Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«Московский государственный технический университет
имени Н. Э. Баумана»

Кафедра «Прикладная математика»

В. В. Пузикова

ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ ЗАДАЧ МНОГОМЕРНОЙ ОПТИМИЗАЦИИ

Методические указания
к выполнению лабораторных работ
по курсу «Методы оптимизации»

Под редакцией *А. В. Аттеткова*

Москва
МГТУ им. Н. Э. Баумана
2016

УДК 519.6

Рецензент

Пузикова В. В.

М30 Численные методы решения задач многомерной оптимизации: метод. указания к выполнению лабораторных работ по курсу «Методы оптимизации» / Под ред. А. В. Аттеткова. — М.: МГТУ им. Н. Э. Баумана, 2016. — 80 с.

Разработан лабораторный практикум по курсу «Методы оптимизации», ориентированный на изучение численных методов решения задач многомерной безуслованой минимизации и задач нелинейного программирования.

Приведены примеры заданий, сформулированы контрольные вопросы и требования, предъявляемые к отчетам по лабораторным работам.

Для студентов 3-го курса, обучающихся по направлению подготовки бакалавров «Прикладная математика».

УДК 519.6

Пузикова Валерия Валентиновна

Численные методы решения задач многомерной оптимизации

Методические указания к выполнению лабораторных работ по курсу «Методы оптимизации»

Компьютерная верстка *И. К. Марчевского*, *В. В. Пузиковой*

Оглавление

ПРЕДИСЛОВИЕ

Курс «Методы оптимизации» входит в цикл дисциплин, которые изучают бакалавры, обучающиеся по направлению подготовки «Прикладная математика». Лабораторный практикум составлен в соответствии с учебной программой дисциплины, рассчитан на один семестр и предполагает выполнение восьми лабораторных работ. Задачами лабораторного практикума являются изучение численных методов оптимизации и алгоритмов, их использование для решения прикладных задач оптимизации и получение навыков вычислительного программирования.

В рамках данного курса лабораторных работ изучаются методы многомерной безусловной минимизации (лабораторные работы № 1–6) и численные методы нелинейного программирования (лабораторные работы № 7, 8). При этом на лабораторных работах № 1 и 2 рассматриваются методы первого порядка (применяются для дифференцируемых целевых функций), на лабораторных работах № 3 и 4 — методы второго порядка (применяются для дважды дифференцируемых целевых функций), на лабораторных работах № 5 и 6 — методы прямого поиска или нулевого порядка (применяются для целевых функций, для которых возможно вычислить значение в любой точке ее области определения).

Для успешного выполнения заданий необходимо знание следующих дисциплин, включенных в учебный план специальности «Прикладная математика»: математический анализ, аналитическая геометрия, линейная алгебра, дифференциальная геометрия, функциональный анализ, информатика, практикум по математическим пакетам.

Выполнение лабораторных работ включает в себя написание программ, реализующих соответствующие численные методы оптимизации, и подготовку отчета. Содержание отчетов приведено в заданиях к лабораторным работам.

На защите лабораторной работы студент должен продемонстрировать знание теории численных методов оптимизации в необходимом объеме и умение применять ее на практике. ***Способ проведения защиты определяется преподавателем.*** При подготовке к защите рекомендуется использовать контрольные вопросы, приведенные в конце каждого задания.

Объем лабораторных работ достаточно велик, поэтому рекомендуется выполнять их последовательно в течение всего семестра в соответствии с лекционным материалом и материалом, изучаемым на практических занятиях. Рекомендуемый график защиты лабораторных работ приведен в таблице.

№ п/п	**Тема лабораторной работы**	**Неделя защиты**
1	Методы градиентного спуска	2
2	Методы сопряженных градиентов	4
3	Метод Ньютона и его модификации	6
4	Квазиньютоновские методы	8
5	Методы спуска	10
6	Методы симплексного поиска	12
7	Прямые численные методы нелинейного программирования	14
8	Методы последовательной безусловной минимизации	16

Лабораторные работы выполняются в системе компьютерной алгебры Wolfram Mathematica и в среде разработки Microsoft Visual Studio на языке C++ в разработанной автором данного пособия программе-шаблоне `optimization_methods` (описание работы с этим программным комплексом будет представлено далее). При этом в Wolfram Mathematica реализуются алгоритмы, в которых на каждой итерации необходимо решать задачи

одномерной минимизации для того, чтобы вместо реализации методов дихотомии, золотого сечения и т.д., которые к тому же могут быть применены только для минимизации унимодальных целевых функций, использовать встроенную функцию `NArgMin`. Таким образом, численные методы решения задач одномерной минимизации остаются для самостоятельного изучения. В учебных целях в рамках лабораторных работ рассматриваются только задачи двумерной минимизации. При реализации всех алгоритмов необходимо учитывать наиболее общий случай неквадратичной целевой функции.

Автор выражает искреннюю благодарность за полезное обсуждение текста пособия и помощь в подготовке заданий для лабораторных работ своим коллегам, а также студентам кафедры «Прикладная математика» МГТУ им. Н. Э. Баумана.

Лабораторная работа № 1

Методы градиентного спуска

Цель работы

Изучение методов градиентного спуска:

- метода наискорейшего спуска;
- метода градиентного спуска с дроблением шага.

Содержание работы

Рассмотрим *задачу безусловной минимизации*

$$f:\ \mathbb{R}^n \to \mathbb{R},\quad f(X) \to \min,\quad X \in \Omega = D(f) \subset \mathbb{R}^n, \tag{1.1}$$

где $f(X)$ — целевая функция, Ω — допустимое множество, совпадающее с областью определения $D(f)$ функции $f(X)$.

Предполагаем, что целевая функция дифференцируема.

Требуется выполнить следующие действия.

1. Для решения задачи (1.1) реализовать алгоритм метода наискорейшего спуска в системе Wolfram Mathematica.

2. Для решения задачи (1.1) реализовать алгоритм метода градиентного спуска с дроблением шага на языке C++ в программе-шаблоне `optimization_methods`, предусмотреть выгрузку результатов решения в файл и считывание результатов из этого файла в системе Wolfram Mathematica.

3. На тестовых примерах продемонстрировать работоспособность программ.

4. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, траекторию поиска точки минимума, график убывания нормы невязки.

5. Экспериментально исследовать свойства методов. Как влияет заданная точность ϵ на скорость сходимости методов? Каким свойством обладают антиградиенты w^{i-1} и w^i? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета?

Методические указания

Общие сведения

Задачи оптимизации возникают в самых различных ситуациях: в процедуре обучения глубинных нейронных сетей (многослойных персептронов), при распознавании образов, в задачах проектировании технических устройств и технологических процессов, в которых какие-либо параметры устройства или процесса могут варьироваться, причем за счет изменения этих параметров можно повысить эксплуатационные характеристики устройства или экономичность проведения технологического процесса и т. д. При этом оптимизация может проводиться не только по одному параметру (*одномерная оптимизация*), но и по нескольким (*многомерная оптимизация*). В данном пособии сосредоточимся на численном решении задач многомерной оптимизации. Будем рассматривать самый простой случай — минимизацию функций двух переменных.

В зависимости от требований, накладываемых на целевую функцию, численные методы оптимизации могут быть разделены на три класса:

1) методы первого порядка (предполагают дифференцируемость целевой функции);
2) методы второго порядка (предполагают, что целевая функция является дважды дифференцируемой);

3) прямые методы (предполагают, что на всем допустимом множестве значение целевой функции может быть вычислено).

Также приведем некоторые базовые сведения из теории функций нескольких переменных.

- Необходимое условие экстремума: если $X^* \in \Omega$ — точка экстремума функции $f(X)$ и $f(X)$ дифференцируема в точке X^*, то $\text{grad} f(X^*) = \vec{0}$.
- Градиент указывает направление наиболее быстрого возрастания функции, а, значит, перпендикулярен линиям уровня. Соответственно, антиградиент $w^k = -\text{grad} f(X^{k-1})$ указывает направление наиболее быстрого уменьшения функции и, как и градиент, перпендикулярен линиям уровня, а в точке минимума равен нулевому вектору.

Простейшие методы численного решения задач многомерной безусловной минимизации

*Как из произвольной точки $X^0 \in \Omega$ (**начального приближения**) дойти до точки минимума X^* и желательно за наименьшее число шагов (**итераций**)?* Опираясь на вышеизложенные сведения, логично предложить следующую траекторию поиска точки минимума (***метод градиентного спуска***):

$$X^k = X^{k-1} + \varkappa_k w^k, \quad k = 1, 2, \ldots, \tag{1.2}$$

где $\varkappa_k = \beta_k / ||w^k||$, β_k — шаг спуска, w^k — направление спуска, совпадающее с направлением антиградиента в точке X^{k-1}. При этом $\varkappa_k > 0$, иначе будем «идти» в противоположную сторону.

До каких пор «шагать», т.е. как понять, что $X^k \approx X^*$ с приемлемой точностью и дальше приближения строить не надо? Другими словами, нужно задать приемлемую ***точность*** ϵ и ***критерий останова***. Поскольку $w^* = \vec{0} \Leftrightarrow ||w^*|| = 0$, логично остановиться, когда $||w^k|| < \epsilon$. Также можно использовать условия $||X^k - X^{k-1}|| < \epsilon$, $|f(X^k) - f(X^{k-1})| < \epsilon$.

Можно ли выбирать $\varkappa_k$ так, чтобы сократить число итераций? Самое логичное — выбирать $\varkappa_k$ так, чтобы

$$\varphi_k(\varkappa) \to \min, \quad \varphi_k(\varkappa) = f(X^{k-1} + \varkappa w^k), \quad \varkappa \in [0, \hat{\varkappa}],$$

т.е. решать на каждой итерации метода градиентного спуска задачу одномерной минимизации. Полученный метод называется ***методом наискорейшего спуска***. Однако, такой выбор шага спуска увеличивает вычислительные затраты на одну итерацию и, в конечном счете, несмотря на уменьшение числа итераций время счета может увеличиться. Для решения задач одномерной минимизации необходимо также самостоятельно реализовать численный метод (метод дихотомии, метод золотого сечения и т.д.) или использовать пакеты, в которых эти методы уже реализованы (например, в системе Wolfram Mathematica есть функция `NArgMin`).

Вместо решения задачи одномерной минимизации можно использовать процедуру ***дробления шага***: уменьшать на k-й итерации $\varkappa_k$ в $1/\gamma$ раз ($\gamma \in (0,1)$), пока убывание значения функции не достигнет приемлемого уровня, например, пока не выполнится условие

$$f(X^{k-1}) - f(X^k) > \omega \varkappa_k ||w^k||^2, \quad \omega \in (0,1).$$

В этом случае скорость сходимости метода зависит от того, насколько удачно были выбраны значения параметров γ и ω.

Способы анализа результатов численного решения задач оптимизации

Для контроля правильности решения задачи оптимизации необходимо построить траекторию поиска точки минимума, совмещенную с топографией целевой функции (рис. 1.1). Для построения подобного графика в Wolfram Mathematica требуется выполнить следующую команду:

```
ContourPlot[f[{x, y}], {x, -3, 3}, {y, -3, 3},
 Contours ->fValues, ColorFunction ->(If[# < 0,
 Darker[Blue,Abs[0.5#]], Lighter[Orange,0.5#]]&),
 ColorFunctionScaling->True, ContourStyle->Black,
```

```
FrameLabel -> Automatic, PlotLabel ->
 Style[Row[{Style["f(x,y)", Italic], " = ",
                              f[{x, y}]}], Large],
LabelStyle -> Large, FrameStyle -> Thick,
ImageSize -> 350, PlotPoints -> 100,
(*траектория поиска точки минимума*)
Epilog -> {Purple, Thickness[0.01],
 Arrowheads[.08], Table[Arrow[points[[i-1;;i]]],
           {i, 2, Length[points]}], Darker[Red],
     PointSize[0.035], Point[points[[1 ;; -1]]],
                  Black, Point[points[[-1]]]}]
```

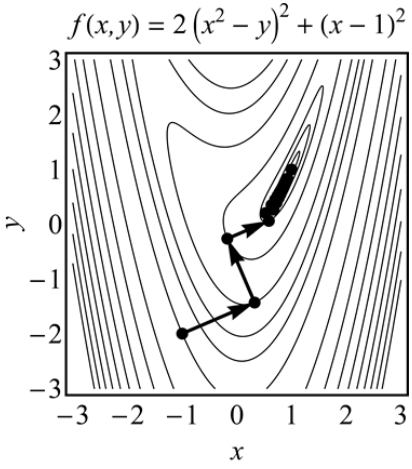
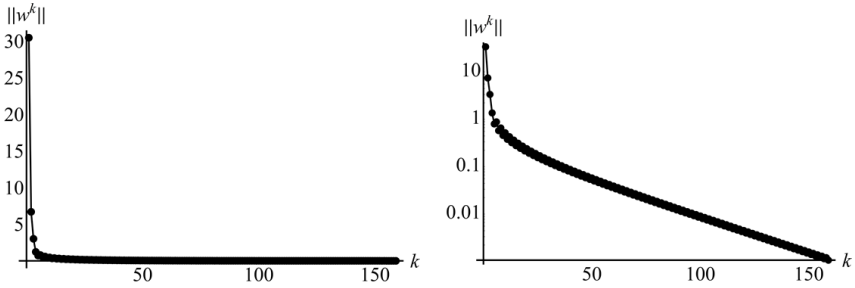


