Metody optymalizacji

Autorzy: Konrad Kowalczyk 249263, Paweł Ptaszyński 249526

Prowadzący: dr inż. Piotr Lechowicz

1. **Cel projektu**

Celem projektu jest porównanie wybranych metod poszukiwania minimum funkcji ciągłej oraz zbadanie wpływu różnych parametrów metod na ich skuteczność i czas działania.

Projekt jest implementowany z użyciem narzędzi :

Python 3.9

NumPy 1.22.2

MatplotLib 3.5.1

1. **Funkcja Qing’a**

Matematyczna definicja

Równanie 1 Ogólny zapis funkcji Qing'a

Dziedzina wejściowa

Funkcja może być zdefiniowana w dowolnej dziedzinie wejściowej, ale zwykle jest obliczana na xi∈[-500,500] dla i= 1,...,n. W tym przypadku n = 2.

Implementacja:

*import* numpy *as* np  
*import* matplotlib.pyplot *as* plt  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 fig = plt.figure()  
 ax = fig.gca(projection='3d')  
 x = np.arange(-500, 500, 0.1)  
 y = np.arange(-500, 500, 0.1)  
 x, y = np.meshgrid(x, y)  
 res = 0  
 *for* i *in* range(3):  
 res = res + ((x \*\* 2 + y \*\* 2) - i) \*\* 2  
  
 ax.plot\_surface(x, y, res, cmap='jet')  
 plt.xlabel('x')  
 plt.ylabel('y')  
 plt.title('Qing function surface chart')  
 plt.show()  
 plt.contour(x, y, res)  
 plt.xlabel('x')  
 plt.ylabel('y')  
 plt.title('Qing function contour chart')

plt.show()

Kod 1 Implementacja funkcji Qing'a z wykorzystaniem środowiska Python oraz modułów numpy oraz matplotlib

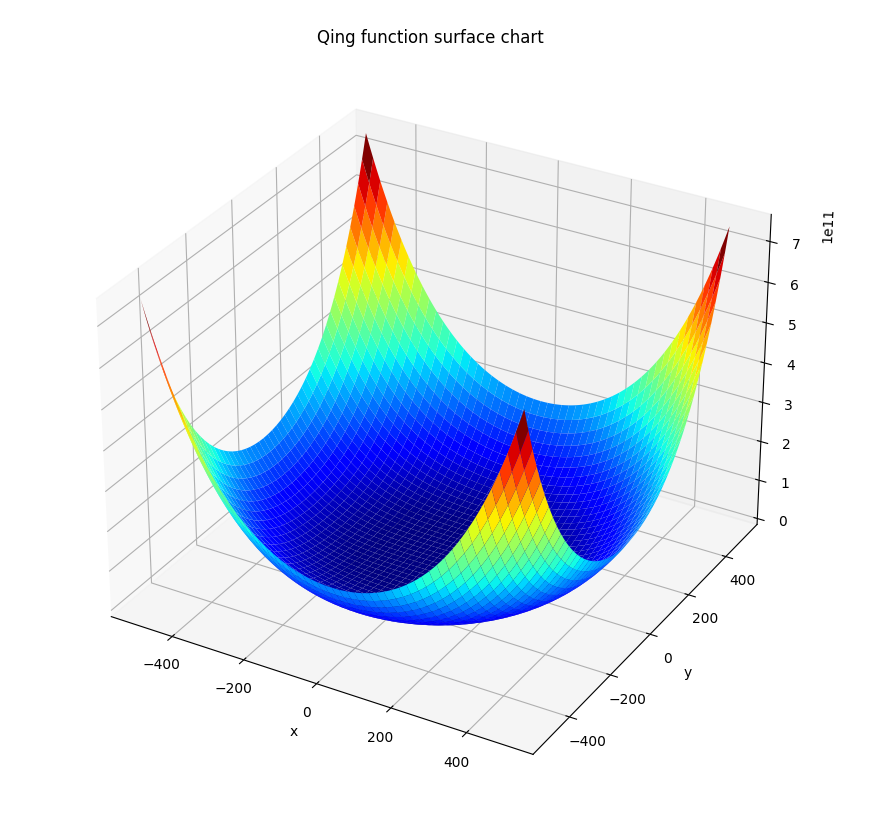
Minima globalne

Globalne minima f(x\*)=0 znajdują się w punktach x\*=(±√i,...,±√i)

