Производная функции нескольких аргументов

Содержание:

Функция нескольких переменных.

Функция - это некоторое соответствие x -> f(x), причём для каждого x определено единственное значение f(x)

Рассмотрим случай, когда x- не число из R, а вектор (x1, x2, ..., xn), где каждое xi из R, а весь x из R^n.

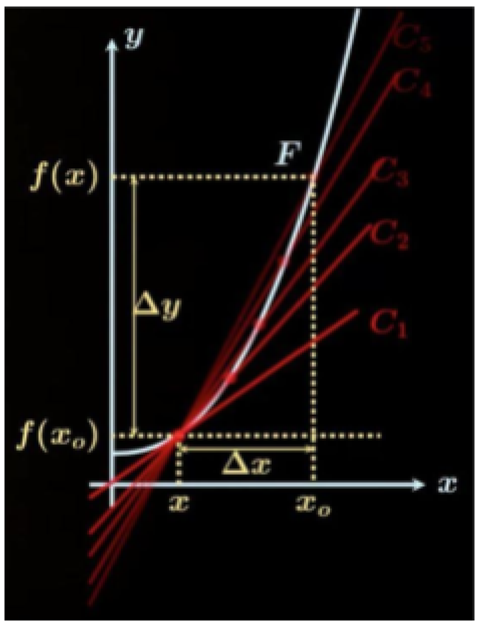
То есть, функция нескольких переменных переводит вектор в число. Например, f(x;y)=2x+y – функция двух переменных.

D(f) - область определения функции – это (как и в случае функции одного аргумента) множество, на котором задается функция. Но только теперь для каждой переменной xi, будут свои допустимые значения.

E(f) - область значений функции – множество, состоящее из всех значений, которые принимает функция.

Будем работать только с функциями, у которых D(f) и E(f) - подмножество R^n.

Геометрический смысл производной – угловой коэффициент касательной.



Как посчитать производную функции нескольких переменных?

Разберем на примере функции двух переменных (x и y).

Фиксируем переменную y , то есть считаем ее числом. И тогда вычисляем производную функции от x. как от функции с одной переменной. Тогда обозначаем такую переменную не просто ***f'***. А ***f'***$_x$, что говорит о том, что дифференцирование было именно по переменной x.

Аналогично, фиксируем ***x*** как константу, и вычисляем производную по переменной y.

$$\frac{\Delta f}{\Delta x} \xrightarrow{\Delta x \to 0} f'_x \text{ , } y \text{ — фиксирован}$$

$$\frac{\Delta f}{\Delta y} \xrightarrow{\Delta y \to 0} f'_y \text{ , } x \text{ — фиксирован}$$

Частная производная

Частная производная – это одно из обобщений понятия производной на случай функции нескольких переменных.

Частная производная функции f(x;y) по x определяется как производная по x, взятая в смысле функции одной переменной, при условии постоянства оставшейся переменной y.

В терминах пределов получим,

$$f'_x(x,y) = \lim_{h\to 0} \frac{f(x+h,y) - f(x,y)}{h}, \qquad f'_y(x,y) = \lim_{h\to 0} \frac{f(x,y+h) - f(x,y)}{h}.$$

То есть в первом случае, приращение есть только у переменной x, а y неизменный. Во втором наоборот. Таким образом, мы определяем скоростью изменения функции при изменении только лишь одной переменной. Т.е. узнаем, как влияет один признак на поведение функции.

Пусть дана некоторая функция двух переменных f : R^2 → R. Она, вообще говоря, определяет некоторую поверхность

z = f(x, y) в трехмерном пространстве.

Если в некоторой точке (x0, y0) функция дифференцируема как функция многих переменных, то в этой точке можно рассмотреть **касательную плоскость** к данной поверхности.

