|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Лабораторная работа № 2 | | |
| по дисциплине «Методы оптимизации» | | |
| МЕТОДЫ СПУСКА (0-го, 1-го и 2-го ПОРЯДКАИ ПЕРЕМЕННОЙ МЕТРИКИ) | | |
|  | | |
|  | Бригада 1 | вострецова екатерина |
| Группа ПМ-13 | исакин даниил |
| Вариант 1 |  |
|  |  |
|  |  |
|  |  |
| Преподаватели | Филиппова Елена Владимировна |
|  |  |
| Новосибирск, 2024 | | |

1. **Цель**

Ознакомиться с методами поиска минимума функции n переменных в оптимизационных задачах без ограничений.

1. **Задание**

Реализовать два метода поиска экстремума функции (разного порядка). Включить в реализуемый алгоритм собственную процедуру, реализующую одномерный поиск по направлению. Методы поиска для самостоятельной реализации выбираются студентом в зависимости от уровня сложности. Выбранные методы должны иметь разный порядок.

С использованием разработанного программного обеспечения исследовать алгоритмы на квадратичной функции , функции Розенброка  и на заданной в соответствии с вариантом тестовой функции, осуществляя спуск из различных исходных точек (не менее двух). Исследовать сходимость алгоритма, фиксируя точность определения минимума/максимума, количество итераций метода и количество вычислений функции в зависимости от задаваемой точности поиска. Результатом выполнения данного пункта должны быть выводы об объеме вычислений в зависимости от задаваемой точности и начального приближения.

Построить траекторию спуска различных алгоритмов из одной и той же исходной точки с одинаковой точностью. В отчете наложить эту траекторию на рисунок с линиями равного уровня заданной функции.

Реализовать метод квадратичной интерполяции (метод парабол) для приближенного нахождения экстремума при одномерном поиске. Исследовать влияние точности одномерного поиска на общее количество итераций и вычислений функции при разных методах одномерного поиска.

Найти максимум заданной функции: 