Opis i cechy

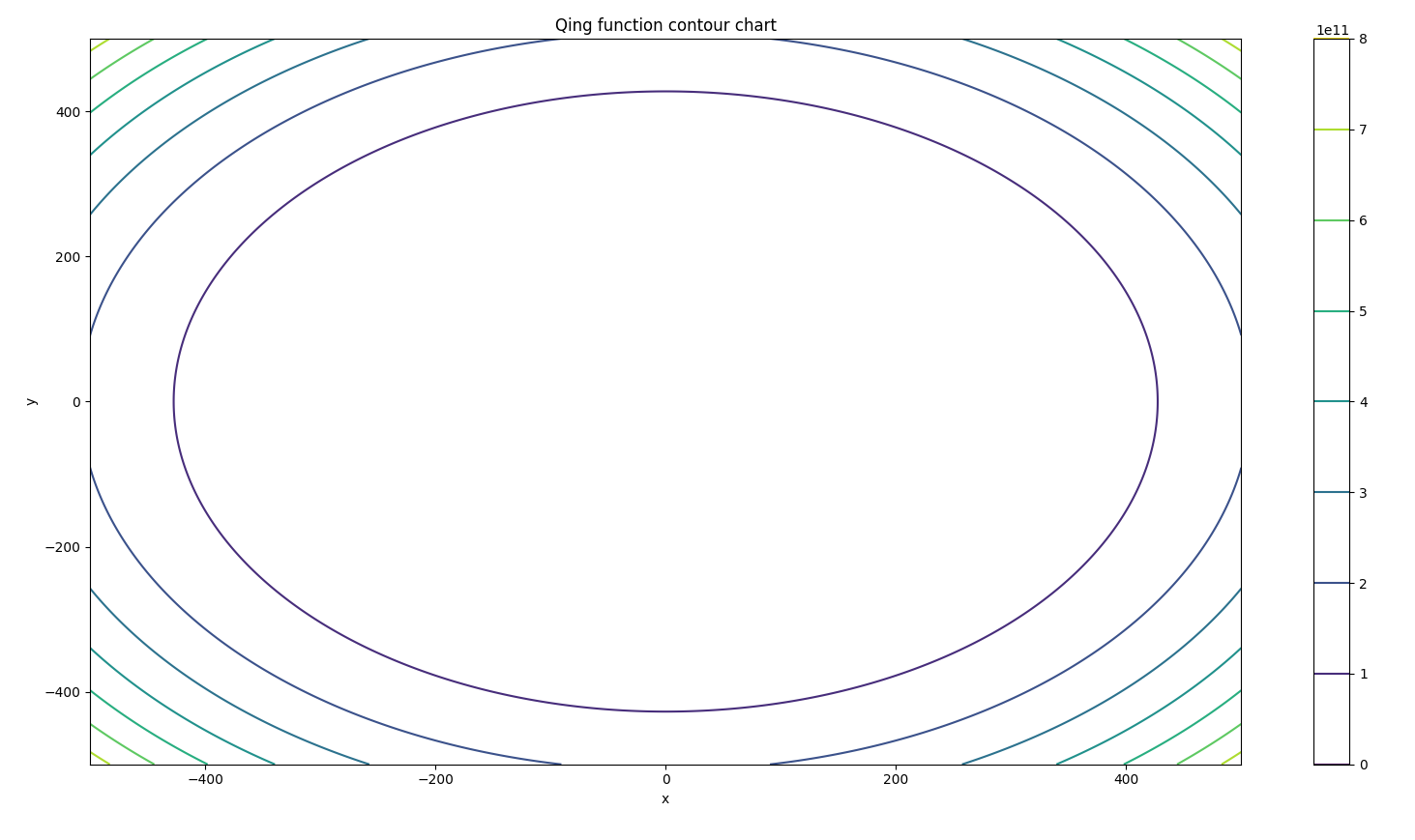
Funkcja jest ciągła, nie wypukła, określona na przestrzeni n-wymiarowej, multimodalna, różniczkowalna, nierozróżnialna