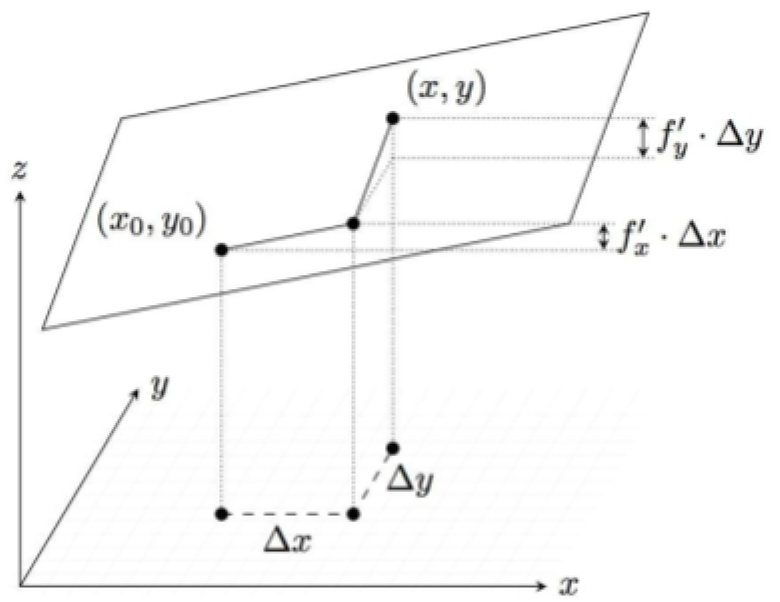


Рис. 1: Геометрический смысл частных производных.

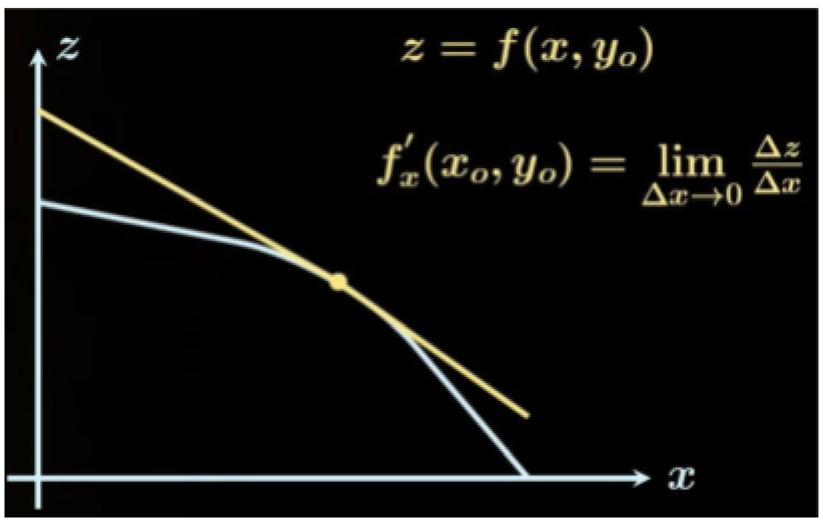
Аналогично, как в функциях одной переменной, в окрестности точки x0 функция приближается к касательной прямой (их поведение вблизи точки x0 не сильно отличаются).

Таким образом, график функции $f(x,y)$ в окрестности точки можно приблизить касательной плоскостью:

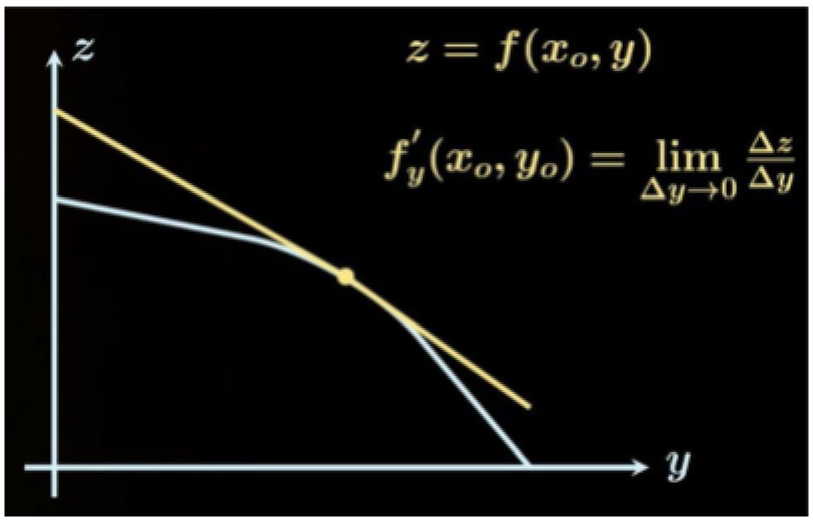
$$f(x_0+\Delta x, y_0+\Delta y) \approx f(x_0,y_0) + f'_x(x_0,y_0)\Delta x + f'_y(x_0,y_0)\Delta y.$$

То есть значение функции в какой-либо точке из окрестности (x0;y0) приблизительно равно соответствующей точке на касательной плоскости. При этом смещение по каждой переменной считается отдельно (см. рисунок выше).Приращение по x мы умножаем на частную производную по x : $f'_x\,\Delta x$, а $f'_y\,\Delta y$.