1. **Результаты исследования**

Метод наискорейшего спуска

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Начальное приближение x0 | Точность по функции | Точность по переменным | Количество итераций | Число вычислений целевой функции | Точка минимума | Значение функции в точке минимума |
| (5.0000000e+00, 1.0000000e+01) | 1.0000000e-03 | 1.0000000e-03 | 4 | 114 | (1.1696674e+00,1.1728937e+00) | 2.9827947e-02 |
| 1.0000000e-04 | 1.0000000e-04 | 8 | 227 | (1.0378030e+00,1.0387034e+00) | 1.5101471e-03 |
| 1.0000000e-05 | 1.0000000e-05 | 6 | 224 | (1.0007293e+00,1.0007495e+00) | 5.7272186e-07 |
| 1.0000000e-06 | 1.0000000e-06 | 7 | 308 | (1.0002553e+00,1.0002563e+00) | 6.5276346e-08 |
| 1.0000000e-07 | 1.0000000e-07 | 7 | 357 | (1.0000408e+00,1.0000410e+00) | 1.6652900e-09 |
| (4.0000000e+00, 7.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 4 | 106 | (1.0761911e+00,1.0750723e+00) | 5.9302431e-03 |
| 1.0000000e-04 | 1.0000000e-04 | 8 | 231 | (1.0190218e+00,1.0199350e+00) | 4.4522080e-04 |
| 1.0000000e-05 | 1.0000000e-05 | 6 | 222 | (1.0017683e+00,1.0018107e+00) | 3.3064383e-06 |
| 1.0000000e-06 | 1.0000000e-06 | 7 | 303 | (1.0001073e+00,1.0001076e+00) | 1.1515858e-08 |
| 1.0000000e-07 | 1.0000000e-07 | 7 | 355 | (1.0000188e+00,1.0000189e+00) | 3.5518202e-10 |
| Метод Ньютона | | | | | | | |
| Начальное приближение x0 | Точность по функции | Точность по переменным | Количество итераций | Число вычислений целевой функции | Точка минимума | Значение функции в точке минимума |
| (5.0000000e+00, 1.0000000e+01) | 1.0000000e-03 | 1.0000000e-03 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-04 | 1.0000000e-04 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-05 | 1.0000000e-05 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-06 | 1.0000000e-06 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-07 | 1.0000000e-07 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| (4.0000000e+00, 7.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-04 | 1.0000000e-04 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-05 | 1.0000000e-05 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-06 | 1.0000000e-06 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| 1.0000000e-07 | 1.0000000e-07 | 1 | 5 | (1.0000000e+00,1.0000000e+00) | 0.0000000e+00 |
| Функция Розенброка | | | | | | | |
| Метод наискорейшего спуска | | | | | | | |
| Начальное приближение x0 | Точность по функции | Точность по переменным | Количество итераций | Число вычислений целевой функции | Точка минимума | Значение функции в точке минимума |
| (2.0000000e+00, 3.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 2 | 38 | (1.7483851e+00,3.0607988e+00) | 5.6163926e-01 |
| 1.0000000e-04 | 1.0000000e-04 | 32 | 616 | (1.0628757e+00,1.1295370e+00) | 3.9561749e-03 |
| 1.0000000e-05 | 1.0000000e-05 | 217 | 5892 | (1.0424041e+00,1.0867918e+00) | 1.8015473e-03 |
| 1.0000000e-06 | 1.0000000e-06 | 537 | 17614 | (1.0066471e+00,1.0133668e+00) | 4.4264320e-05 |
| 1.0000000e-07 | 1.0000000e-07 | 1000 | 37022 | (1.0051811e+00,1.0103603e+00) | 2.6926665e-05 |
| (2.0000000e+00, 1.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 44 | 583 | (6.1056366e-01,3.6930812e-01) | 1.5287161e-01 |
| 1.0000000e-04 | 1.0000000e-04 | 3 | 69 | (1.1049098e+00,1.2212722e+00) | 1.1026009e-02 |
| 1.0000000e-05 | 1.0000000e-05 | 114 | 2678 | (1.0340832e+00,1.0691727e+00) | 1.1640717e-03 |
| 1.0000000e-06 | 1.0000000e-06 | 407 | 11430 | (1.0116213e+00,1.0234294e+00) | 1.3532214e-04 |
| 1.0000000e-07 | 1.0000000e-07 | 746 | 25193 | (1.0029717e+00,1.0059340e+00) | 8.8637347e-06 |
| Метод Ньютона | | | | | | | |
| Начальное приближение x0 | Точность по функции | Точность по переменным | Количество итераций | Число вычислений целевой функции | Точка минимума | Значение функции в точке минимума |
| (2.0000000e+00, 3.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 7 | 13370 | (1.0017793e+00,1.0037548e+00) | 6.8879281e-06 |
| 1.0000000e-04 | 1.0000000e-04 | 8 | 13429 | (9.9997949e-01,9.9996104e-01) | 8.4763008e-10 |
| 1.0000000e-05 | 1.0000000e-05 | 8 | 13429 | (9.9997949e-01,9.9996104e-01) | 8.4763008e-10 |