Wykres powierzchniowy



Wykres 1 Powierzchniowy wykres funkcji Qing'a wygenerowany w środowisku matplotlib

Wykres konturowy



Wykres 2 Konturowy wykres funkcji Qing'a wygenerowany w środowisku matplotlib

1. Funkcja alpejska

Matematyczna definicja

Równanie 2 Ogólny zapis funkcji Alpejskiej

Równanie 3 Ograniczenia funkcji Alpejskiej

Implementacja:

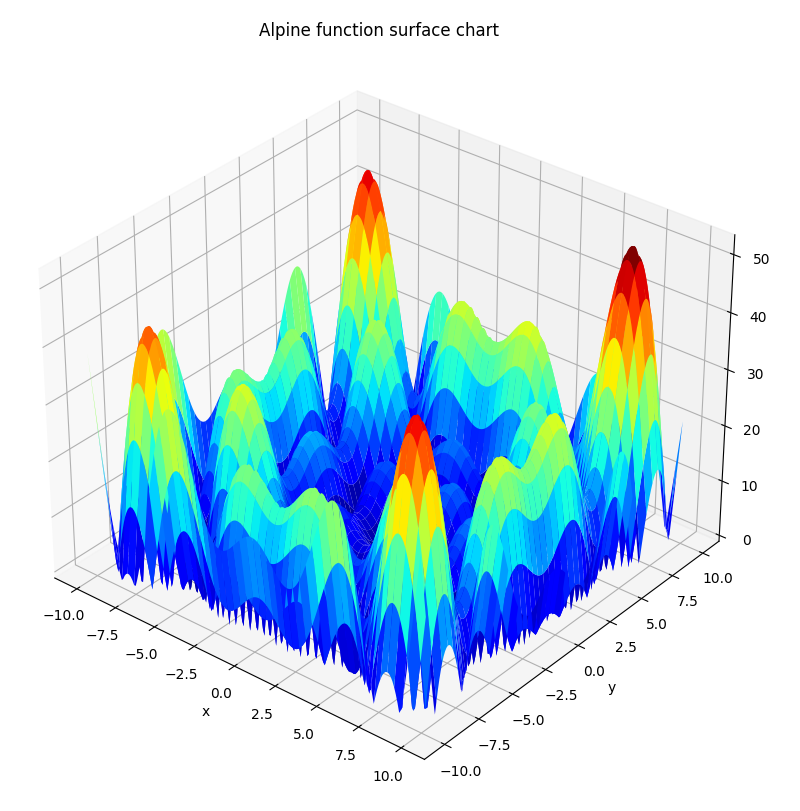
*import* numpy *as* np  
*import* matplotlib.pyplot *as* plt  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 fig = plt.figure()  
 ax = fig.gca(projection='3d')  
 x = np.arange(-10, 10, 0.1)  
 y = np.arange(-10, 10, 0.1)  
 x, y = np.meshgrid(x, y)  
 res = 0  
 *for* i *in* range(3):  
 res = res + np.abs(x \* np.sin(x) + 0.1 \* x + y \* np.sin(y) + 0.1 \* y)  
 *# print(f'{res}')* ax.plot\_surface(x, y, res, cmap='jet')  
 plt.xlabel('x')  
 plt.ylabel('y')  
 plt.title('Alpine function surface chart')  
 plt.show()  
 plt.contour(x, y, res)  
 plt.xlabel('x')  
 plt.ylabel('y')  
 plt.title('Alpine function contour chart')  
 plt.show()

Kod 2 Implementacja funkcji Alpejskiej z wykorzystaniem środowiska Python oraz modułów numpy oraz matplotlib

Funkcja ta jest interesująca do badania poszukiwania ekstremum z następujących powodów:

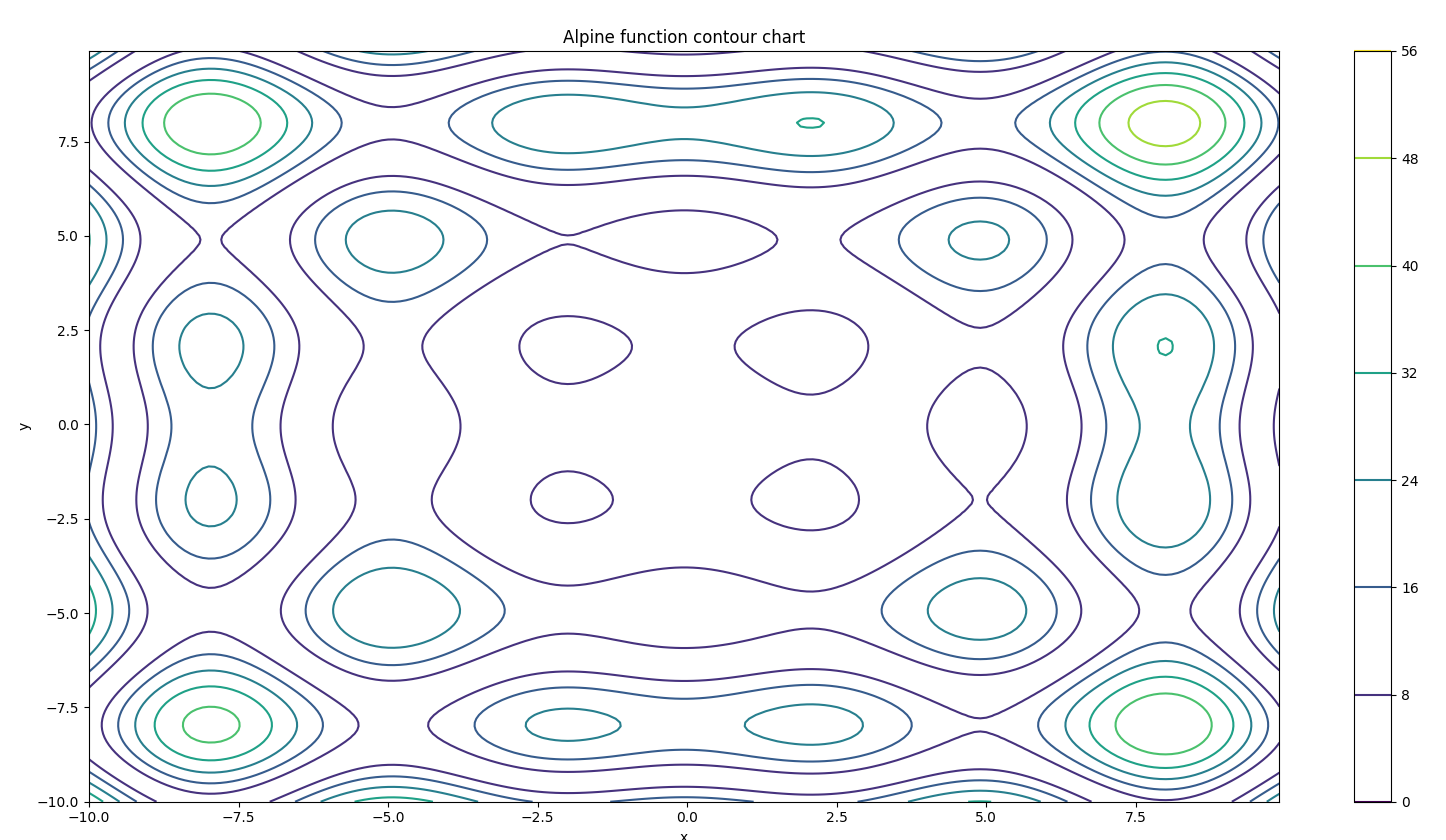
* istnieje tyle ekstremów lokalnych, ile chcemy, wystarczy zwiększyć xmax,
* istnieje tylko jedna ekstrema globalna,
* rozwiązanie można łatwo obliczyć bezpośrednio.

Wykres powierzchniowy



Wykres 3 Powierzchniowy wykres funkcji Alpejskiej wygenerowany w środowisku matplotlib

Wykres konturowy



Wykres 4 Konturowy wykres funkcji Alpejskiej wygenerowany w środowisku matplotlib

Link GitHub:  
https://github.com/kondiiq/Optimization-with-gradient-and-no-gradient