То есть мы как бы считаем, что сначала функция сместилась только по координате x , при фиксированном y0.



А потом, по координате y, при фиксированном x0.



Если функция зависит от n переменных, то мы можем найти n частных производных. Вектор, составленный из этих производных, называется градиентом функции.

То есть, если f(x1;x2;...xn) - функция n переменных, то получим n- мерный вектор из частных производных

$$\operatorname{grad} f = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n}\right)$$

Линией уровня называется множество точек, в которых функция принимает одно и то же фиксированное значение.

График функции двух переменных z=f(x;y)

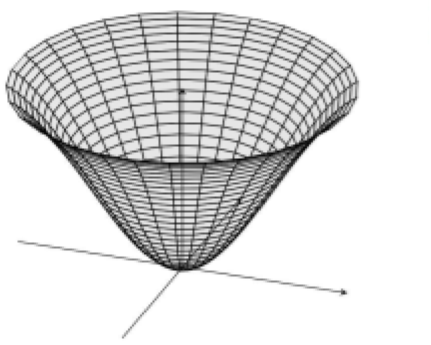


Рис. 2: Функция двух переме

Здесь линии уровня изображены кругами. Действительно, зафиксировать какое –либо значение z, значит провести через это значение плоскость, параллельно плоскости (xOy). А эта плоскость в нашем примере пересечет график функции z=f(x;y) (воронку) по кругу. То есть множество точек в нашем примере – круг. Для каждого z свой круг (на своей высоте z).
Оказывается, что градиент перпендикулярен линии уровня.
На рисунке это линии, перпендикулярные кругам.
Задачей оптимизации называется задача по нахождению экстремума функции. Например, минимума

$$f(x_1, \ldots, x_n) \to \min$$

Такая задача часто встречается в приложениях, например при выборе оптимальных параметров рекламной компании, а также в задачах классификации.

Для функции одной переменной необходимым условием было равенство производной нулю в точке. А для функции нескольких переменных будет условие равенства нулю всех ее частных производных. То есть условие равенства градиента нулевому вектору.

В нашем примере, экстремум находится в начале координат.

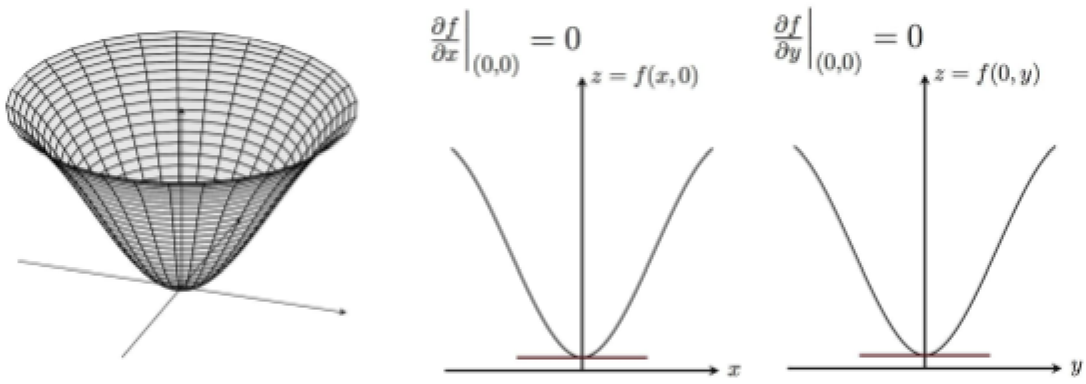


Рис. 2: Функция двух переменных достигает минимума в начале координат.

Но это только необходимое условие, но недостаточное. Более того не всегда задачу можно решить аналитически. В таком случае используется численная оптимизация. Наиболее простым для реализации из всех методов численной оптимизации является метод градиентного спуска.

Градиентный спуск – это итерационный метод. Решение задачи начинается с выбора начального приближения $\vec{x}^{[0]}$.

После вычисляется приблизительное значение $\vec{x}^1$ Затем $\vec{x}^2$ и так далее...

Делается это следующим образом.

$$\vec{x}^{[j+1]} = \vec{x}^{[j]} - \gamma^{[j]} \nabla F(\vec{x}^{[j]}), \qquad \text{где } \gamma^{[j]} \text{ — шаг градиентного спуска.}$$

а $\nabla F(\vec{x}^{[j]})$ - это градиент.