| 1.0000000e-06 | 1.0000000e-06 | 9 | 13435 | (9.9999999e-01,9.9999998e-01) | 4.1582144e-16 |
| 1.0000000e-07 | 1.0000000e-07 | 9 | 13435 | (9.9999999e-01,9.9999998e-01) | 4.1582144e-16 |
| (2.0000000e+00, 1.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 7 | 13388 | (1.0019401e+00,1.0040987e+00) | 8.3792187e-06 |
| 1.0000000e-04 | 1.0000000e-04 | 8 | 13452 | (9.9997371e-01,9.9995001e-01) | 1.3557974e-09 |
| 1.0000000e-05 | 1.0000000e-05 | 8 | 13452 | (9.9997371e-01,9.9995001e-01) | 1.3557974e-09 |
| 1.0000000e-06 | 1.0000000e-06 | 9 | 13458 | (9.9999999e-01,9.9999998e-01) | 5.1875879e-16 |
| 1.0000000e-07 | 1.0000000e-07 | 9 | 13458 | (9.9999999e-01,9.9999998e-01)) | 5.1875879e-16 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Тестовая функция | | | | | | | | |
| **Метод наискорейшего спуска** | | | | | | | | |
| Начальное приближение x0 | Точность по функции | Точность по переменным | Количество итераций | Число вычислений целевой функции | Точка максимума | Значение функции в точке минимума |
| (1.2600000e+00, 1.3300000e+00) | 1.0000000e-03 | 1.0000000e-03 | 1 | 13 | (1.2626298e+00,1.3343754e+00) | 3.1693171e+00 |
| 1.0000000e-04 | 1.0000000e-04 | 1 | 13 | (1.2626298e+00,1.3343754e+00) | 3.1693171e+00 |
| 1.0000000e-05 | 1.0000000e-05 | 4 | 73 | (1.2627217e+00,1.3344412e+00) | 3.1693172e+00 |
| 1.0000000e-06 | 1.0000000e-06 | 2 | 51 | (1.2626727e+00,1.3344467e+00) | 3.1693171e+00 |
| 1.0000000e-07 | 1.0000000e-07 | 2 | 68 | (1.2626726e+00,1.3344466e+00) | 3.1693171e+00 |
| (1.5000000e+00, 0.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 3 | 71 | (1.3311639e+00,1.3143170e+00) | 3.1657330e+00 |
| 1.0000000e-04 | 1.0000000e-04 | 3 | 71 | (1.3311639e+00,1.3143170e+00) | 3.1657330e+00 |
| 1.0000000e-05 | 1.0000000e-05 | 4 | 82 | (1.3304899e+00,1.3135783e+00) | 3.1657336e+00 |
| 1.0000000e-06 | 1.0000000e-06 | 4 | 82 | (1.3304899e+00,1.3135783e+00) | 3.1657336e+00 |
| 1.0000000e-07 | 1.0000000e-07 | 11 | 132 | (1.3304940e+00,1.3135745e+00) | 3.1657330e+00 |
| Метод Ньютона | | | | | | | | |
| Начальное приближение x0 | Точность по функции | Точность по переменным | Количество итераций | Число вычислений целевой функции | Точка максимума | Значение функции в точке минимума | |
| (1.2600000e+00, 1.3300000e+00) | 1.0000000e-03 | 1.0000000e-03 | 1 | 1973 | (1.2647765e+00,1.3326518e+00) | 3.1693107e+00 | |
| 1.0000000e-04 | 1.0000000e-04 | 1 | 1973 | (1.2647765e+00,1.3326518e+00) | 3.1693107e+00 | |
| 1.0000000e-05 | 1.0000000e-05 | 2 | 3977 | (1.2647765e+00,1.3326518e+00) | 3.1693107e+00 | |
| 1.0000000e-06 | 1.0000000e-06 | 2 | 3777 | (1.2647765e+00,1.3326518e+00) | 3.1693107e+00 | |
| 1.0000000e-07 | 1.0000000e-07 | 2 | 3777 | ((1.2647765e+00,1.3326518e+00)) | 3.1693107e+00 | |
| (1.5000000e+00, 0.0000000e+00) | 1.0000000e-03 | 1.0000000e-03 | 4 | 9663 | (1.0277332e+00,1.1696086e+00) | 3.1117328e+00 | |
| 1.0000000e-04 | 1.0000000e-04 | 5 | 11954 | (1.0237734e+00,1.1718803e+00) | 3.1117525e+00 | |
| 1.0000000e-05 | 1.0000000e-05 | 6 | 13866 | (1.0248622e+00,1.1712630e+00 | 3.1117540e+00 | |
| 1.0000000e-06 | 1.0000000e-06 | 7 | 15894 | (1.0245652e+00,1.1714319e+00) | 3.1117541e+00 | |
| 1.0000000e-07 | 1.0000000e-07 | 8 | 17891 | (1.0246508e+00,1.1713833e+00) | 3.1117541e+00 | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Квадратичная функция , точность поиска по переменным и функции начальная точка (5.0000000e+00, 1.0000000e+01) | | | | | | | | | |
| Метод наискорейшего спуска | | | | | | | | | |
| Iter | (x, y) | f(x, y) | (s1, s2) | lambda | |x(i) - x(i-1)|  |y(i) - y(i-1)|  |f(i) - f(i-1)| | grad(x, y) |  | |
| 1 | (7.4739339e+00, 7.5061150e+00) | 4.2015383e+01 | (2.0475000e+00, 8.1915000e+00) | 3.5128068e+00 | 2.4739339e+00  2.4938850e+00  2.4739846e+03 | (-9.9200000e+02, 1.0000000e+03) |  | |
| 2 | (1.1224225e+00, 1.2281862e+00) | 1.1335845e+00 | (4.0955000e+00, 1.6383500e+01) | 8.9305144e+00 | 3.8775775e+00  8.7718138e+00  4.0881798e+01 | (6.5116529e+00, 6.4362150e+00) |  | |
| 3 | (1.1747180e+00, 1.1752783e+00) | 3.0557761e-02 | (3.1500000e-02, 1.2750000e-01) | 7.4391303e-02 | 3.8252820e+00  8.8247217e+00  1.1030267e+00 | (-2.0907907e+01, 2.1152752e+01) |  | |