Рис. 1.1. Траектория поиска точки минимума методом наискорейшего спуска и топография целевой функции

Для анализа скорости сходимости метода необходимо построить график убывания нормы невязки. Отметим, что для этой цели удобно строить данный график в логарифмическом масштабе(рис. 1.2 *(а)*), а не в линейном(рис. 1.2 *(б)*):

```
ListLogPlot[norm, PlotRange -> All,
 PlotStyle -> {Blue, Thick, PointSize[Large]},
 AxesStyle -> Thick, LabelStyle -> Large,
 AxesLabel -> {"k", "||w^k||"},
 ImageSize -> 700, Joined -> True, Mesh -> Full,
```

```
   MeshStyle -> Directive[PointSize[0.02], Red]]
```



(а) *(б)*

Рис. 1.2. Убывание нормы невязки: *(а)* в линейном масштабе; *(б)* в логарифмическом масштабе

Также в Wolfram Mathematica можно представить результаты вычислительного эксперимента в табличном виде. Для этого нужно выполнить следующие команды.

```
(*"шапка" таблицы*)
gridLabel = {"k", "(X^k)", "f(X^k)",
                  "\[Kappa]_k", "||w^k||"};
gridData = Table[Table[{}, {j, 1,
    Length[gridLabel]}], {i, 1, Length[points]}];
(*в 1-м столбце - номер итерации*)
gridData[[All, 1]] = Table[i,
                     {i, 0, Length[points] - 1}];
(*во 2-м столбце - точки траектории*)
For[i = 1, i <= Length[points], i++,
 gridData[[i, 2]] = "( " <>
 ToString[SetAccuracy[points[[i, 1]], 4]] <> ", "
 <> ToString[SetAccuracy[points[[i, 2]], 4]] <>
                                          " )"]
(*в 3-м столбце - значение функции*)
gridData[[All, 3]] = SetAccuracy[fValues, 4];
(*в 4-м столбце - значение шага спуска*)
gridData[[All, 4]] = SetAccuracy[\[Kappa]Array,4];
(*в 5-м столбце - значение нормы невязки*)
```

```
gridData[[All, 5]] = SetAccuracy[norm, 4];
gridData = {gridLabel}~Join~gridData;
(*выводим всю таблицу*)
table=Grid[gridData, Frame -> All,
         FrameStyle -> Thick, Spacings -> {2, 2},
          ItemStyle -> Directive[FontSize -> 15]]
```

Полученную таблицу можно легко перенести в отчет, набираемый в системе LaTex. Для этого нужно скопировать в отчет результат выполнения команды `TeXForm[table]`. В результате получается таблица, подобная табл. 1.

Содержание отчета

1. Постановка задачи.
2. Номер варианта и исходные данные.
3. Краткое описание методов решения.
4. Ответы на контрольные вопросы.
5. Тестовые примеры, демонстрирующие работоспособность программы.
6. Результаты расчетов (топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки) и их анализ. Сравнение решений, полученных различными методами с разной точностью.
7. Сравнение скорости сходимости рассмотренных методов при поиске точки минимума функции Розенброка[1]

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

   при различных значениях α.
8. Листинг кода программы.

[1] Хо́вард Га́рри Розенбро́к (1920–2010) — английский математик, специалист по теории управления и методам оптимизации, разработал методы численного решения задач многомерной оптимизации, предложил функцию для оценки производительности алгоритмов оптимизации.

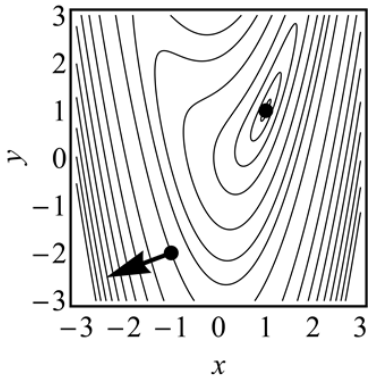
Таблица 1.1 . Минимизация функции $f(x,y) = (x^2 - y)^2 + (x-1)^2$ методом наискорейшего спуска

k	X^k	$f(X^k)$	$\varkappa_k$	$\|\|w^k\|\|$
0	(-1.0000, -2.0000)	13.0000	1.0000	17.0880
1	(0.3786, -1.4830)	3.0312	0.0862	3.4739
2	(-0.0298, -0.3941)	1.2165	0.3348	2.2499
3	(0.5899, -0.1617)	0.4280	0.2941	1.0887
4	(0.4911, 0.1017)	0.2785	0.2584	0.7945
5	(0.6933, 0.1775)	0.1860	0.2719	0.6476
6	(0.6403, 0.3189)	0.1377	0.2331	0.5190
7	(0.7582, 0.3631)	0.1033	0.2426	0.4524
8	(0.7234, 0.4560)	0.0810	0.2194	0.3831
...	...	...	...	...
92	(0.9989312, 0.9976)	$1.2 \cdot 10^{-6}$	0.1782	0.0013
93	(0.9991230, 0.9977)	$1.1 \cdot 10^{-6}$	0.1569	0.0012
94	(0.9990508, 0.9979)	$9.0 \cdot 10^{-7}$	0.1782	0.0012
95	(0.9992210, 0.9980)	$8.0 \cdot 10^{-7}$	0.1569	0.0010
96	(0.9991569, 0.9981)	$7.0 \cdot 10^{-7}$	0.1782	0.0010
97	(0.9993081, 0.9982)	$7.0 \cdot 10^{-7}$	0.1568	0.0009

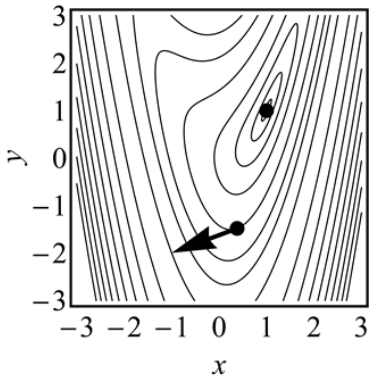
Контрольные вопросы

1. Что такое линия уровня?
2. Что называют градиентом функции нескольких переменных?

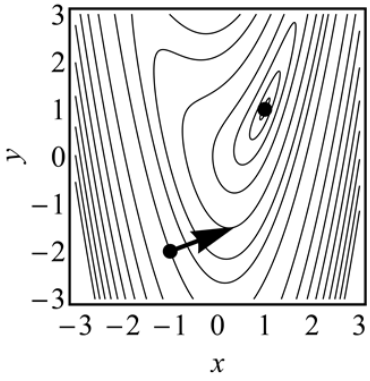
3. На каком из рисунков изображен антиградиент функции Розенброка? Градиент?



(а)



(б)



(в)

4. Сформулируйте необходимое условие экстремума функции нескольких переменных.

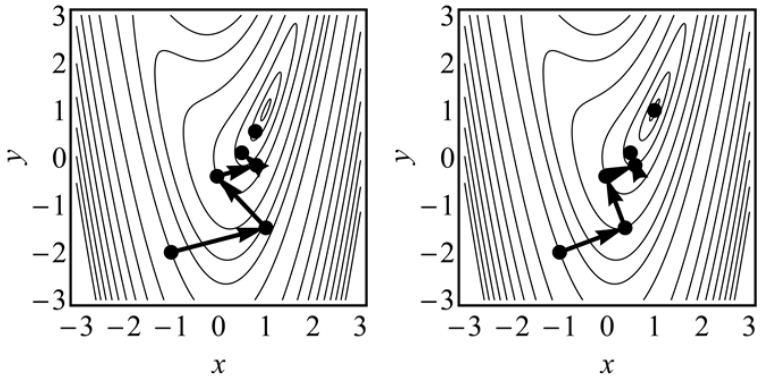
5. Сформулируйте достаточные условия экстремума функции нескольких переменных.

6. При каком условии у квадратичной функции

$$f(x,y) = ax^2 + 2bxy + cy^2$$

существует точка глобального минимума? Точка глобального максимума? Не существует точки глобального экстремума?

7. Что нужно изменить в формуле (1.2), чтобы метод градиентного спуска позволял искать не точку минимума, а точку максимума?
8. На каком из рисунков изображена траектория метода наискорейшего спуска?



(а) *(б)*

9. Привести пример графика функции, для которой метод наискорейшего спуска позволяет получить решение за 1 итерацию.

Примеры заданий

1. При помощи методов градиентного спуска с точностью $\epsilon = 10^{-3}$ найти точку минимума функции Розенброка

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

при $\alpha = 1$ и $\alpha = 50$. Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи метода наискорейшего спуска с точностью $\epsilon = 10^{-8}$ найти точку минимума функции Химмельблау[2]

$$f(x,y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2.$$

Начальное приближение: а) $(0, 0)^T$; б) $(5, -5)^T$; в) $(0, -4)^T$; г) $(-5, 5)^T$; д) $(-1, -2)^T$; е) $(-3, -2)^T$.

Ответ: а) $(3.00, 2.00)^T$; б) $(3.58, -1.85)^T$; в) $(-2.81, 3.13)^T$; г) $(-2.81, 3.13)^T$; д) $(3.58, -1.85)^T$; е) $(-3.78, -3.28)^T$.

[2] Дэ́вид Мо́нтер Химмельбла́у (1924–2011) — американский специалист по химии, нелинейному программированию, статистическим методам, автор ряда книг.

Лабораторная работа № 2

Методы сопряженных градиентов

Цель работы

Изучение методов сопряженных градиентов:

- метода Флетчера[1] — Ривса;
- метода Полака[2] — Рибьера;
- варианта метода сопряженных градиентов при $Q = H(X^k)$.

Содержание работы

Рассмотрим *задачу безусловной минимизации* (1.1). Предполагаем, что целевая функция дифференцируема.

Требуется выполнить следующие действия.

1. Для решения задачи (1.1) реализовать в системе Wolfram Mathematica алгоритмы следующих методов сопряженных градиентов:
   - метода Флетчера — Ривса;
   - метода Полака — Рибьера;
   - метода сопряженных градиентов при $Q = H(X^k)$.

2. На тестовых примерах продемонстрировать работоспособность программ.

---

[1] Ро́джер Фле́тчер — британский математик, профессор Университета Данди, выиграл премию Лагранжа, награжден медалью Королевского общества Эдинбурга.

[2] Илиа́ По́лак (родился в 1931 г.) — польский специалист по теории управления, интеллектуальным системам, робототехнике, численным методам оптимизации, автор более чем 200 статей и четырех книг.

3. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, траекторию поиска точки минимума, график убывания нормы невязки.

4. Экспериментально исследовать свойства методов. Как влияет заданная точность ϵ на скорость сходимости методов? Как отличается скорость сходимости методов при минимизации квадратичной функции? Как влияет «овражность» целевой функции на скорость сходимости методов?

5. Провести сравнение методов сопряженных градиентов с методами градиентного спуска. Чем отличаются траектории поиска точки минимума? Как отличается скорость сходимости методов при минимизации овражной функции? При минимизации квадратичной функции?

Методические указания

Метод сопряженных направлений

Попробуем в формуле (1.2) заменить направление спуска с антиградиента w^k на вектор p^k с целью ускорения сходимости метода:

$$X^k = X^{k-1} + \varkappa_k p^k, \quad k = 1, 2, \ldots. \tag{2.1}$$

Значение $\varkappa_k$ будем выбирать как в методе наискорейшего спуска:

$$\varphi_k(\varkappa) \to \min, \quad \varphi_k(\varkappa) = f(X^{k-1} + \varkappa p^k), \quad \varkappa \in [0, \hat{\varkappa}].$$

Пусть целевая функция является квадратичной:

$$f(X) = \frac{1}{2}(QX, X) + (b, X). \tag{2.2}$$

Здесь Q — положительно определенная симметрическая матрица, (u, v) — скалярное произведение векторов u и v. Когда направлением спуска служили антиградиенты, направления с соседних

итераций были ортогональны друг другу:

$$(w^{i-1}, w^i) = 0.$$

Теперь потребуем, чтобы направления с соседних итераций были ***сопряженными относительно матрицы*** Q, т.е. ***Q-ортогональными***:

$$(Qp^{i-1}, p^i) = 0.$$

Потребуем также, чтобы для системы векторов, составленной из направлений спуска, выполнялось также следующее соотношение:

$$p^{k+1} = \gamma_k p^k + w^{k+1}. \tag{2.3}$$

Здесь $\gamma_k \in \mathbb{R}$ — параметр, значение которого нам предстоит вычислить. На первой итерации p^1 полагаем равным w^1. Поскольку при построении системы векторов p^i, $i = 1, 2, \ldots$, используются антиградиенты, построенный метод, как и методы градиентного спуска, относится к методам первого порядка.

Умножая (2.3) на Qp^k и учитывая Q-ортогональность векторов p^k и p^{k+1}, получаем следующую формулу для вычисления значения γ_k:

$$\gamma_k = -\frac{(Qp^k, w^{k+1})}{(Qp^k, p^k)}. \tag{2.4}$$

Полученный метод (2.1), (2.3), (2.4) называют ***методом сопряженных направлений*** [1]. Поскольку при вычислении значения γ_k используется матрица Q, метод подходит только для минимизации квадратичной функции вида (2.2). Пример траектории поиска точки минимума квадратичной функции методом сопряженных направлений представлен на рис. 2.1. Первый из методов сопряженных направлений был предложен в 1952 г. М.Р. Хестенсом и Е. Штифелем[3].