По аналогии с функцией одной переменной , где производная показывала скорость роста функции, можно сказать, что градиент также показывает как растет функция. Но так как градиент – это вектор, то он показывает еще и направление роста функции. Поэтому для нахождения минимума, надо идти по направлению антиградиента, то есть $-\nabla F$.

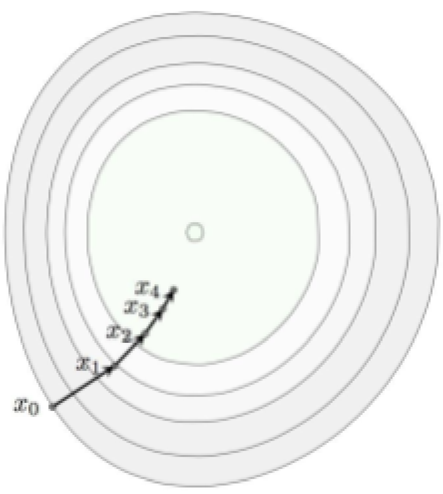


Рис. 3: Градиентный спуск

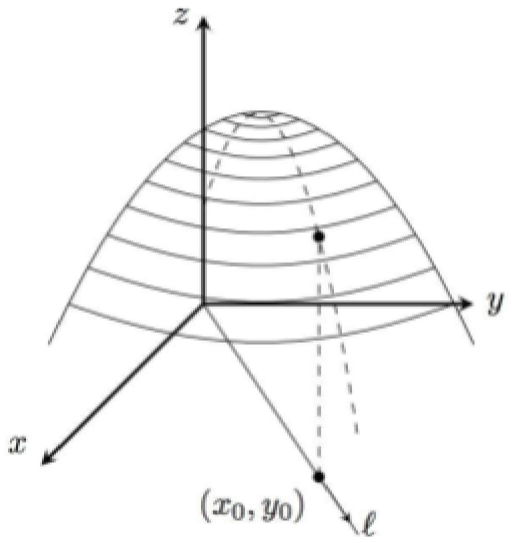
Можно провести аналогию с ситуацией, когда вы потерялись в горах, и знаете что ваш домик в самом низу. Тогда Вы должны двигаться по направлению наискорейшего спуска.

То есть вы находитесь в точке x0 (на высоте x0). Вам надо спуститься на следующую высоту. Понятно, что надо двигаться перпендикулярно линии высоты, то есть линии уровня. А это, как говорили выше, и есть направление градиента. Точнее противоположно е ему, так как спускаемся, а не поднимаемся. Мы попадем на линию уровня x1. В ней считаем градиент (то есть, определяем направление пути, перпендикулярное уже новой линии уровня) и идем в точкух2. И так далее итеративно идем к цели максимально быстро.

Производная по направлению

Пусть $f(\vec{x}) = f(x_1, x_2, \ldots, x_n)$ — функция n переменных, $\vec{\ell} \in \mathbb{R}^n$, $|\vec{\ell}| = 1$, тогда частной производной в точке x_0 по направлению $\vec{\ell}$ называется

$$\frac{\partial f}{\partial \vec{\ell}}(\vec{x}_0) = \lim_{t \to 0} \frac{f(\vec{x}_0 + t \cdot \vec{\ell}) - f(\vec{x}_0)}{t}.$$

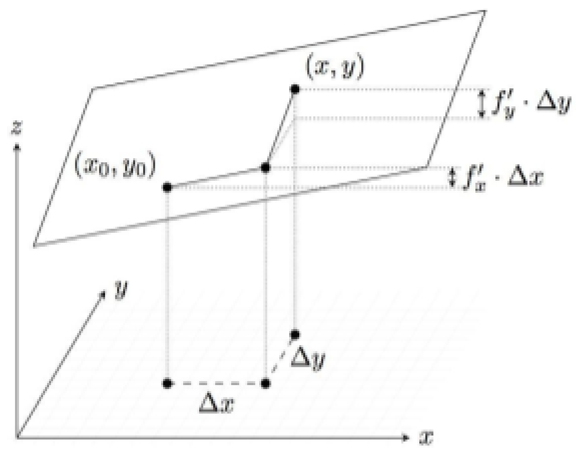


Производная по направлению показывает, насколько быстро функция изменяется при движении вдоль заданного направления.