| 4 | (1.1696674e+00, 1.1728937e+00) | 2.9827947e-02 | (3.5000000e-03, 1.5500000e-02) | 5.5851449e-03 | 3.8303326e+00  8.8271063e+00  7.2981389e-04 | (2.3736584e-01, 1.1207006e-01) |  | |
| Метод Ньютона | | | | | | | | | |
| Iter | (x, y) | f(x, y) | (s1, s2) | lambda | |x(i) - x(i-1)|  |y(i) - y(i-1)|  |f(i) - f(i-1)| | grad(x, y) | Hesse(dx2, dxdy, dydx, dy2) |
| 1 | (1.0000000e+00, 1.0000000e+00) | 0.0000000e+00 | (-4.0000000e+00, -9.0000000e+00) | 1.0000000e+00 | 4.0000000e+00  9.0000000e+00  2.5160000e+03 | (-9.9200000e+02, 1.0000000e+03) | (5.0000000e-01, 5.0000000e-01, 5.0000000e-01, 5.0500000e-01) |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Функция Розенброка , точность поиска по переменным и функции  начальная точка (2.0000000e+00, 3.0000000e+00)  Метод наискорейшего спуска | | | | | | | | |
| Iter | (x, y) | f(x, y) | (s1, s2) | lambda | |x(i) - x(i-1)|  |y(i) - y(i-1)|  |f(i) - f(i-1)| | grad(x, y) |  | |
| 1 | (1.7494767e+00, 3.0624746e+00) | 5.6204146e-01 | (1.2750000e-01, 5.1150000e-01) | 2.5819564e-01 | 2.5052327e-01  6.2474631e-02  1.0043796e+02 | (8.0200000e+02, -2.0000000e+02) |  | |
| 2 | (1.7483851e+00, 3.0607988e+00) | 5.6163926e-01 | (5.0000000e-04, 3.5000000e-03) | 2.0000000e-03 | 2.5161493e-01  6.0798838e-02  4.0219642e-04 | (2.3527003e-01, 3.6116040e-01) |  | |
| Метод Ньютона | | | | | | | | |
| Iter | (x, y) | f(x, y) | (s1, s2) | lambda | |x(i) - x(i-1)|  |y(i) - y(i-1)|  |f(i) - f(i-1)| | grad(x, y) | Hesse(dx2, dxdy, dydx, dy2) |
| 1 | (1.9983361e+00, 3.9933444e+00) | 9.9667498e-01 | (-1.6638935e-03, 2.9933444e+00) | 1.0000000e+00 | 1.6638935e-03  2.9933444e+00  9.0000333e+02 | (2.4020000e+03, -6.0000000e+02) | (8.3194676e-04, 3.3277870e-03, 3.3277870e-03, 1.8311148e-02) |
| 2 | (1.8376929e+00, 3.3513068e+00) | 7.6833759e-01 | (-1.6230706e-01, 2.3513068e+00) | 1.6100000e-01 | 1.6230706e-01  2.3513068e+00  2.2833740e-01 | (1.9988852e+00, -5.5370832e-04) | (4.9972330e-01, 1.9972302e+00, 1.9972302e+00, 7.9872745e+00) |
| 3 | (1.6329502e+00, 2.6376658e+00) | 4.8391864e-01 | (-3.6704983e-01, 1.6376658e+00) | 1.5060000e+00 | 3.6704983e-01  1.6376658e+00  2.8441895e-01 | (2.0646661e+01, -5.1617098e+00) | (8.1146308e-02, 2.9824400e-01, 2.9824400e-01, 1.1011618e+00) |
| 4 | (1.3693804e+00, 1.8582598e+00) | 1.6514819e-01 | (-6.3061956e-01, 8.5825985e-01) | 2.8200000e+00 | 6.3061956e-01  8.5825985e-01  3.1877044e-01 | (2.0116990e+01, -5.7720958e+00) | (7.3832387e-02, 2.4112922e-01, 2.4112922e-01, 7.9250399e-01) |
| 5 | (1.1924584e+00, 1.4093267e+00) | 5.2992691e-02 | (-8.0754162e-01, 4.0932667e-01) | 2.1020000e+00 | 8.0754162e-01  4.0932667e-01  1.1215550e-01 | (1.0019288e+01, -3.3885860e+00) | (1.1393191e-01, 3.1203227e-01, 3.1203227e-01, 8.5958176e-01) |
| 6 | (1.0229549e+00, 1.0442985e+00) | 9.8416342e-04 | (-9.7704505e-01, 4.4298517e-02) | 3.1055000e+00 | 9.7704505e-01  4.4298517e-02  5.2008527e-02 | (6.4093626e+00, -2.5260613e+00) | (1.4180128e-01, 3.3818425e-01, 3.3818425e-01, 8.1154127e-01) |
| 7 | (1.0019401e+00, 1.0040987e+00) | 8.3792187e-06 | (-9.9805993e-01, 4.0987314e-03) | 1.3070000e+00 | 9.9805993e-01  4.0987314e-03  9.7578421e-04 | (9.2086489e-01, -4.2766058e-01) | (3.5022330e-01, 7.1652532e-01, 7.1652532e-01, 1.4709462e+00) |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Тестовая функция, точность поиска по переменным и функции , начальная точка (1.2600000e+00, 1.3300000e+00) | | | | | | | | |
| Метод наискорейшего спуска | | | | | | | | |
| Iter | (x, y) | f(x, y) | (s1, s2) | lambda | |x(i) - x(i-1)|  |y(i) - y(i-1)|  |f(i) - f(i-1)| | grad(x, y) |  | |
| 1 | (1.2626298e+00, 1.3343754e+00) | 3.1693171e+00 | (1.5000000e-03, 7.5000000e-03) | 5.1048784e-03 | 2.6297638e-03  4.3754001e-03  2.4286208e-05 | (2.2926571e-01, 3.8145221e-01) |  | |
| Метод Ньютона | | | | | | | | |
| Iter | (x, y) | f(x, y) | (s1, s2) | lambda | |x(i) - x(i-1)|  |y(i) - y(i-1)|  |f(i) - f(i-1)| | grad(x, y) | Hesse(dx2, dxdy, dydx, dy2) |
| 1 | (1.2647765e+00, 1.3326518e+00) | 3.1693107e+00 | (4.7764901e-03, 2.6517760e-03) | 1.6000000e-02 | 4.7764901e-03  2.6517760e-03  1.7876497e-05 | (2.2926571e-01, 3.8145221e-01) | (1.1861104e+00, 6.9723510e-02, 6.9723510e-02, 3.9258074e-01) |

