1, 0)$$

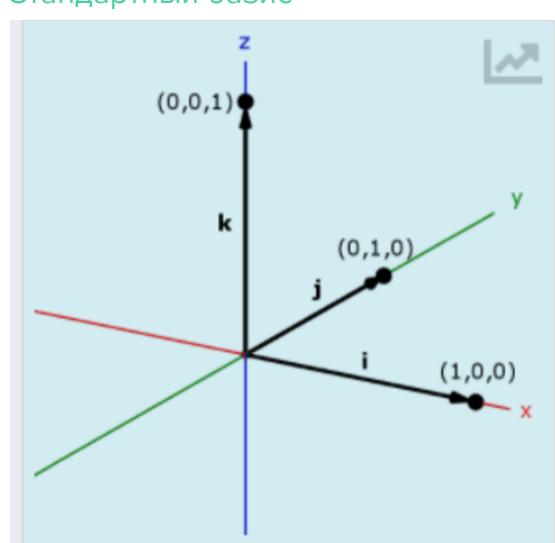
$$u_2 = (0, 1, 0, 0)$$

$$u_3 = (0, 0, 1)$$

$$u_3 = (0, 0, 1, 0)$$

Таблица слева еще называется стандартным базисом, потому что вектор u_1 — это единичный вектор по оси x, вектор u_2 — это единичный вектор по оси y и вектор u_3 — это единичный вектор по оси z.

Стандартный базис



Разложение по базису.

Базис по сути — это система координат в векторном пространстве. Разберем это подробнее.

Если векторы u_1, \dots, u_k образуют базис, то, добавив к ним любой другой вектор v , получаем линейно зависимый набор векторов. Это значит, что тот новый вектор v есть линейная комбинация векторов u_1, \dots, u_k :

$$v = a_1u_1 + \dots + a_ku_k$$

То есть вектор v можно представить как сумму векторов базиса с какими-то численными коэффициентами. Тогда числа a_1, \dots, a_k называются координатами вектора v в базисе u_1, \dots, u_k . К примеру просто стандартные компоненты вектора, которые у нас были всегда (числа в строчку, которые образуют вектор) — это на самом деле координаты вектора в стандартном базисе. Как же найти координаты вектора в каком-нибудь нестандартном базисе? Это не слишком сложная и не слишком простая задача, но ее особенно удобно и просто решать, когда у нас хороший базис. Например когда векторы базиса взаимно ортогональны и все имеют единичную длину. Это так называемый **ортонормированный** базис.

Если все векторы базиса u_1, \dots, u_k взаимно ортогональны и длины 1,

то a_1, \dots, a_k можно найти так:

$$a_i = (v, u_i)$$

a_i равно просто скалярному произведению нашего вектора v на u_i вектор базиса. Тут мы еще видим как скалярное произведение может быть нам полезно.

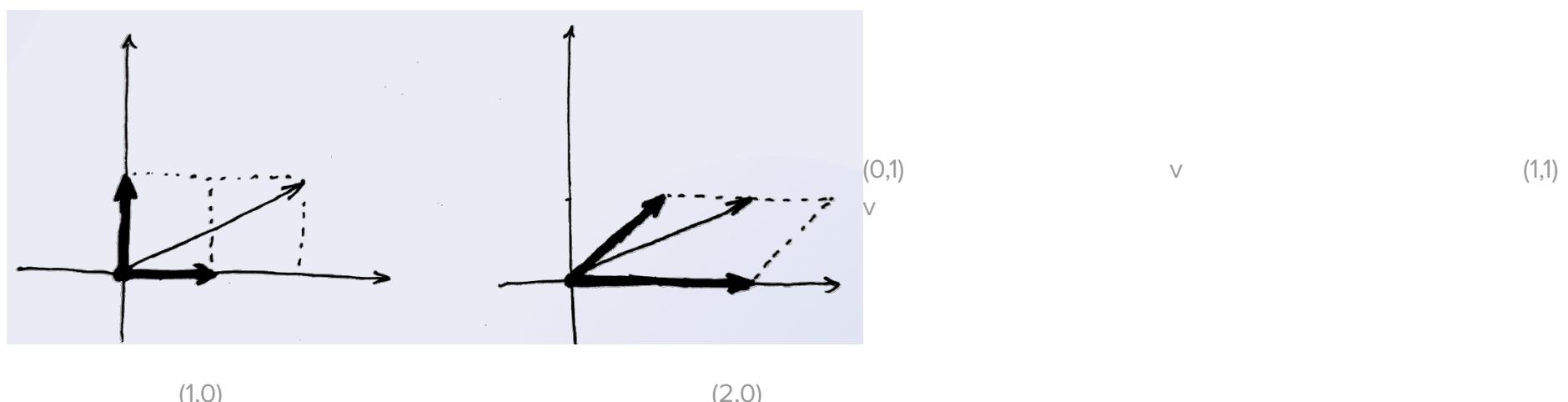
Замена координат.

Базис задает нам некоторую систему координат в пространстве векторов. И на самом деле для разных задач могут быть удобны разные базисы. У нас есть какое-то изначальное написание вектора в стандартном базисе, но может так оказаться, что есть какой-то более удобный базис, в котором например вектор записывается более компактно, у него большая часть компонент оказывается нулями, и на самом деле он живет не в каком-то огромном пространстве, как например было наше пространство картинок, а в каком-то пространстве меньшей размерности. И с помощью такого представления можно сжимать изображение, и у этого есть еще разные полезные применения.

Переход от представления вектора в одних координатах к представлению в других называется заменой координат. Сами векторы при этом не меняются, меняется только базис.

Слева у нас изображен вектор, который в стандартном базисе имеет запись $(2,1)$, что означает, что нам нужно 2 умножить на первый вектор стандартного базиса и 1 умножить на второй вектор стандартного базиса. $v = 2 \cdot (1,0) + 1 \cdot (0,1)$. А теперь мы попытаемся записать наш тот же самый вектор v , но в другом базисе, который состоит из векторов $(2,0)$ и $(1,1)$. Легко видеть, что в этом базисе вектор v имеет координаты $\frac{1}{2}$ и 1. То есть нам нужно умножить первый вектор $(2,0)$ на $\frac{1}{2}$ и добавить 1 умножить на $(1,1)$, чтобы получить тот же самый вектор v . Это пример как один и тот же вектор v может быть записан в двух разных системах координат.

Тут мы сделали замену координат вручную, а вообще замены координат удобно задавать матрицами.



$$v = 2 \cdot (1,0) + 1 \cdot (0,1)$$

$$v = 0.5 \cdot (2,0) + 1 \cdot (1,1)$$