[3] Hestenes M.R., Stefel E. Methods of conjugate gradients for solving linear systems // J. Res. Nation: Bureau Stand. 1952. V. 49, № 6. P. 409–436.

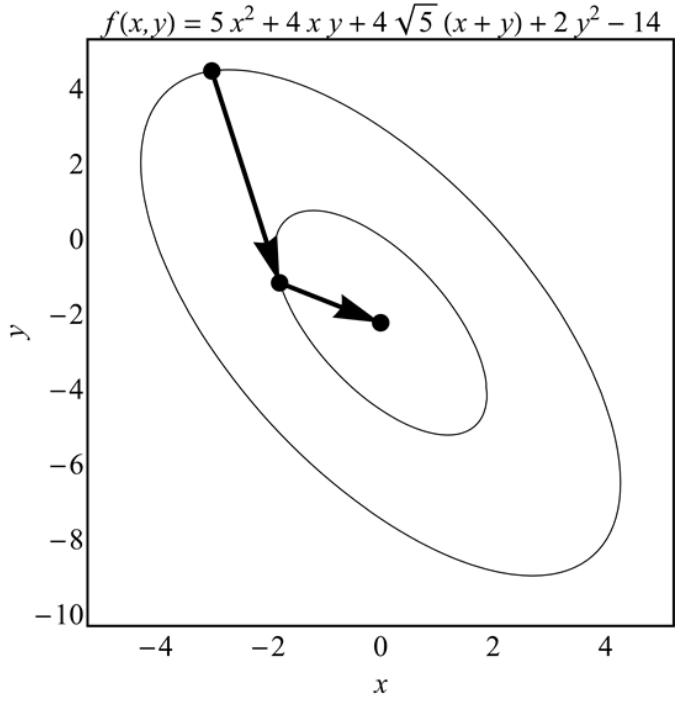


Рис. 2.1. Траектория поиска точки минимума квадратичной функции методом сопряженных направлений

Методы сопряженных градиентов

Построим аналог метода сопряженных направлений, подходящий для минимизации неквадратичных функций. Для этого нужно вывести формулу для вычисления знаения γ_k, не содержащую матрицу Q. Учтем, что для квадратичной функции (2.2) антиградиент равен

$$w^j = -Qx^{j-1} - b, \quad j = 1,2,\ldots.$$

Тогда

$$w^k - w^{k-1} = -Q(x^{k-1} - x^{k-2})$$

и, с учетом (2.1), получаем

$$w^{k-1} - w^k = \varkappa_{k-1} Q p^{k-1}, \quad k > 1. \tag{2.5}$$

С учетом (2.5) и свойства ортогональности антиградиентов получаем, что

$$(Qp^{k-1}, w^k) = \frac{(w^{k-1} - w^k, w^k)}{\varkappa_{k-1}} = -\frac{||w^k||^2}{\varkappa_{k-1}}. \tag{2.6}$$

Кроме того, с учетом (2.5) и ортогональности векторов w^k и p^{k-1}, имеем

$$(Qp^{k-1}, p^{k-1}) = \frac{(w^{k-1} - w^k, p^{k-1})}{\varkappa_{k-1}} = \frac{(w^{k-1}, p^{k-1})}{\varkappa_{k-1}}. \tag{2.7}$$

Перепишем формулу (2.4) с учетом (2.6) и (2.7):

$$\gamma_k = \frac{||w^{k+1}||^2}{(w^k, p^k)}. \tag{2.8}$$

Полученный метод (2.1), (2.3), (2.8) называют ***методом сопряженных градиентов***. Поскольку в формулу (2.8) не входит матрица Q, *метод сопряженных градиентов в отличие от метода сопряженных направлений можно использовать для минимизации неквадратичных функций.*

Умножим (2.3) на w^{k+1}. Тогда с учетом ортогональности векторов w^{k+1} и p^k получаем

$$(w^{k+1}, p^{k+1}) = \gamma_k (w^{k+1}, p^k) + (w^{k+1}, w^{k+1}) = ||w^{k+1}||^2$$

и (2.8) принимает вид

$$\gamma_k = \frac{||w^{k+1}||^2}{||w^k||^2}. \tag{2.9}$$

Метод (2.1), (2.3), (2.9) называют ***методом Флетчера — Ривса***.

В силу свойства ортогональности антиградиентов выполняется равенство

$$||w^{k+1}||^2 = (w^{k+1} - w^k, w^{k+1})$$

и (2.9) принимает вид

$$\gamma_k = \frac{(w^{k+1} - w^k, w^{k+1})}{||w^k||^2}. \tag{2.10}$$

Метод (2.1), (2.3), (2.10) называют ***методом Полака — Рибьера***.

Кроме того, можно заметить, что для квадратичной функции (2.2) матрица Q совпадает с матрицей Гессе. Тогда, если целевая функция (возможно неквадратичная) является дважды дифференцируемой, формулу (2.4) можно переписать в виде

$$\gamma_k = -\frac{(H(X^k)p^k, w^{k+1})}{(H(X^k)p^k, p^k)}, \tag{2.11}$$

где $H(X^k)$ — матрица Гессе целевой функции $f(X)$ в точке X^k. Метод (2.1), (2.3), (2.11) является еще ***одним вариантом метода сопряженных градиентов***.

Отметим, что формулы (2.9), (2.10) и (2.11) для квадратичной функции эквиваленты формуле (2.4), в то время как для неквадратичных функций перечисленные формулы дают различные значения γ_k, что приводит к разным итерационным процессам (рис. 2.2 и 2.3).

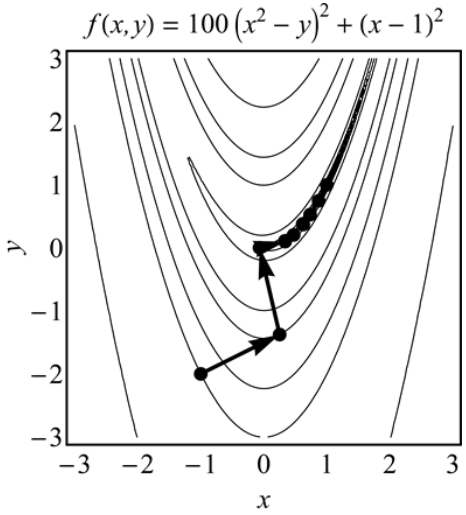


Рис. 2.2. Траектория поиска точки минимума функции Розенброка при $\alpha = 100$ методом Флетчера — Ривса

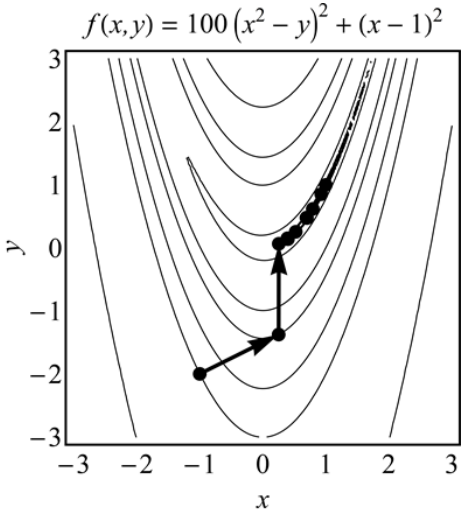


Рис. 2.3. Траектория поиска точки минимума функции Розенброка при $\alpha = 100$ вариантом метода сопряженных градиентов при $Q = H(X^k)$

Содержание отчета

1. Постановка задачи.
2. Краткое описание методов решения.
3. Ответы на контрольные вопросы.
4. Тестовые примеры, демонстрирующие работоспособность программы.
5. Результаты вычислительных экспериментов: топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки.
6. Анализ результатов. Сравнение эффективности рассмотренных методов сопряженных градиентов и методов градиентного спуска.
7. Листинг кода программы.

Контрольные вопросы

1. В чем состоит отличие методов сопряженных градиентов от метода наискорейшего спуска?

2. Дать определение сопряженных относительно матрицы Q (Q-ортогональных) векторов.
3. Как модифицируется формула для расчета γ_k для случая минимизации неквадратичной функции в вышеперечисленных методах сопряженных градиентов? Какое свойство при этом используется?
4. Какой из вышеперечисленных методов сопряженных градиентов позволяет найти точку минимума квадратичной функции за меньшее по сравнению с двумя другими методами число итераций?
5. За какое число итераций можно найти точку минимума квадратичной функции любым из вышеперечисленных методов?
6. Для минимизации каких целевых функций и зачем в алгоритмы вышеперечисленных методов необходимо добавить рестарты?
7. График какой функции имеет овражную структуру?

Примеры заданий

1. При помощи методов сопряженных градиентов с точностью $\epsilon = 10^{-3}$ найти точку минимума функции Розенброка

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

при $\alpha = 1$, $\alpha = 50$ и $\alpha = 1000$. Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи методов сопряженных градиентов с точностью $\epsilon = 10^{-3}$ найти точку минимума квадратичной функции

$$f(x,y) = 6x^2 + 3y^2 - 4xy + 4\sqrt{5}(x+2y) + 22.$$

Начальное приближение $X^0 = (-2\sqrt{5}, 1)^T$.

Ответ: $(-2.24, -4.47)^T$.

3. Попробуйте при помощи методов сопряженных градиентов с точностью $\epsilon = 10^{-3}$ найти точку минимума квадратичной функции

$$f(x, y) = 6x^2 - 3y^2 - 4xy + 4\sqrt{5}(x + 2y) + 22.$$

Начальное приближение $X^0 = (-2\sqrt{5}, 1)^T$.

Лабораторная работа № 3

Метод Ньютона и его модификации

Цель работы

Изучение методов второго порядка:

- метода Ньютона;
- модификаций метода Ньютона (на примере спуска с дроблением шага).

Содержание работы

Рассмотрим *задачу безусловной минимизации* (1.1). Предполагаем, что целевая функция дважды дифференцируема.

Требуется выполнить следующие действия.

1. Для решения задачи (1.1) реализовать на языке C++ в программе-шаблоне `optimization_methods` алгоритмы следующих методов:
   - метода Ньютона;
   - модификаций метода Ньютона (на примере спуска с дроблением шага).

2. На тестовых примерах продемонстрировать работоспособность программ.

3. Результаты решения задач выгружать в файл. Данные из этих файлов загружать в систему Wolfram Mathematica.

4. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию

целевой функции, траекторию поиска точки минимума, график убывания нормы невязки.

5. Экспериментально исследовать свойства методов. Как влияет выбор начального приближения на сходимость методов? Как влияет заданная точность ϵ на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета?

6. Провести сравнение метода Ньютона с методами первого порядка. Чем отличаются траектории поиска точки минимума? Как отличается скорость сходимости методов при минимизации овражных функций? При минимизации квадратичной функции?

Методические указания

Метод Ньютона

Если известно, что целевая функция является дважды дифференцируемой, то для решения задачи минимизации (1.1) можно использовать методы, относящиеся к методам второго порядка. При этом помимо информации об антиградиенте в таких методах используется информация о матрице Гессе.

Аппроксимируем целевую функцию $f(X)$ в окрестности текущей точки траектории поиска точки минимума X^{k-1} квадратичной функцией. Для этого запишем разложение целевой функции в окрестности точки X^{k-1} в ряд Тейлора с остаточным членом в форме Пеано:

$$f(X) = f(X^{k-1}) - w^k(X - X^{k-1}) + \\ + \frac{1}{2}(H_k(X - X^{k-1}), X - X^{k-1}) + o(||X - X^{k-1}||^2).$$

Здесь H_k — матрица Гессе функции $f(X)$ в точке X^{k-1}; w^k — антиградиент функции $f(X)$ в точке X^{k-1}. Пренебрегая остаточным

членом, получаем искомую квадратичную функцию

$$\varphi_k(X) = f(X^{k-1}) - w^k(X - X^{k-1}) + \frac{1}{2}(H_k(X - X^{k-1}), X - X^{k-1}),$$

аппроксимирующую целевую функцию $f(X)$ в окрестности текущей точки траектории поиска точки минимума X^{k-1}.

В качестве следующей точки траектории поиска точки минимума целевой функцию $f(X)$ возьмем точку минимума построенной квадратичной функции $\varphi_k(X)$, которую найдем из необходимого условия экстремума

$$\operatorname{grad}(\varphi_k(X)) = -w^k + H_k(X - X^{k-1}) = \vec{0}.$$

Таким образом, при $X = X^k$ получаем следующее рекуррентное соотношения для определения точки X^k ***методом Ньютона***:

$$X^k = X^{k-1} + H_k^{-1} w^k = X^{k-1} + p^k, \quad k = 1, 2, \ldots, \tag{3.1}$$

где $p^k = H_k^{-1} w^k$ — ***ньютоновское направление спуска***.

Отметим, что в случае, если H_k не является положительно определенной матрицей, *требуется дополнить ее до положительно определенной* при помощи положительно определенной диагональной матрицы. При этом направление спуска будет задаваться вектором

$$\widetilde{p}^k = (H_k + \eta_k \cdot I)^{-1} w^k,$$

где η_k — положительное число, выбираемое так, чтобы матрица $\widetilde{H_k} = H_k + \eta_k \cdot I$ была положительно определенной.