1. **Вывод**

Метод наискорейшего спуска обладает линейной скоростью сходимости, Метод Ньютона – квадратичной. С повышением точности, результат работы программы становится близким к истинному, что влечет за собой увеличение количества итераций и числа вычислений функции. Следует отметить, что в случае квадратичной функции метод Ньютона находит экстремум за одну итерацию. Упомянем, что метод наискорейшего спуска может иметь трудности в патологических случаях овражных функций, так, к примеру, в случае функции Розенброка. Нахождение матрицы Гессе связано с большими вычислительными затратами, и поэтому Метод Ньютона сложнее и затратнее Метода наискорейшего спуска.

1. **Текст программы**

*File - Source.cpp*

#include "Function.h"

#include <fstream>

#include <iomanip>

using namespace std;

ofstream fout;

//Интервал, содержащий минимум функции

int IntervalMinimumFunction(method &md, double x0, int &countf)

{

double x1, x01, x2, f0, f01, f1, f2, h = e / 2;

int count = 1;

bool flag = true;

f0 = LambdaFunсtion(md.point, md.grad, x0, countf);

x1 = x0 + h;

f1 = LambdaFunсtion(md.point, md.grad, x1, countf);

if (f0 > f1)

x1 = x0 + h;

else

{

x01 = x0 - h;

f01 = LambdaFunсtion(md.point, md.grad, x01, countf);

if (f01 > f0)

{

md.a = x01;

md.b = x1;

return 1;

}

h \*= -1;

}

f1 = LambdaFunсtion(md.point, md.grad, x1, countf);

while (flag)