Если провести аналогию с функцией одной переменной, то там направлением было само Δx. То есть точка смещалась на какую-то величину, и мы смотрели, как изменится от этого значение функции. Здесь функция может смещаться по каждой переменной на разную величину (Δx и Δy и т.д. разные). Получится, что точка в окрестности точки x0 перемещается в каком то заданном направлении. Функция в разные стороны растет по-разному. И чтобы оценить это изменение и нужна производная по этому направлению.

Как связаны градиент и производная по направлению?

Производная по направлению показывает изменение по всем координатам сразу. То есть точка (x0;y0) из примера сразу перемещается в (x;y).



Если мы разложим перемещение по каждой координате по отдельности, спроецируем вектор l на орты то, по сути получим частные производные , то есть вектор градиент.

Таким образом, производную по направлению дифференцируемой по совокупности переменных функции можно рассматривать как проекцию градиента функции на это направление, или иначе, как скалярное произведение градиента на орт направления:

$$\frac{\partial f}{\partial e} = \nabla f \cdot \vec{e}$$

Отсюда следует, что максимальное значение в точке производная по направлению принимает, если направление совпадает с направлением градиента функции в данной точке.

Градиент - направление роста функций перпендикулярен линиям уровня.

Если функция дифференцируема в (x0,y0), то в окрестности её можно приблизить линейно

Выше мы записывали это так

$$f(x_0 + \Delta x, y_0 + \Delta y) \approx f(x_0, y_0) + f'_x(x_0, y_0)\Delta x + f'_y(x_0, y_0)\Delta y.$$

В матричной форме эту запись можно записать следующим образом

$$\Delta f \approx \left\langle \nabla f(x_0, y_0), \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right\rangle$$

, то есть произведение вектора частных производных (градиента) на вектор приращений.

Пусть $\vec{\ell} \in \mathbb{R}^2$, $|\vec{\ell}| = 1$, тогда приращения можно задать вдоль вектора $\vec{\ell}$:

$$\Delta x = t \cdot \ell_x, \qquad \Delta y = t \cdot \ell_y.$$

То есть Δx – это проекция этого вектора на ось x, а Δy – проекция на ось y.

Подставим в первое выражение

$$\Delta f \approx \left\langle \nabla f(x_0, y_0), \begin{pmatrix} t \cdot \ell_x \\ t \cdot \ell_y \end{pmatrix} \right\rangle = t \cdot \left\langle \nabla f(x_0, y_0), \vec{\ell} \right\rangle$$

Тогда по определению производной по направлению получим,

$$\frac{\partial f}{\partial \vec{\ell}}(x_0, y_0) = \lim_{t \to 0} \frac{\Delta f}{t} = \left\langle \nabla f(x_0, y_0), \vec{\ell} \right\rangle$$

Таким образом, производная по направлению может быть вычислена как скалярное произведение градиента на соответствующий единичный вектор:

$$\frac{\partial f}{\partial \vec{\ell}}(x_0, y_0) = \left\langle \nabla f(x_0, y_0), \vec{\ell} \right\rangle$$

Согласно этой формуле направление максимального роста - направление задаваемое градиентом, а оно максимально при сонаправленности векторов. Ведь скалярное произведение – это произведение длин векторов на косинус угла между ними. То есть максимум будет достигаться при косинусе равном 1 , то есть угол должен быть равен нулю, что и означает сонаправленность вектора направления и градиента. Получаем, что градиент – направление наискорейшего роста функции.

Тогда и касательную плоскость и линейное приближение можно переписать в матричной форме.

Пусть функция $f(x, y)$ дифференцируема в точке (x_0, y_0), тогда в окрестности этой точки можно записать:

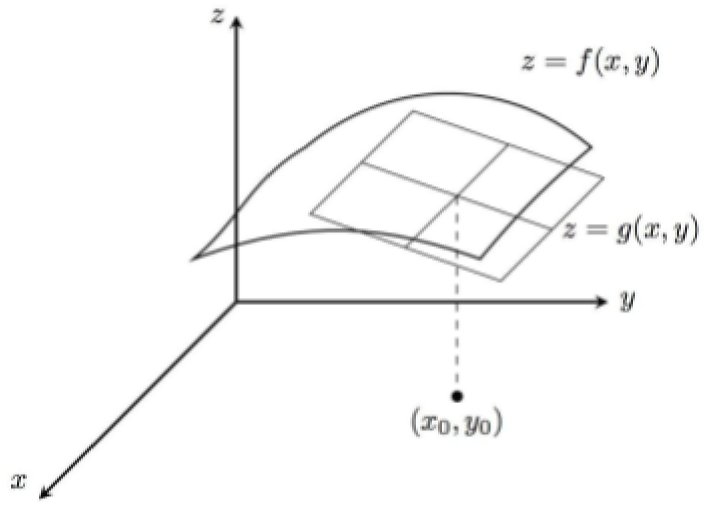
$$f(x_0 + \Delta x, y_0 + \Delta y) \approx f(x_0, y_0) + f'_x(x_0, y_0)\Delta x + f'_y(x_0, y_0)\Delta y.$$