При большом количестве параметров, по которым идет минимизация (т.е. когда n велико), определение ньютоновского направления спуска представляет собой трудоемкую вычислительную задачу. Вместо обращения матрицы H_k направление спуска для (3.1) можно определять как решение системы линейных алгебраических уравнений

$$H_k p^k = w^k. \tag{3.2}$$

В случае заполненной[1] матрицы H_k систему (3.2) решают прямыми методами. В случае разреженной[2] матрицы H_k для решения системы (3.2) логично использовать итерационные методы. Однако, если целевая функция $f(X)$ имеет овражную структуру, матрица H_k является плохо обусловленной. Это означает, что наиболее простые итерационные методы (метод простой итерации, метод Якоби, метод Гаусса — Зейделя и т.д.) будут демонстрировать весьма медленную сходимость при решении системы (3.2). *Таким образом, для решения системы (3.2) с разреженной плохо обусловленной положительно определенной матрицей H_k целесообразным представляется использовать методы подпространств Крылова (например, метод сопряженных градиентов CG или метод обобщенных минимальных невязок GMRES) с предобуславливанием.*

Пример траектории поиска точки минимума методом Ньютона представлен на рис. 3.1.

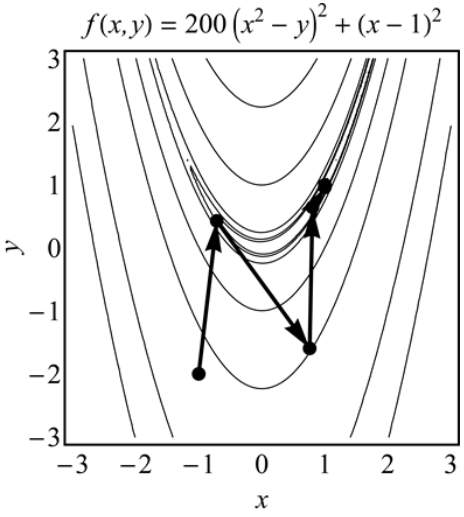


Рис. 3.1. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Ньютона

[1] Заполненной называется матрица, в которой мало элементов, равных нулю.

[2] Разреженной называется матрица, большинство элементов которой равно нулю.

Модификации метода Ньютона

Поскольку ньютоновское направление является направлением спуска, метод Ньютона можно трактовать как метод спуска с шагом спуска $\varkappa_k \equiv 1$:

$$X^k = X^{k-1} + \varkappa_k p^k, \quad k = 1, 2, \ldots. \tag{3.3}$$

Возникает идея использования других значений шага спуска, чтобы ускорить сходимость метода и уменьшить чувствительность к начальному приближения, характерную для метода Ньютона.

Во-первых, можно находить шаг спуска из решения задачи одномерной минимизации:

$$\varphi_k(\varkappa) \to \min, \quad \varphi_k(\varkappa) = f(X^{k-1} + \varkappa p^k), \quad \varkappa \in [0, \hat{\varkappa}]. \tag{3.4}$$

Метод (3.3), (3.4) называют ***методом исчерпывающего спуска***. Однако при этом следует учитывать, что решение задачи (3.4) само по себе является трудоемким.

Во-вторых, для определения значения $\varkappa_k$ можно использовать процедуру ***дробления шага***: необходимо уменьшать на k-й итерации значение $\varkappa_k$, изначально равное единице, в $1/\nu$ раз ($\nu \in (0,1)$), пока убывание значения функции не достигнет приемлемого уровня, например, пока не выполнится условие

$$f(X^{k-1}) - f(X^k) \geq \omega \varkappa_k (w^k, p^k), \quad \omega \in (0, 1/2).$$

В этом случае скорость сходимости метода зависит от того, насколько удачно были выбраны значения параметров ν и ω.

Пример траектории поиска точки минимума методом Ньютона с дроблением шага представлен на рис. 3.2.

Из результатов расчетов видно, что при минимизации функции Розенброка при $\alpha = 200$ метод Ньютона с дроблением шага сходится медленнее метода Ньютона (рис. 3.3). Однако, модификации метода Ньютона менее чувствительны к выбору начального приближения по сравнению с методом Ньютона. При этом в рассмотренных модификациях, как и в исходном методе Ньютона, существует необходимость решения системы линейных алгебраических уравнений (3.2) для определения ньютоновского направления спуска p^k.

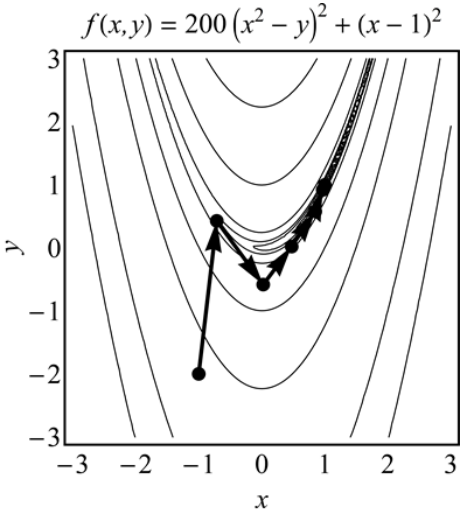


Рис. 3.2. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Ньютона с дроблением шага

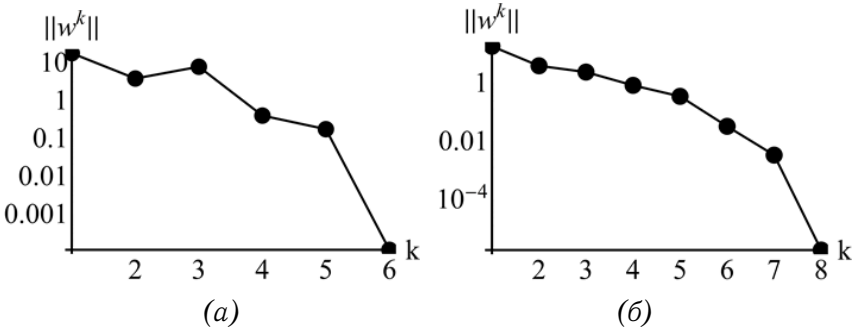


Рис. 3.3. Убывание нормы невязки при поиске точки минимума функции Розенброка при $\alpha = 200$: *(а)* методом Ньютона; *(б)* методом Ньютона с дроблением шага

Содержание отчета

1. Постановка задачи.

2. Ответы на контрольные вопросы.

3. Тестовые примеры, демонстрирующие работоспособность программы.
4. Результаты расчетов (топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки) и их анализ. Исследование зависимости сходимости методов от начального приближения. Сравнение эффективности методов оптимизации второго порядка с методами оптимизации первого порядка.
5. Листинг кода программы.

Контрольные вопросы

1. Для каких функций можно использовать методы второго порядка?
2. Какая информация о целевой функции используется в методах второго порядка?
3. Вывести формулу расчета x^k для метода Ньютона. Какие требования при этом накладываются на матрицу Гессе?
4. В чем состоит основной недостаток метода Ньютона?
5. Какими численными методами можно воспользоваться для вычисления ньютоновского направления спуска? Какие трудности возникают при использовании этих методов?
6. Какая матрица называется положительно определенной?
7. Как проверить положительную определенность симметричной матрицы?
8. Что делать, если матрица Гессе на очередной итерации не будет положительно определенной?
9. В чем смысл модификации метода Ньютона (спуска с дроблением шага)?

Примеры заданий

1. При помощи метода Ньютона и его модификаций с заданной точностью $\epsilon = 10^{-3}$ найти точку минимума функции

Розенброка

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

при $\alpha = 1$, $\alpha = 50$, $\alpha = 1000$. Начальное приближение полагать равным $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи метода Ньютона и его модификаций с точностью $\epsilon = 10^{-3}$ найти точку минимума квадратичной функции

$$f(x,y) = 8x^2 + 5y^2 - 4xy + 8\sqrt{5}(x+2y) + 64.$$

Начальное приближение: $X^0 = (-\sqrt{5}, 0)^T$, $X^0 = (0, 2\sqrt{5})^T$.

Ответ: $(-2.24, -4.47)^T$.

Лабораторная работа № 4

Квазиньютоновские методы

Цель работы

Изучение квазиньютоновских методов:

- метода ДФП (Давидона[1] — Флетчера — Пауэлла[2]);
- метода БПГШ (Бройдена[3] — Пауэлла — Гольдфарба — Шенно[4]), в некоторых источниках именуемый методом БФШ (Бройдена — Флетчера — Шенно) [1];
- метода Пауэлла;
- метода Мак-Кормика.

Содержание работы

Рассмотрим *задачу безусловной минимизации* (1.1). Предполагаем, что целевая функция дважды дифференцируема.

Требуется выполнить следующие действия.

1. Для решения задачи (1.1) реализовать в системе Wolfram Mathematica алгоритмы следующих квазиньютоновских методов:
   - метода ДФП (Давидона — Флетчера — Пауэлла);

[1] Уи́льям Ку́пер Давидо́н (1927–2013) — американский физик и математик, активист движения за мир.

[2] Майкл Джеймс Дэ́вид Па́уэлл (1936–2015) — британский математик, известен своими работами в области численного анализа, нелинейной оптимизации и теории аппроксимации.

[3] Чарльз Джордж Бро́йден (1933–2011) — британский математик и физик, специалист по методам оптимизации и численной линейной алгебре.

[4] Дэ́вид Ф. Шенно́ (родился в 1936 г.) — американский математик, специалист по методам оптимизации и исследованию операций.

- метода БПГШ (Бройдена — Пауэлла — Гольдфарба — Шенно);
- метода Пауэлла;
- метода Мак-Кормика.

2. На тестовых примерах продемонстрировать работоспособность программ.

3. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, траекторию поиска точки минимума, график убывания нормы невязки.

4. Экспериментально исследовать свойства методов. Как влияет выбор начального приближения на сходимость методов? Как влияет заданная точность ϵ на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета?

5. Провести сравнение квазиньютоновских методов с методом Ньютона. Чем отличаются графики убывания нормы невязки? Как отличается скорость сходимости методов при минимизации овражных функций? При минимизации квадратичной функции? Сравнить трудоемкость методов.

Методические указания

Траекторию поиска точки минимума в ***квазиньютоновских методах*** строят по рекуррентной формуле (3.3) с шагом спуска, значение которого находится из решения задачи одномерной минимизации (3.4). Однако для определения направления спуска p^k не требуется решать систему линейных алгебраических уравнений (3.2). Вместо этого вычисляется результат матрично-векторного умножения:

$$p^k = A_k w^k. \tag{4.1}$$

Здесь A_k — положительно определенная матрица порядка n, которая на каждой итерации вычисляется по определенному алгоритму, причем последовательность матриц $\{A_k\}$ при $k \to \infty$ сходится к матрице $H^{-1}(X^*)$, обратной к матрице Гессе в точке минимума X^* целевой функции $f(X)$. По этой причине на последних итерациях траектория поиска точки минимума квазиньютоновскими методами оказывается похожей на траекторию поиска точки минимума методом Ньютона.

Для построения последовательности матриц $\{A_k\}$ используют рекуррентное соотношение

$$A_{k+1} = A_k + \Delta A_k, \tag{4.2}$$

где ΔA_k — положительно определенная матрица порядка n, называемая ***поправочной***. На первой итерации полагаем, что $A_1 = I$, где I — единичная матрица порядка n.

Существует несколько формул для вычисления поправочной матрицы. Так, в ***методе Давидона — Флетчера — Пауэлла*** ΔA_k вычисляется по формуле

$$\Delta A_k = -\frac{\Delta X^k (\Delta X^k)^T}{(\Delta w^k, \Delta X^k)} - \frac{A_k \Delta w^k (\Delta w^k)^T A_k^T}{(A_k \Delta w^k, \Delta w^k)},$$

где $\Delta X^k = X^k - X^{k-1}$, $\Delta w^k = w^{k+1} - w^k$. Пример траектории поиска точки минимума методом Давидона — Флетчера — Пауэлла представлен на рис. 4.1.

В ***методе Бройдена — Пауэлла — Гольдфарба — Шенно*** ΔA_k вычисляется по формуле

$$\Delta A_k = -\frac{\Delta X^k (\Delta X^k)^T}{(\Delta w^k, \Delta X^k)} - \frac{A_k \Delta w^k (\Delta w^k)^T A_k^T}{(A_k \Delta w^k, \Delta w^k)} + (A_k \Delta w^k, \Delta w^k) r^k (r^k)^T,$$

где

$$r^k = \frac{A_k \Delta w^k}{(A_k \Delta w^k, \Delta w^k)} - \frac{\Delta X^k}{(\Delta w^k, \Delta X^k)}$$

Пример траектории поиска точки минимума методом Бройдена — Флетчера — Шенно представлен на рис. 4.2.