{

h \*= 2;

x2 = x1 + h;

f2 = LambdaFunсtion(md.point, md.grad, x2, countf);

if (f1 > f2)

{

x0 = x1;

f0 = f1;

x1 = x2;

f1 = f2;

count++;

}

else

flag = false;

}

if (x2 < x0)

{

double t = x2;

x2 = x0, x0 = t;

}

md.a = x0;

md.b = x2;

return 1;

}

//Метод золотого сечения для решения одномерной задачи оптимизации

int GoldenRatioMethod(method &md, int &countf)

{

const double c1 = (3 - sqrt(5.)) / 2;

double x1, x2, f1, f2, a = md.a, b = md.b;

int count = 1, k;

x1 = a + c1 \* (b - a);

x2 = b - c1 \* (b - a);

f1 = LambdaFunсtion(md.point, md.grad, x1, countf);

f2 = LambdaFunсtion(md.point, md.grad, x2, countf);

while (abs(b - a) > e)

{

if (f1 > f2)

{

a = x1;

x1 = x2;

f1 = f2;

k = 0;

}

else

{

b = x2;

x2 = x1;

f2 = f1;

k = 1;

}

if (abs(b - a) < e)

{

md.lambda = (a + b) / 2.0;

return 1;

}

if (k)

{

x1 = a + c1 \* (b - a);

f1 = LambdaFunсtion(md.point, md.grad, x1, countf);

}

else

{

x2 = b - c1 \* (b - a);

f2 = LambdaFunсtion(md.point, md.grad, x2, countf);

}

count++;

}

}

//Метод наискорейшего спуска

void SteepestDescent(double \_x, double \_y)

{

int count = 0, countf = 0; //Количество итераций

double fpred, fnext;

method sd;

sd.point.x = \_x, sd.point.y = \_y; //Начальное приближение (x, y)

coords pointpred = sd.point;

fout << scientific << setprecision(7) << "Iter\t(x, y)\t f(x, y)\t (s1, s2)\t lambda\t |x(i) - x(i-1)|\t |y(i) - y(i-1)|\t |f(i) - f(i-1)|\t grad(x, y)" << endl;

fnext = Funсtion(sd.point, countf); //Считаем значение функции

fpred = fnext;

double fend = abs(fnext - fpred), xend = abs(sd.point.x - pointpred.x), yend = abs(sd.point.y - pointpred.y);

do

{

fpred = fnext;

GradFunсtion(sd.point, sd.grad); //Считаем частные производные

if (Norm(sd.grad) != 0)

{

IntervalMinimumFunction(sd, 0, countf); //Находим интервал минимума функции

GoldenRatioMethod(sd, countf); //Находим значение лямбды

sd.point.x -= sd.lambda \* sd.grad.x / Norm(sd.grad); //Находим координаты новой точки (- минимум, + максимум)

sd.point.y -= sd.lambda \* sd.grad.y / Norm(sd.grad);

fnext = Funсtion(sd.point, countf); //Считаем значение функции в новой точке

count++;

fend = abs(fnext - fpred);

xend = abs(sd.point.x - pointpred.x);

yend = abs(sd.point.y - pointpred.y);

fout << count << "\t(" << sd.point.x << ", " << sd.point.y << ")\t" << fnext << "\t(" << sd.a << ", " << sd.b << ")\t" << sd.lambda;

fout << "\t" << xend << "\t" << yend << "\t" << fend << "\t(" << sd.grad.x << ", " << sd.grad.y << ") " << endl;

}

else

{

fnext = fpred;

fend = abs(fnext - fpred);

}

} while (fend > sd.ef && (xend > sd.epoint || yend > sd.epoint) && count < sd.maxiter);

fout << "\nStart point\t(" << \_x << ", " << \_y << ")\t" << "Eps f:\t" << sd.ef << "\tEps point:\t" << sd.epoint << "\tIter: " << count << "\tFunction iter: " << countf << endl;

fout << "Minimum of function in \t(" << sd.point.x << "," << sd.point.y << ")\tValue = " << fnext << endl;

}

//Обратная матрица Гессе

void InverseHesse(coords point, hesse &H)

{

Hessian(point, H); //Считаем вторые частные производные в точке

double det = H.dx2 \* H.dy2 - H.dxdy \* H.dydx; //Считаем определитель матрицы

if (det != 0)

{

double tmp = H.dx2; //Получаем обратную матрицу

H.dx2 = H.dy2 / det;