Тогда

$$\Delta f \approx \left\langle \nabla f(x_0, y_0), \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right\rangle$$

Выражение для delta f линейно по delta x и delta y

$$f(x, y) \approx f(x_o, y_o) + \left\langle \nabla f(x_o, y_o), \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \right\rangle$$



Notebook

В библиотеке numpy есть функция gradient(x), но что она считает не очень понятно, так что ей лучше не пользоваться

Вычисление частных производных

Разберем на примере $f=1*x_1^2+2*x_2^2$

```
In [4]:def func(x, c0, c1):
    "Coordinate vector `x` should be an array of size two."
    return c0 * x[0]**2 + c1 * x[1]**2
```

вычислим градиент в точке (1;1)

```
In [5]:x = np.ones(2)
```

есть встроенная функция , импортируем ее

```
from scipy.optimize import approx_fprime
```

и получаем

approx_fprime(x, func, [eps, eps], c0, c1) – *первая координата – точка в которой вычисляем градиент, вторая – наша фунция, третья – это погрешность вычисления, и коэффициенты из функции.*

целиком выглядит так:

```
In [4]:def func(x, c0, c1):
    "Coordinate vector `x` should be an array of size two."
    return c0 * x[0]**2 + c1 * x[1]**2
In [5]:x = np.ones(2)
c0, c1 = (1, 2)
eps = np.sqrt(np.finfo(float).eps)
approx_fprime(x, func, [eps, eps], c0, c1)
Out[5]:array([2.        , 4.00000003])
```

Вычисление частных производных (аналитически по формулам)
Хотим посчитать частные производные функции $f(x;y)= x^3y-x^2y^2+x-1$ в точке (1, 1)

f'_x=3(x^2)y-2x(y^2)+1

f'_y= (x^3)-2(x^2)y

f'x(1;1)=3-2+1=2

f'_y(1;1)= 1-2=-1

$grad_f$(1;1)=(2;-1)

Как найти минимум функции нескольких переменных

Пусть дана какая то функция от двух переменных . Здесь такая

```
In [6]:def rosen(x):
    return sum(100.0*(x[1:]-x[:-1]**2.0)**2.0 + (1-x[:-1])**2.0)
```

и начальная точка x0

```
In [7]:x0 = np.array([1.3, 5])
```

Импортируем функцию **from scipy.optimize import** minimize

minimize(rosen, x0, method='nelder-mead', options={'xtol': 1e-8, 'disp': **True**}):*первая координата – функция, вторая – начальная точка, третья – мето, с помощью которого выполняется поиск минимума (можно всегда использовать такой как здесь), четвертая – параметры оптимизации (на них не смотрим)*

Optimization terminated successfully.
Current function value: 0.000000 *- вот он выдал результат, что минимум 0*
Iterations: 106
Function evaluations: 199

Целиком выглядит так:

Минимизация функций

```
In [6]:def rosen(x):
    return sum(100.0*(x[1:]-x[:-1]**2.0)**2.0 + (1-x[:-1])**2.0)
In [7]:x0 = np.array([1.3, 5])
In [8]:res = minimize(rosen, x0, method='nelder-mead', options={'xtol': 1e-8, 'disp': True})
Optimization terminated successfully.
        Current function value: 0.000000
        Iterations: 106
        Function evaluations: 199
```

Посчитаем минимумы еще нескольких функций

```
In [9]:def f1(x):
    return (x[0]**2) + (x[1]**2)
In [10]:res = minimize(f1, [2,5], method='nelder-mead', options={'xtol': 1e-6, 'disp': True})
Optimization terminated successfully.
        Current function value: 0.000000
        Iterations: 64
        Function evaluations: 121
In [11]:def f2(x):
    return 2**(x[0]**2) + (x[1]**2) - 5
In [12]:res = minimize(f2, [2,5], method='nelder-mead', options={'xtol': 1e-6, 'disp': True})
Optimization terminated successfully.
        Current function value: -4.000000
        Iterations: 59
        Function evaluations: 115
```