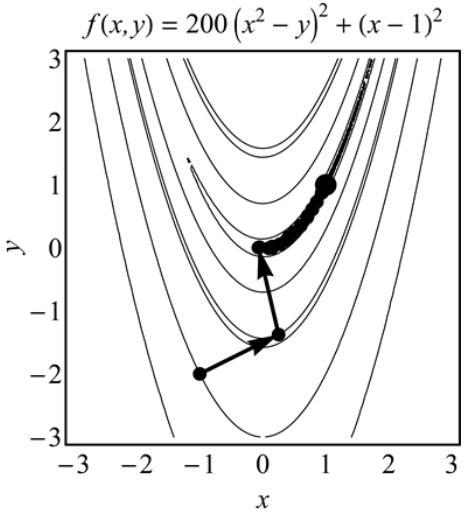


Рис. 4.1. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Давидона — Флетчера — Пауэлла

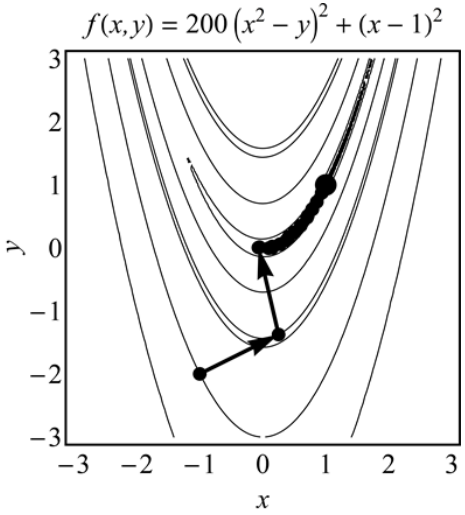


Рис. 4.2. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Бройдена — Пауэлла — Гольдфарба — Шенно

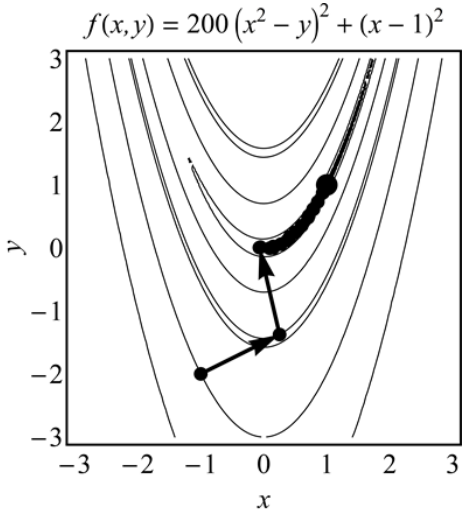


Рис. 4.3. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Пауэлла

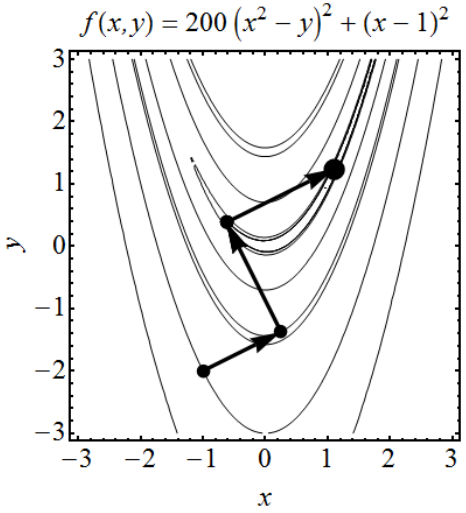


Рис. 4.4. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Мак-Кормика

В ***методе Пауэлла***

$$\Delta A_k = -\frac{\Delta\widetilde{X}^k(\Delta\widetilde{X}^k)^T}{(\Delta w^k, \Delta\widetilde{X}^k)},$$

где $\Delta\widetilde{X}^k = \Delta X^k + A_k\Delta w^k$. Пример траектории поиска точки минимума методом Пауэлла представлен на рис. 4.3.

В ***методе Мак-Кормика*** ΔA_k вычисляется по формуле

$$\Delta A_k = -\frac{\Delta X^k(\Delta X^k)^T}{(\Delta w^k, \Delta X^k)} - \frac{A_k\Delta w^k(\Delta X^k)^T}{((\Delta X^k)^T, \Delta w^k)}.$$

Пример траектории поиска точки минимума методом Мак-Кормика представлен на рис. 4.4.

Содержание отчета

1. Постановка задачи.
2. Краткое описание методов решения.
3. Ответы на контрольные вопросы.
4. Результаты вычислительных экспериментов: топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки.
5. Анализ результатов. Исследование зависимости сходимости методов от начального приближения. Сравнение эффективности квазиньютоновских методов с методом Ньютона.
6. Листинг кода программы.

Контрольные вопросы

1. В чем состоит основная идея квазиньютоновских методов?
2. Как определяется направление спуска в квазиньютоновских методах?
3. С каким методом совпадает первая итерация любого квазиньютоновского метода?

4. Как строится последовательность матриц $\{A_k\}$ в квазиньютоновских методах?
5. Корректно ли сравнивать квазиньютоновские методы с методом Ньютона только по числу итераций?

Примеры заданий

1. При помощи квазиньютоновских методов с заданной точностью $\epsilon = 10^{-3}$ найти точку минимума функции Розенброка

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

при $\alpha = 1$, $\alpha = 50$, $\alpha = 1000$. Начальное приближение полагать равным $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи квазиньютоновских методов с заданной точностью $\epsilon = 10^{-3}$ найти точку минимума квадратичной функции

$$f(x,y) = 11x^2 + 3y^2 + 6xy - 2\sqrt{10}(x - 3y) - 22.$$

Начальное приближение: $X^0 = (\sqrt{10}, 0)^T$, $X^0 = (0, \sqrt{10})^T$.

Ответ: $(1.58, -4.74)^T$.

Лабораторная работа № 5

Методы спуска

Цель работы

Изучение методов спуска (нулевого порядка):

- метода циклического покоординатного спуска;
- модификации метода Хука — Дживса, в которой для выбора направления спуска используется метод циклического покоординатного спуска, а ускоряющий множитель выбирается как шаг спуска в методе наискорейшего спуска;
- метода Розенброка;
- метода Пауэлла.

Содержание работы

Рассмотрим *задачу безусловной минимизации* (1.1).
Требуется выполнить следующие действия.

1. Для решения задачи (1.1) реализовать в системе Wolfram Mathematica алгоритмы следующих методов спуска:
    - метода циклического покоординатного спуска;
    - модификации метода Хука — Дживса, в которой для выбора направления спуска используется метод циклического покоординатного спуска, а ускоряющий множитель выбирается как шаг спуска в методе наискорейшего спуска;
    - метода Розенброка;
    - метода Пауэлла.

2. На тестовых примерах продемонстрировать работоспособность программ.

3. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, траекторию поиска точки минимума, график убывания нормы невязки.

4. Экспериментально исследовать свойства методов. Как влияет заданная точность ϵ и критерий останова на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета? Сравнить трудоемкость методов.

5. Провести сравнение методов спуска с методами первого и второго порядка. Чем отличаются графики убывания нормы невязки? Как отличается скорость сходимости методов при минимизации овражных функций? При минимизации квадратичной функции? Сравнить трудоемкость методов.

Методические указания

Для решения задачи минимизации (1.1) построим методы прямого поиска. Это означает, что в итерационном процессе не используются антиградиент и матрица Гессе целевой функции. Единственная информация о целевой функции, используемая в методах прямого поиска, — значения целевой функции в точках, принадлежащих допустимому множеству.

В методах спуска для определения новой точки траектории поиска точки минимума используется следующее рекуррентное соотношение:

$$X^k = X^{k-1} + \sum_{i=1}^{m} \alpha_i u^i(k), \quad k = 1, 2, \ldots, \quad m \geq n. \qquad (5.1)$$

Здесь $\{u^i(k)\}$ — система векторов, являющихся направлениями спуска на k-й итерации; α_i — шаг спуска по направлению $u^i(k)$, который находится из решения задачи одномерной минимизации:

$$\varphi_{k,i}(\alpha) \to \min,\ \varphi_{k,i}(\alpha) = f\left(X^{k-1} + \sum_{j=1}^{i-1} \alpha_j u^j(k) + \alpha u^i(k)\right),\ \alpha \in [0, \hat{\alpha}].$$

Если в качестве направлений спуска на каждой итерации использовать векторы стандартного базиса в $\mathbb{R}^n$, т.е. $u^i(k) = e^i$, $i = \overline{1,n}$, $m = n$, получим **метод циклического покоординатного спуска**. Пример траектории поиска точки минимума при использовании этого метода представлен на рис. 5.1.

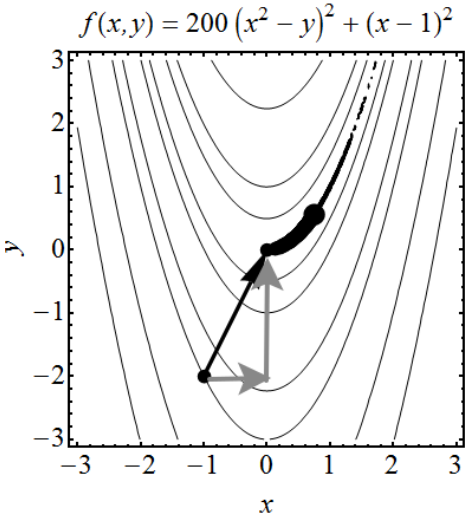


Рис. 5.1. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом циклического покоординатного спуска

Если дополнить систему векторов, являющихся направлениями спуска в методе циклического покоординатного спуска, вектором

$$u^{n+1}(k) = \sum_{i=1}^{n} \alpha_i e^i,$$

получим ***метод Хука — Дживса***. Пример траектории поиска точки минимума при использовании этого метода представлен на рис. 5.2.

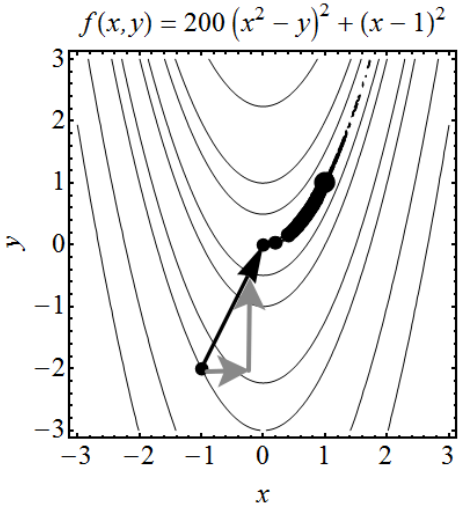


Рис. 5.2. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Хука — Дживса

Если $m = n$, но вместо векторов стандартного базиса использовать векторы других ортонормированных базисов, получаем ***метод Розенброка***. На первой итерации метода Розенброка полагают, что $u^i(1) = e^i$, $i = \overline{1,n}$. После вычисления всех шагов спуска и определения новой точки траектории поиска точки минимума по формуле (5.1), определяют первое направление спуска для следующей итерации по формуле

$$u^1(k+1) = \sum_{i=1}^{n} \alpha_i u^i(k).$$

Остальные направления спуска $u^i(k+1)$, $i = \overline{2,n}$ находят при помощи процесса ортогонализации Грамма — Шмидта. Пример траектории поиска точки минимума при использовании метода Розенброка представлен на рис. 5.3.

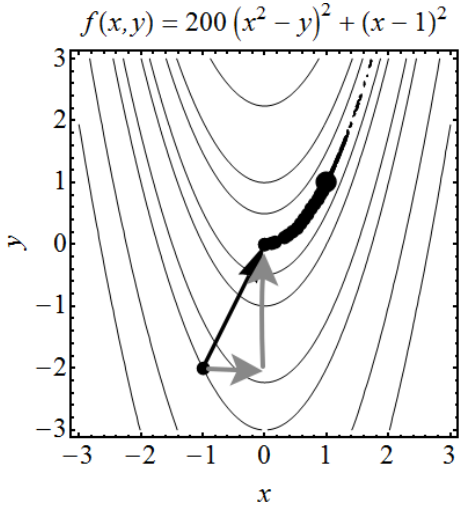


Рис. 5.3. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Розенброка

В ***методе Пауэлла*** построение системы векторов $\{u^i(k)\}$ на одной итерации выполняется n раз. Обозначим как $u^i(k,j)$ вектор, являющийся i-м направлением спуска в j-м базисе на k-й итерации $(i = \overline{1,n+1},\ (j = \overline{1,n}))$. На первой итерации в качестве первых n векторов первого базиса выбираем векторы стандартного базиса: $u^i(1,1) = e^i,\ i = \overline{1,n}$. Последний вектор j-й системы векторов на k-й итерации определяется как

$$u^{n+1}(k,j) = \sum_{i=1}^{n} \alpha_i u^i(k,j).$$

После этого первые n векторов $(j+1)$-го базиса строятся следующим образом:

$$u^i(k,j+1) = u^{i+1}(k,j), \quad i = \overline{1,n}.$$

Пример траектории поиска точки минимума при использовании метода Пауэлла представлен на рис. 5.4.

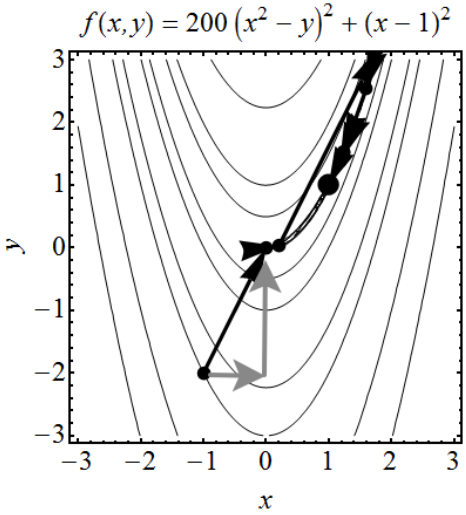


Рис. 5.4. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Пауэлла

Содержание отчета

1. Постановка задачи, номер варианта и исходные данные.
2. Ответы на контрольные вопросы.
3. Тестовые примеры, демонстрирующие работоспособность программы.
4. Результаты вычислительных экспериментов: топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки.
5. Анализ результатов: сравнение решений, полученных разными методами при различных критериях останова; исследование точности и эффективности рассмотренных методов.
6. Листинг кода программы.