H.dy2 = tmp / det;

H.dxdy = -H.dxdy / det;

H.dydx = H.dxdy;

}

}

double H(method nw, int &countf)

{

double h = 1., delta = e / 2, f1, f2, f3, fres;

coords p1, p2, p3, res, mn;

mn.x = nw.H.dx2 \* nw.grad.x + nw.H.dxdy \* nw.grad.y; //Произведение матрицы Гессе на градиент

mn.y = nw.H.dxdy \* nw.grad.x + nw.H.dy2 \* nw.grad.y;

p1.x = nw.point.x - h \* mn.x;

p1.y = nw.point.y - h \* mn.y;

f1 = Funсtion(p1, countf);

p2.x = nw.point.x - (h + delta) \* mn.x;

p2.y = nw.point.y - (h + delta) \* mn.y;

f2 = Funсtion(p2, countf);

p3.x = nw.point.x - (h - delta) \* mn.x;

p3.y = nw.point.y - (h - delta) \* mn.y;

f3 = Funсtion(p3, countf);

if (f2 > f1 && f3 > f1)

return h;

else

{

if (f2 > f3)

delta \*= -1;

res = p1;

fres = Funсtion(res, countf);

}

h += delta;

p1.x = nw.point.x - h \* mn.x;

p1.y = nw.point.y - h \* mn.y;

f1 = Funсtion(p1, countf);

while (f1 < fres)

{

h += delta;

res = p1;

fres = f1;

p1.x = nw.point.x - h \* mn.x;

p1.y = nw.point.y - h \* mn.y;

f1 = Funсtion(p1, countf);

}

return h;

}

//Метод Ньютона

void Newton(double \_x, double \_y)

{

int count = 0, countf = 0; //Количество итераций

double fpred, fnext;

method nw;

nw.point.x = \_x, nw.point.y = \_y; //Начальное приближение (x, y)

coords pointpred = nw.point;

fout << scientific << setprecision(7) << "Iter\t(x, y)\tf(x, y)\t(s1, s2)\tlambda\t|x(i) - x(i-1)|\t|y(i) - y(i-1)|\t|f(i) - f(i-1)|\tgrad(x, y)\tHesse(dx2, dxdy, dydx, dy2)" << endl;

fnext = Funсtion(nw.point, countf); //Считаем значение функции

fpred = fnext;

double fend = abs(fnext - fpred), xend = abs(nw.point.x - pointpred.x), yend = abs(nw.point.y - pointpred.y);

do

{

fpred = fnext;

GradFunсtion(nw.point, nw.grad); //Считаем частные производные

if (Norm(nw.grad) != 0)

{

InverseHesse(nw.point, nw.H); //Получаем обратную матрицу Гессе

nw.lambda = H(nw, countf); //Находим шаг

nw.point.x -= nw.lambda \* (nw.H.dx2 \* nw.grad.x + nw.H.dxdy \* nw.grad.y); //Находим координаты новой точки (- минимум, + максимум)

nw.point.y -= nw.lambda \* (nw.H.dxdy \* nw.grad.x + nw.H.dy2 \* nw.grad.y);

fnext = Funсtion(nw.point, countf); //Считаем значение функции в новой точке

count++;

nw.a = nw.point.x - pointpred.x; //Направление

nw.b = nw.point.y - pointpred.y;

fend = abs(fnext - fpred);

xend = abs(nw.point.x - pointpred.x);

yend = abs(nw.point.y - pointpred.y);

fout << count << "\t(" << nw.point.x << ", " << nw.point.y << ")\t" << fnext << "\t(" << nw.a << ", " << nw.b << ")\t" << nw.lambda;

fout << "\t" << xend << "\t" << yend << "\t" << fend << "\t(" << nw.grad.x << ", " << nw.grad.y << ") " << "\t(" << nw.H.dx2 << ", " << nw.H.dxdy << ", " << nw.H.dydx << ", " << nw.H.dy2 << ")" << endl;

}

else

{

fnext = fpred;

fend = abs(fnext - fpred);

}

} while (fend > nw.ef && (xend > nw.epoint || yend > nw.epoint) && count < nw.maxiter);

fout << "\nStart point \t(" << \_x << ", " << \_y << ")\t" << "Eps f: \t" << nw.ef << "\tEps point:\t" << nw.epoint << "\tIter:\t" << count << "\tFunction iter: \t" << countf << endl;

fout << "Minimum of function in \t(" << nw.point.x << "," << nw.point.y << ")\tValue = \t" << fnext << endl;