Контрольные вопросы

1. Какой информацией о целевой функции можно пользоваться при использовании методов прямого поиска?

2. Какие критерии останова могут быть использованы в алгоритмах методов прямого поиска?
3. Как представляется направление спуска в методе циклического покоординатного спуска?
4. Как в методе циклического покоординатного спуска выбираются шаги спуска?
5. Описать алгоритм метода циклического покоординатного спуска. Как его можно модифицировать?
6. Описать модификацию метода Хука — Дживса, в которой для выбора направления спуска используется метод циклического покоординатного спуска, а ускоряющий множитель выбирается как шаг спуска в методе наискорейшего спуска.
7. Привести пример графика функции двух переменных, точки x^{k-1} на нем и ортонормированного базиса (ОНБ) $\{p_0^k, p_1^k\}$, для которых представление направления спуска в виде суммы векторов этого ОНБ дает лучшие результаты, чем представление направления спуска в виде суммы векторов стандартного базиса.
8. В чем различие между алгоритмами методов Розенброка и Пауэлла?
9. Сравнить рассматриваемые методы по числу задач одномерной минимизации, решаемых на одной итерации.
10. Корректно ли сравнивать рассматриваемые методы только по числу итераций?

Примеры заданий

1. При помощи методов спуска с точностью $\epsilon = 10^{-3}$ найти точку минимума функции Розенброка

$$f(x, y) = (x^2 - y)^2 + (x - 1)^2.$$

Начальное приближение $X^0 = (-1, -2)^T$. Исследовать зависимость числа итераций от критерия останова.

Ответ: $(1, 1)^T$.

2. При помощи методов спуска с точностью $\epsilon = 10^{-3}$ найти точку минимума квадратичной функции

$$f(x, y) = 10x^2 + 7y^2 - 4xy - 4\sqrt{5}(5x - y) - 16.$$

Начальное приближение: $X^0 = (0, -\sqrt{5})^T$, $X^0 = (0, \sqrt{10})^T$.

Ответ: $(2.24, 0.00)^T$.

Лабораторная работа № 6

Методы симплексного поиска

Цель работы

Изучение методов симплексного поиска:

- метода симплексного поиска при помощи регулярного симплекса (метода Спендлея — Хекста — Химсфорта);
- метода Нелдера[1] — Мида (простейшей схемы метода симплексного поиска при помощи нерегулярного симплекса).

Содержание работы

Рассмотрим *задачу безусловной минимизации* (1.1).

Требуется выполнить следующие действия.

1. Для решения задачи (1.1) реализовать на языке C++ в программе-шаблоне `optimization_methods` алгоритмы следующих методов:
   - метода симплексного поиска при помощи регулярного симплекса;
   - метода Нелдера — Мида (простейшей схемы метода симплексного поиска при помощи нерегулярного симплекса).

2. На тестовых примерах продемонстрировать работоспособность программ.

[1] Джон Э́шворт Не́лдер (1924–2010) — британский статистик, удостоен премии Пирсона за книгу «Обобщенные линейные модели».

3. Результаты решения задач выгружать в файл. Данные из этих файлов загружать в систему Wolfram Mathematica при помощи следующих команд:

```
(*загружаем данные из файла*)
data = Import["simplex.dat"];
(*массив точек траектории поиска*)
points = data[[All,{1,2}]];
(*массив координат вершин симплексов*)
simplexes = Partition[Partition[
  Flatten[data[[All,{3,4,5,6,7,8}]]],2],3];
(*массив значений функции в точках
                                  траектории*)
fValues = data[[All,9]];
(*массив значений нормы невязки*)
norm = data[[All,10]];
```

4. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, траекторию поиска точки минимума (вместе с симплексами), график убывания нормы невязки. Для построения симплексов на фоне топографии функции можно воспользоваться следующими командами:

```
(*отрисовываем топографию целевой функции*)
Show[ContourPlot[f[{x,y}],...],
 (*отрисовываем массив симплексов*)
 Table[Graphics[{FaceForm[Pink],
   EdgeForm[Directive[Thick, Dashed, Blue]],
   Polygon[simplexes[[i]]]}],
                       {i,1,Length[norm]}],
 (*отрисовываем найденную точку минимума*)
 Graphics[{Black, PointSize[0.03],
                    Point[points[[-1]]]}]]
```

5. Экспериментально исследовать свойства методов. Как влияет выбор начального приближения, начальной длины ребра симплекса на сходимость методов? Как влияет заданная точность ϵ на скорость сходимости методов? Как

влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета? Исследовать влияние коэффициентов редукции, отражения, сжатия и растяжения на сходимость методов.

6. Провести сравнение методов симплексного поиска с изученными ранее методами оптимизации. Чем отличаются траектории поиска точки минимума? Как отличается скорость сходимости методов при минимизации овражных функций? При минимизации квадратичной функции?

Методические указания

Для решения задачи минимизации (1.1) построим методы прямого поиска, отличающиеся от методов спуска. Исследуем значения целевой функции в окрестности начального приближения X^0. Для этого построим многогранник с центром в точке X^0 и проанализируем значения целевой функции в вершинах этого многогранника. При этом число вершин многогранника должно быть минимально возможным, поскольку операция вычисления значения целевой функции в точке сама по себе может оказаться весьма трудоемкой. Таким образом, в $\mathbb{R}^n$ будем использовать многогранник с $n+1$-й вершиной, т.е. ***симплекс***.

Анализ значений целевой функции в вершинах симплекса должен показывать направление перемещения симплекса в пространстве, приводящее к точке минимума целевой функции. Сначала предположим, что в процессе перемещения симплекс не деформируется и остается ***регулярным***, т.е. все вершины симплекса на каждой итерации остаются равноудаленными от центра симплекса. Такой метод называется ***методом симплексного поиска при помощи регулярного симплекса***.

Координаты вершин $X_j^{1,i}$ $(i=\overline{1,n+1},\ j=\overline{1,n})$ регулярного симплекса с центром в точке X^0 с координатами X_j^0 $(j=\overline{1,n})$

и длиной ребра l определяются следующим образом:

$$X_j^{1,i} = \begin{cases} X_j^0, & j < i-1, \\ X_j^0 + l\sqrt{\dfrac{j}{2(j+1)}}, & j = i-1, \\ X_j^0 - \dfrac{l}{\sqrt{2j(j+1)}}, & j > i-1. \end{cases}$$

В дальнейших рассуждениях *будем использовать* ***правильную нумерацию*** *вершин симплекса* — нумерацию, при которой выполняются неравенства

$$f(X^{k,1}) \le \ldots \le f(X^{k,i}) \le \ldots \le f(X^{k,n+1}).$$

Будем использовать следующую стратегию перемещения симплекса в пространстве. Сначала ***отразим вершину*** симплекса $f(X^{k,n+1})$, в которой значение целевой функции наибольшее:

$$X^{k+1,n+1} = 2X^{k,C} - X^{k,n+1}, \quad X^{k,C} = \frac{1}{n}\sum_{i=1}^{n} X^{k,i}. \tag{6.1}$$

Если после отражения оказалось, что $f(X^{k+1,n+1}) \ge f(X^{k,n+1})$, то отражение признается неудачным, и происходит возврат к исходному симплексу с последующей его ***редукцией***:

$$X^{k+1,i} = X^{k,i} + \delta(X^{k,i} - X^{k,1}), i = \overline{2, n+1}. \tag{6.2}$$

Здесь $\delta \in (0,1)$ — ***коэффициент редукции***. В результате редукции длина ребра симплекса уменьшается в $1/\delta$ раз и симплекс сжимается к вершине $X^{k,1}$, в которой значение целевой функции меньше, чем в других вершинах симплекса. Когда длина ребра симплекса окажется меньше заданной точности ϵ, процесс поиска точки минимума прекращается. Пример траектории поиска точки минимума методом симплексного поиска при помощи регулярного симплекса представлен на рис. 6.1.

Как видно из рис. 6.1, регулярный симплекс малоэффективен при поиске точки минимума целевой функции с ярко

выраженной овражной структурой. Эффективность симплексного поиска для таких целевых функций можно повысить, если допустить деформирования симплекса, за счет чего симплекс станет ***нерегулярным*** и сможет «вытянуться» вдоль оси «оврага». Рассмотрим простейший вариант метода симплексного поиска при помощи нерегулярного симплекса — ***метод Нелдера — Мида***.

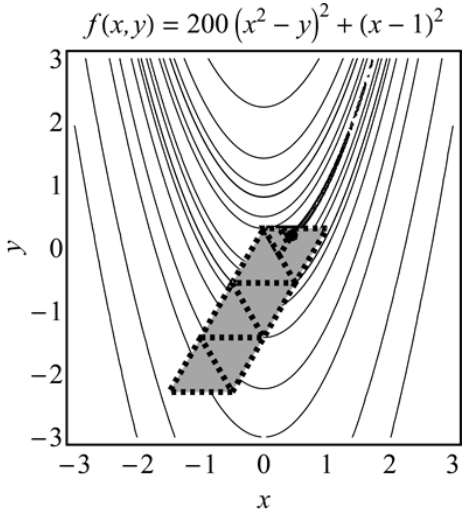


Рис. 6.1. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом симплексного поиска при помощи регулярного симплекса

Как и в методе симплексного поиска при помощи регулярного симплекса, сначала происходит отражение вершины $X^{k,n+1}$, но вместо формулы (6.1) используется следующая формула:

$$X^{k+1,n+1} = X^{k,C} + \alpha(X^{k,C} - X^{k,n+1}), \tag{6.3}$$

где $\alpha > 0$ — ***коэффициент отражения***. Если выполняется неравенство $f(X^{k+1,n+1}) < f(X^{k,1})$, т.е. если отражение оказалось очень удачным, пробуют провести ***растяжение симплекса***:

$$X_*^{k+1,n+1} = X^{k,C} + \beta(X^{k+1,n+1} - X^{k,C}), \tag{6.4}$$

где $\beta > 1$ — ***коэффициент растяжения***. Если выполняется неравенство $f(X_*^{k+1,n+1}) < f(X^{k,1})$, то $X^{k+1,n+1} = f(X_*^{k+1,n+1})$. Если

же после отражения $f(X^{k+1,n+1}) > f(X^{k,n})$, пробуют провести сжатие симплекса:

$$X_{**}^{k+1,n+1} = X^{k,C} + \gamma(\tilde{X}^{k+1,n+1} - X^{k,C}). \tag{6.5}$$

Здесь $\gamma \in (0,1)$ — ***коэффициент сжатия***; $\tilde{X}^{k+1,n+1} = X^{k,n+1}$, если отражение неудачно (т.е. $f(X^{k+1,n+1}) > f(X^{k,n+1})$), если же отражение удачно, то $\tilde{X}^{k+1,n+1} = X^{k+1,n+1}$. Если выполняется неравенство $f(X_{**}^{k+1,n+1}) < f(X^{k,n+1})$, то $X^{k+1,n+1} = f(X_{**}^{k+1,n+1})$, иначе симплекс редуцируют по формуле (6.2). В качестве условия останова можно использовать неравенство

$$\frac{1}{n+1}\sum_{i=1}^{n+1}(f(X^{k,i}) - f(X^{k,C}))^2 < \epsilon^2.$$

Пример траектории поиска точки минимума методом Нелдера — Мида представлен на рис. 6.2.

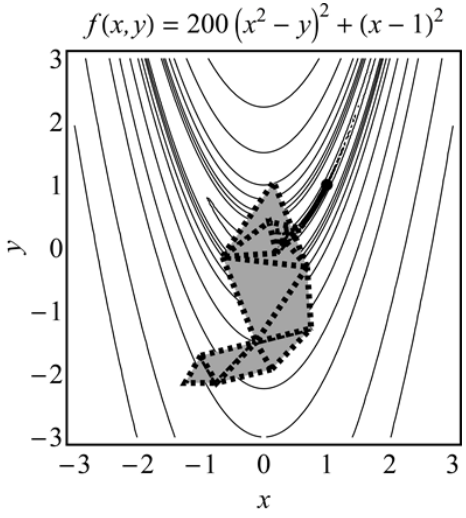


Рис. 6.2. Траектория поиска точки минимума функции Розенброка при $\alpha = 200$ методом Нелдера — Мида

Содержание отчета

1. Постановка задачи.

2. Краткое описание методов решения.
3. Ответы на контрольные вопросы.
4. Тестовые примеры, демонстрирующие работоспособность программы.
5. Результаты вычислительных экспериментов: топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки.
6. Анализ результатов. Сравнение решений, полученных при различных значениях начальной длины ребра симплекса и коэффициентов редукции, отражения, сжатия, растяжения.
7. Листинг кода программы.

Контрольные вопросы

1. В чем основная идея поиска точки минимума при помощи образца (шаблона)?
2. Дать определение симплекса, регулярного симплекса. Привести примеры в $\mathbb{R}^2$, $\mathbb{R}^3$.
3. Почему на практике в качестве шаблона чаще всего используют не n-мерный куб, а симплекс?
4. Описать алгоритм метода симплексного поиска при помощи регулярного симплекса.
5. Описать алгоритм метода Нелдера — Мида.
6. Какие критерии останова могут быть использованы для методов симплексного поиска?
7. Для каких функций метод симплексного поиска при помощи регулярного симплекса неэффективен? За счет чего для этих функций хорошо работает метод Нелдера — Мида?
8. За счет изменения каких параметров можно уменьшить число итераций в методах симплексного поиска?

Примеры заданий

1. При помощи методов симплексного поиска с точностью $\epsilon = 10^{-3}$ найти точку минимума функции Розенброка

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

при $\alpha = 1$, $\alpha = 50$, $\alpha = 1000$. Начальное приближение полагать равным $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи методов симплексного поиска с точностью $\epsilon = 10^{-3}$ найти точку минимума квадратичной функции

$$f(x,y) = 11x^2 + 3y^2 + 6xy - 2\sqrt{10}(x - 3y) - 22.$$

Начальное приближение: $X^0 = (\sqrt{10}, 0)^T$, $X^0 = (0, \sqrt{10})^T$.

Ответ: $(1.58, -4.74)^T$.

Лабораторная работа № 7

Прямые численные методы нелинейного программирования

Цель работы

Изучение численных методов нелинейного программирования, относящихся к прямым методам (методам спуска):

- метода условного градиента;
- метода проектирования точки на множество;
- метода проекции антиградиента.

Содержание работы

Рассмотрим *задачу нелинейного программирования*

$$f:\ \mathbb{R}^n \to \mathbb{R}, \quad f(X) \to \min, \quad X \in \Omega \subset \mathbb{R}^n, \tag{7.1}$$

где f — целевая функция, $\Omega \neq D(f)$ — допустимое множество, т.е. есть дополнительные ограничения $g_i(X)$ на Ω, причем среди этих ограничений есть нелинейные.

Требуется выполнить следующие действия.

1. Для решения задачи (7.1) реализовать в системе Wolfram Mathematica алгоритмы следующих методов спуска:
   - метода условного градиента;
   - метода проектирования точки на множество;
   - метода проекции антиградиента.

2. На тестовых примерах продемонстрировать работоспособность программ.

3. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, допустимое множество, траекторию поиска точки минимума, график убывания нормы невязки.

4. Экспериментально исследовать свойства методов. Как влияет заданная точность ϵ на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета? Сравнить трудоемкость методов. Для каких методов в ходе решения задачи выполнялось проектирование точки на допустимое множество?

Методические указания

Для решения задачи (7.1) необходимо построить такой метод, при котором точки траектории поиска точки минимума целевой функции не выходят за границу допустимого множества. Например, можно использовать изученные ранее численные методы решения задач безусловной оптимизации, с той лишь разницей, что на каждой итерации проверяется, принадлежит ли точка X^{k+1} допустимому множеству Ω, и, если не принадлежит, точка X^{k+1} проектируется на границу допустимого множества $\partial\Omega$. Полученный метод называется ***методом проектирования точки на множество***. Пример траектории поиска точки минимума методом проектирования точки на множество представлен на рис. 7.1.

Теперь рассмотрим применяемый для решения задачи (7.1) ***метод условного градиента***. На k-й итерации рассмотрим приращение целевой функции $f(X)$ в окрестности точки X^{k-1}:

$$\Delta f_k = f(X) - f(X^{k-1}) = (\text{grad} f(X^{k-1}), X - X^{k-1}) + o(||X - X^{k-1}||).$$

Тогда главная линейная часть приращения целевой функции, дающая ее линейное приближение, равна

$$f_k(X) = (\text{grad} f(X^{k-1}), X - X^{k-1}).$$

В качестве направления спуска p^k выберем направление, задаваемое вектором $\tilde{X}^k - X^{k-1}$, где $\tilde{X}^k$ находится следующим образом:

$$\tilde{X}^k = \min_{X\in\Omega} f_k(X) = \min_{X\in\Omega} (\operatorname{grad} f(X^{k-1}), X - X^{k-1}).$$

После этого следующую точку траектории поиска точки минимума целевой функции на допустимом множестве находим из рекуррентного соотношения

$$X^k = X^{k-1} + \varkappa_k p^k = X^{k-1} + \varkappa_k(\tilde{X}^k - X^{k-1}), \quad \varkappa_k \in [0,1]. \qquad (7.2)$$

Как и в методе наискорейшего спуска $\varkappa_k$ выбирают так, чтобы

$$\varphi_k(\varkappa) \to \min, \quad \varphi_k(\varkappa) = f(X^{k-1} + \varkappa(\tilde{X}^k - X^{k-1})), \quad \varkappa \in [0, \hat{\varkappa}].$$

Пример траектории поиска точки минимума методом условного градиента представлен на рис. 7.2.

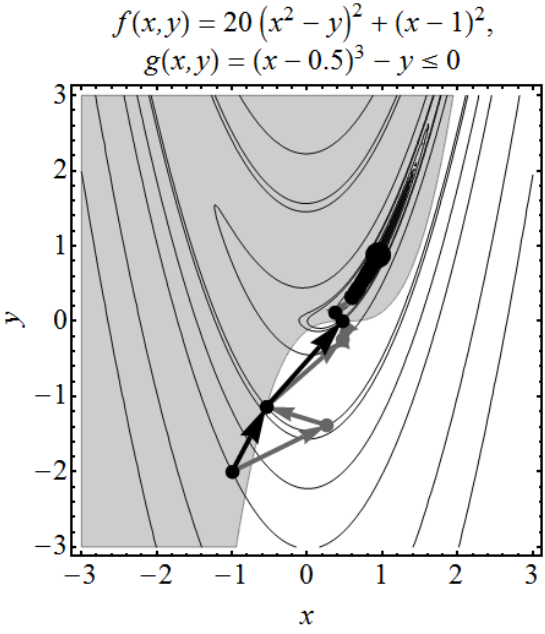


Рис. 7.1. Траектория поиска точки минимума функции Розенброка при $\alpha = 20$ методом проектирования точки на множество

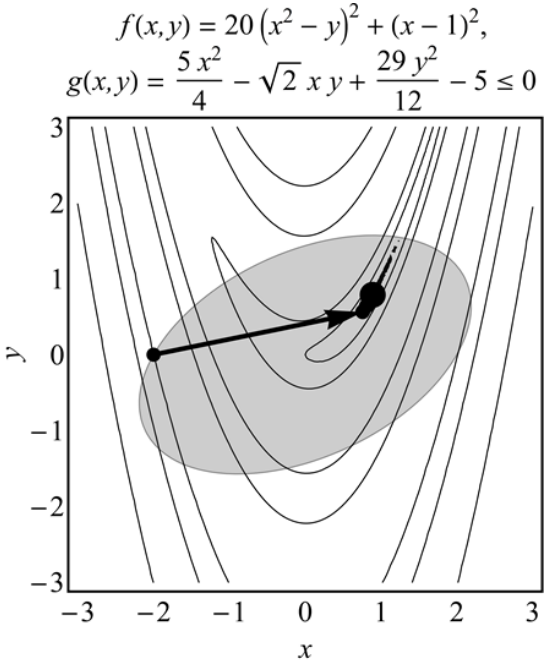


Рис. 7.2. Траектория поиска точки минимума функции Розенброка при $\alpha = 20$ методом условного градиента

В ***методе проекции антиградиента*** для определения точек траектории поиска точки минимума целевой функции используется следующее рекурректное соотношение:

$$X^k = X^{k-1} + \varkappa_k p^k = X^{k-1} + \varkappa_k (P^* w^k), \quad \varkappa_k > 0. \tag{7.3}$$

Здесь P^* — проекционная матрица, а $\varkappa_k$ как и в методе наискорейшего спуска выбирают так, чтобы

$$f(X^k) = f(X^{k-1} + \varkappa_k (P^* w^k)) \to \min.$$

Матрицу P^* строят следующим образом:

$$P^* = I - A_k^T (A_k A_k^T)^{-1} A_k.$$

Здесь матрица A_k — матрица, составленная из матриц-строк $(\mathrm{grad} g_i(X^{k-1}))^T$; g_i — дифференцируемые функции, которые

задают границу допустимого множества ($i = \overline{1,m}$), такие что $g_i(X^{k-1}) = 0$ (активные ограничения).

В случае нелинейных ограничений (функций g_i) матрица P^* не является проекционной, поэтому необходимо на каждой итерации проверять, принадлежит ли точка X^{k+1} допустимому множеству Ω, и, если не принадлежит, спроектировать ее на границу допустимого множества $\partial\Omega$, как это показано на рис. 7.3.

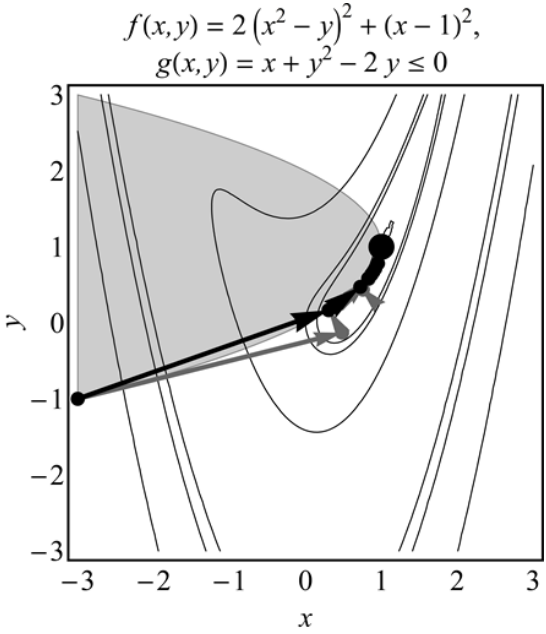


Рис. 7.3. Траектория поиска точки минимума функции Розенброка при $\alpha = 2$ методом проекции антиградиента

В случае линейных ограничений (линейных функций g_i) матрица P^* является проекционной, поэтому нет необходимости сочетать метод проекции антиградиента с методом проектирования точки на множество.

Содержание отчета

1. Постановка задачи.

2. Краткое описание методов решения.
3. Ответы на контрольные вопросы.
4. Тестовые примеры, демонстрирующие работоспособность программы.
5. Результаты расчетов (топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки) и их анализ.
6. Анализ результатов. Как влияет заданная точность ϵ на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета? Сравнить трудоемкость методов. Для каких методов в ходе решения задачи выполнялось проектирование точки на допустимое множество?
7. Листинг кода программы.

Контрольные вопросы

1. Сформулируйте общую постановку задачи нелинейного программирования.
2. Почему нельзя решать задачи нелинейного программирования методами безусловной минимизации?
3. Записать формулу расчета x^k для метода условного градиента.
4. Из какого условия находится вспомогательное приближение в методе условного градиента?
5. Что называется проекцией точки на множество?
6. Описать алгоритм метода проектирования точки на множество.
7. Какая матрица называется идемпотентной?
8. Какая матрица называется проекционной?
9. Какие свойства проекционных матриц используются в методе проекции антиградиента?

10. Записать формулу расчета x^k для метода проекции антиградиента.
11. Для каких ограничений на допустимое множество x^k, найденное методом проекции антиградиента, может не принадлежать допустимому множеству?

Примеры заданий

1. При помощи метода условного граниента с точностью $\epsilon = 10^{-3}$ найти в области

$$g(x,y) = -5 + \frac{29x^2}{12} + \frac{5y^2}{4} - \sqrt{2}xy \le 0$$

точку минимума функции Розенброка

$$f(x,y) = (x^2 - y)^2 + (x-1)^2.$$

Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи метода проектирования точки на множество с точностью $\epsilon = 10^{-3}$ найти в области

$$g(x,y) = \left(x - \frac{1}{2}\right)^3 - y \le 0$$

точку минимума функции Розенброка

$$f(x,y) = (x^2 - y)^2 + (x-1)^2.$$

Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

3. При помощи метода проекции антиградиента с точностью $\epsilon = 10^{-3}$ найти в области

$$g(x,y) = x + y^2 - 2y \leq 0$$

точку минимума функции Розенброка

$$f(x,y) = (x^2 - y)^2 + (x-1)^2.$$

Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

Лабораторная работа № 8

Методы последовательной безусловной минимизации

Цель работы

Изучение численных методов нелинейного программирования, относящихся к методам последовательной безусловной минимизации:

- метода внутренних штрафных (барьерных) функций;
- метода внешних штрафных функций.

Содержание работы

Рассмотрим *задачу нелинейного программирования* (7.1).
Требуется выполнить следующие действия.

1. Для решения задачи (7.1) реализовать в системе Wolfram Mathematica алгоритмы следующих методов спуска:
   - метода внутренних штрафных (барьерных) функций;
   - метода внешних штрафных функций.

2. На тестовых примерах продемонстрировать работоспособность программ.

3. Для каждой целевой функции и для каждого метода в системе Wolfram Mathematica построить топографию целевой функции, допустимое множество, топографию барьерных и штрафных функций, траекторию поиска точки минимума, график убывания нормы невязки.

4. Экспериментально исследовать свойства методов. Как влияет заданная точность ϵ и выбор штрафной функции на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета? Сравнить трудоемкость методов.

5. Сравнить с прямыми численными методами нелинейного программирования.