}

int main()

{

fout.open("Out.txt");

SteepestDescent(1.26, 1.33); //Начальное приближение

//Newton(1.5, 0);

return 0;

}

*File - Function.h*

#pragma once

#include<stdio.h>

#include<math.h>

struct coords { double x, y; };

struct hesse { double dx2, dxdy, dydx, dy2; };

int num = 2;

double e = 1e-07; //Точность для поиска лямбды

struct method

{

coords point; //Точка

coords grad; //Градиент в точке

hesse H; //Матрица Гессе для метода Ньютона

double lambda;

double a, b; //Интервал для минимизации лямбды

double ef = 1e-07, epoint = 1e-07; //Точность по функции, точность по переменным

int maxiter = 1000;

};

double Funсtion(coords point, int &countf)

{

countf++;

if (num == 0) //Квадратичная функция

return 100 \* pow(point.y - point.x, 2) + pow(1 - point.x, 2);

if (num == 1) //Функция Розенброка

return 100 \* pow(point.y - point.x \* point.x, 2) + pow(1 - point.x, 2);

if (num == 2) //Функция по варианту

{

double a = -pow((point.x - 1) / 2, 2) - pow(point.y - 1, 2);

double b = -pow((point.x - 2) / 3, 2) - pow((point.y - 3) / 2, 2);

return -(2 \* exp(a) + 3 \* exp(b));

}

}

void GradFunсtion(coords point, coords &grad)

{

if (num == 0) //Квадратичная функция

{

grad.x = -200 \* (point.y - point.x) - 2 \* (1 - point.x);

grad.y = 200 \* (point.y - point.x);

}

if (num == 1) //Функция Розенброка

{

grad.x = -400 \* point.x \* (point.y - pow(point.x, 2)) - 2 \* (1 - point.x);

grad.y = 200 \* (point.y - pow(point.x, 2));

}

if (num == 2) //Функция по варианту

{

double a = -pow((point.x - 1) / 2, 2) - pow(point.y - 1, 2);

double b = -pow((point.x - 2) / 3, 2) - pow((point.y - 3) / 2, 2);

grad.x = -((point.x - 1) \* exp(a) + 2 / 3 \* (point.x - 2) \* exp(b));

grad.y = -(4 \* (point.y - 1) \* exp(a) + 3 / 2 \* (point.y - 3) \* exp(b));

}

}

//Норма функции

double Norm(coords p)

{

return sqrt(pow(p.x, 2) + pow(p.y, 2));

}

double LambdaFunсtion(coords point, coords grad, double lambda, int &countf)

{

coords l;

l.x = point.x - lambda \* grad.x / Norm(grad);

l.y = point.y - lambda \* grad.y / Norm(grad);

return Funсtion(l, countf);

}

void Hessian(coords point, hesse &matrix)

{

if (!num) //Квадратичная функция

{

matrix.dx2 = 202;

matrix.dxdy = -200;

matrix.dydx = matrix.dxdy;

matrix.dy2 = 200;

}

if (num) //Функция Розенброка

{

matrix.dx2 = -400 \* (point.y - 3 \* pow(point.x, 2)) + 2;

matrix.dxdy = -400 \* point.x;

matrix.dydx = matrix.dxdy;

matrix.dy2 = 200;

}

if (num == 2) //Функция по варианту

{

double a = -pow((point.x - 1) / 2, 2) - pow(point.y - 1, 2);

double b = -pow((point.x - 2) / 3, 2) - pow((point.y - 3) / 2, 2);

matrix.dx2 = (1 - pow(point.x - 1, 2) / 2) \* exp(a) + 2 / 3 \* (1 - 2 / 9 \* pow(point.x - 2, 2)) \* exp(b);

matrix.dxdy = -2 \* (point.x - 1) \* (point.y - 1) \* exp(a) - 1 / 3 \* (point.x - 2) \* (point.y - 3) \* exp(b);

matrix.dydx = matrix.dxdy;

matrix.dy2 = 4 \* (1 - 2 \* pow(point.y - 1, 2)) \* exp(a) + 3 / 2 \* (1 - pow(point.y - 3, 2) / 2) \* exp(b);

}

}