Методические указания

Методы штрафных функций

Методы последовательной безусловной минимизации предполагают преобразование общей задачи нелинейного программирования (7.1) в последовательность задач безусловной минимизации для сконструированных специальным образом вспомогательных функций. Формально задача (7.1) эквивалентна задаче безусловной минимизации

$$\min_{X \in \mathrm{R}^n} (f(X) + \delta_\Omega(X)), \tag{8.1}$$

где $\delta_\Omega(X)$ — индикаторная функция допустимого множества Ω:

$$\delta_\Omega(X) = \begin{cases} 0, & X \in \Omega, \\ +\infty, & X \notin \Omega. \end{cases}$$

Точки минимума исходной целевой функции $f(X)$ и функции $f(X) + \delta_\Omega(X)$ на допустимом множестве Ω совпадают, однако, поскольку индикаторная функция имеет точки разрыва на границе $\partial\Omega$ допустимого множества, для задачи (8.1) нельзя использовать изученные ранее численные методы безусловной оптимизации. Способом решения данной проблемы является построение последовательности дифференцируемых функций $\delta_k(X)$, таких что

$$\forall X \in \mathrm{R}^n \quad \lim_{k \to \infty} \delta_k(X) = \delta_\Omega(X).$$

Таким образом, исходную задачу нелинейного программирования (7.1) можно свести к последовательности задач безусловной минимизации:

$$f_k(X) = f(X) + \delta_k(X) \to \min, \quad X \in \tilde{\Omega} = D(f_k) \subset \mathbb{R}^n, \quad k = 1, 2, \ldots. \tag{8.2}$$

Методы, основанные на этом подходе, называются ***методами штрафных функций***. Функции $f_k(X) = f(X) + \delta_k(X)$ являются ***штрафными функциями***, а функции $\delta_k(X)$ — ***функциями штрафа***.

Методы внутренних штрафных (барьерных) функций

Для функции внутреннего штрафа выполняется следующее:

$$\delta_k(X) = \begin{cases} > 0, & X \in \Omega \setminus \partial\Omega, \\ +\infty, & X \notin \Omega \setminus \partial\Omega. \end{cases}$$

Часто в качестве функции внутреннего штрафа выбирают функции

$$\delta_k(X) = -r_k \sum_{i=1}^{m} \frac{1}{g_i(X)}, \quad X \in \Omega \setminus \partial\Omega$$

и

$$\delta_k(X) = -r_k \sum_{i=1}^{m} \ln(-g_i(X)), \quad X \in \Omega \setminus \partial\Omega,$$

где $r_k > 0$ — ***параметр штрафа***, $r_k \to 0$ при $k \to \infty$; $g_i(X)$ — функции, задающие границу допустимого множества. Эффективность метода внутренних штрафных функций зависит от выбора параметров штрафа и выбора функции внутреннего штрафа.

Методы внешних штрафных функций

Для функции внешнего штрафа выполняется следующее:

$$\delta_k(X) = \begin{cases} = 0, & X \in \Omega, \\ > 0, & X \notin \Omega. \end{cases}$$

Если допустимое множество Ω задано ограничениями типа неравенства, то в качестве функции внешнего штрафа обычно используют функцию

$$\delta_k(X) = r_k \sum_{i=1}^{m} (g_i^+(X))^2,$$

где $g_i^+(X) = \max\{0, g_i(X)\}$, $r_k > 0$ — параметр штрафа, $r_k \to +\infty$ при $k \to \infty$. Если Ω задано не только ограничениями типа неравенства, но и равенствами, то функцию внешнего штрафа задают в виде

$$\delta_k(X) = r_k \sum_{i=1}^{m} (g_i^+(X))^2 + r_k \sum_{i=1}^{m} (g_i(X))^2.$$

Содержание отчета

1. Постановка задачи.
2. Краткое описание методов решения.
3. Ответы на контрольные вопросы.
4. Тестовые примеры, демонстрирующие работоспособность программы.
5. Результаты расчетов (топография целевой функции, траектории поиска точки минимума, графики убывания нормы невязки) и их анализ.
6. Анализ результатов. Как влияет заданная точность ϵ и выбор штрафной функции на скорость сходимости методов? Как влияет «овражность» целевой функции на скорость сходимости методов? Как изменяется скорость сходимости методов в течение расчета? Сравнить трудоемкость методов.
7. Листинг кода программы.

Контрольные вопросы

1. В чем состоит основная идея использования индикаторной функции?

2. Почему вместо индикаторной функции на практике используется последовательность штрафных функций?
3. Перечислить отличия внутренних и внешних штрафных функций.
4. Привести примеры построения внутренних и внешних штрафных функций.

Примеры заданий

1. При помощи метода барьерных функций с заданной точностью $\epsilon = 10^{-3}$ найти в области

$$g(x,y) = \frac{3x^2}{4} + \frac{5x}{4} - 2 - y \le 0$$

точку минимума функции Розенброка

$$f(x,y) = (x^2 - y)^2 + (x-1)^2.$$

Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

2. При помощи метода внешних штрафных функций с точностью $\epsilon = 10^{-3}$ найти в области

$$g(x,y) = \frac{3x^2}{4} + \frac{5x}{4} - 2 - y \le 0$$

точку минимума функции Розенброка

$$f(x,y) = (x^2 - y)^2 + (x-1)^2.$$

Начальное приближение $X^0 = (-1, -2)^T$.

Ответ: $(1, 1)^T$.

ОПИСАНИЕ ПРОГРАММНОГО КОМПЛЕКСА `optimization_methods`

Программный комплекс `optimization_methods` является программой-шаблоном для написания нескольких лабораторных работ в рамках курса «Методы оптимизации». Программа написана на языке C++ с использованием парадигмы объектно-ориентированного программирования. В качестве среды разработки использовалась Microsoft Visual Studio. Проект можно открыть как в среде Microsoft Visual Studio 2008 (файл `optimization_methods.vcproj`), так и в среде Microsoft Visual Studio 2010 (файл `optimization_methods.vcxproj`).

В данном программном комплексе должны быть реализованы алгоритмы следующих методов оптимизации:

- метода градиентного спуска с дроблением шага;
- метода Ньютона;
- модификаций метода Ньютона (на примере спуска с дроблением шага);
- метода симплексного поиска при помощи регулярного симплекса;
- метода Нелдера — Мида (простейшей схемы метода симплексного поиска при помощи нерегулярного симплекса).

В качестве примера в `optimization_methods` реализован метод градиентного спуска с дроблением шага (в файле `gradient_descent_with_spalling_step_method.h`). Для реализации алгоритмов остальных методов созданы заготовки структур (в `newton_method.h` — для метода Ньютона, в `newton_with_spalling_step_method.h` — для модификации метода Ньютона, в `regular_simplex_method.h` — для метода симплексного поиска при помощи регулярного симплекса, в файле `non_regular_simplex_method.h` — для метода Нелдера — Мида).

Отметим, что в optimization_methods по умолчанию решается тестовая задача поиска минимума функции Розенброка

$$f(x,y) = \alpha(x^2 - y)^2 + (x-1)^2$$

при $\alpha = 1$. Целевая функция задана в структуре GoalFuncInform, производной от FuncInformInterface, а начальное приближение (по умолчанию $X^0 = (-1,-2)^T$) — в конструкторе управляющей структуры OptimizationTask (поле Vect X0). Изменить значение α можно в конструкторе структуры GoalFuncInform. При замене целевой функции необходимо учитывать, что для новой целевой функции нужно реализовать следующие методы:

- double func(double* point) — вычисление значения функции в точке point;
- double* antiGrad(double* point) — вычисление антиградиента функции в точке point;
- void calcHesse(double* point) — вычисление матрицы Гессе функции в точке point.

Идентификаторы методов определены в перечислении enum OptMethods в структуре OptimizationMethodInterface:

```
enum OptMethods // перечисление идентификаторов
{               // производных структур
  Unknown = -1, // метод не задан
  GradientDescentWithSpallingStep,// метод
       // градиентного спуска с дроблением шага
  Newton,       // метод Ньютона
  NewtonWithSpallingStep, // модификация метода
       // Ньютона (спуск с дроблением шага)
  RegularSimplex,// метод симплексного поиска
       // при помощи регулярного симплекса
  NonRegularSimplex // метод Нелдера-Мида
};
```

Метод решения задачи минимизации задается следующим образом. В главной функции приложения (функция main в файле optimization_methods.cpp) вызывается метод void

`solve(OptimizationMethodInterface::OptMethods_id)` экземпляра `task` управляющей структуры `OptimizationTask`, при этом в качестве параметра в данный метод передается идентификатор требуемого алгоритма:

```
task.solve(OptimizationMethodInterface::
                    OptMethods::идентификатор);
```

Так, чтобы задача многомерной оптимизации была решена методом Ньютона, в файле `optimization_methods.cpp` должно быть написано следующее:

```
#include "stdafx.h"
// vvp подключаем необходимые файлы
#include "optimization_task.h"
// vvp для избежания конфликтов имен необходимо
using namespace OPT_METH;  // использовать
                           // пространства имен

int main(int argc, char* argv[])
{ // vvp разрешить выводить русские буквы
  setlocale(LC_ALL,"Russian");
  // главная структура, в которой хранится
  // описание задачи и методы решения
  OptimizationTask task;
  // вызываем метод, в котором реализовано
  // решение задачи
  task.solve(OptimizationMethodInterface::
                           OptMethods::Newton);
  // vvp чтобы окно программы не закрылось сразу
  system ("PAUSE");          // после выполнения
  return 0;
}
```

Таким образом, при выполнении лабораторных работ требуется поставить нужный идентификатор алгоритма в главной функции приложении и заполнить в файле с заготовкой соответствующей структуры в методе

```
double* findMin(FuncInformInterface* _func,
                  double* _X0, const double _eps)
```

строки, отмеченные комментарием

```
// алгоритм
// ........
// алгоритм
```

Вывод результатов расчета в файл для всех методов уже реализован в функции `virtual void printRes()` базовой структуры `OptimizationMethodInterface` (название файла задается в функции `virtual void setFileName()`). Рекомендуется не забывать на каждой итерации также выводить информацию о состоянии счета в консоль, как это показано в файле `gradient_descent_with_spalling_step_method.h`:

```
iter++;// счетчик итераций прибавляем
// в консоль выводим промежуточную информацию,
// чтобы понимать, что программа что-то считает
std::cout << "норма невязки = " << residualNorm
                                        << "\n";
```

Для того чтобы избежать зацикливания, помимо условия останова лучше проверять и ограничение на максимальное число итераций:

```
while((residualNorm > m_eps) &&
                        (iter < MAX_ITER_COUNT))
{
    //...
}
```

Значение `MAX_ITER_COUNT` определено в базовой структуре `OptimizationMethodInterface`:

```
static const int MAX_ITER_COUNT = 500;
```

Структуры для работы с функциями, векторами, матрицами и симплексами в разработанном программном комплексе `optimization_methods` уже реализованы. К целевой функции из структуры, реализующей алгоритм метода оптимизации, можно обращаться по указателю `FuncInformInterface* m_func`. Таким образом, чтобы получить значение целевой функции `double curValF` в точке с координатами `Vect point`, необходимо написать следующий код:

```
curValF = m_func->func(&point);
```

Для вычисления матрицы Гессе целевой функции в точке с координатами `Vect point` необходимо написать

```
m_func->calcHesse(&point);
```

В структуре `NewtonMethod` для упрощения обращения к матрице Гессе хранится указатель базового типа

```
MatrixInterface* m_Hesse = m_func->pHesse;
```

Таким образом, чтобы проверить, является ли матрица Гессе положительно определенной, необходимо проверить, возвращает ли `true` метод

```
m_Hesse->isPositivelyDefinite();
```

Если матрица Гессе не является положительно определенной, и мы хотим прибавить к ней матрицу вида $\eta \cdot I$[1], так, чтобы получилась положительно определенная матрица, нужно вызвать метод

```
m_Hesse->supplementToPositiveDefinite();
```

Наконец, если нужно умножить матрицу, обратную к матрице Гессе, на вектор `Vect point` и записать результат в `Vect res`, следует написать следующий код:

```
m_Hesse->solve(&point, &res);
```

[1] I — единичная матрица того же порядка, что и данная матрица Гессе

Также поясним, как выполнять операции над экземплярами структуры `Vect`.

- Прибавление к `Vect res Vect point`:

  ```
  res += &point;
  ```

- Умножение всех элементов `Vect res` на действительное число `double coef`:

  ```
  res *= coef;
  ```

- Копирование `Vect res` в `Vect point`:

  ```
  res.set(&point);
  ```

- Вычисление скалярного произведения векторов `Vect point` и `Vect point2` с сохранением результата в переменную `double product`:

  ```
  product = point * &point2;
  ```

При работе с экземплярами структур `SimplexRegular` и `SimplexNonRegular`, реализующих работу с регулярным и нерегулярным симплексом соответственно, необходимо помнить, что для доступа как к полям структур, так и к методам, необходимо использовать оператор «.» и имя члена. Так для экземпляра структуры `SimplexRegular simplex`, чтобы вызвать метод `reduction()`, нужно написать

```
simplex.reduction();
```

Соответственно, чтобы присвоить переменной `double val` значение поля `fMin` этой структуры, нужно написать

```
val = simplex.fMin;
```

Подчеркнем, что структура `SimplexNonRegular` отнаследована от базовой структуры `SimplexRegular`. Таким образом, все открытые (`public`) и защищенные (`protected`) члены структуры `SimplexRegular` являются также членами структуры `SimplexNonRegular`. UML-диаграмма структур представлена на рис 8.1. Как принято в языке UML, открытые поля и методы структур отмечены знаком «+», защищенные — знаком «#», закрытые (`private`) — знаком «-». Необходимые для реализации алгоритмов метода симплексного поиска методы

структур `SimplexRegular` и `SimplexNonRegular` достаточно подробно прокомментированы в файлах `simplex_regular.h` и `simplex_non_regular.h` соответственно.

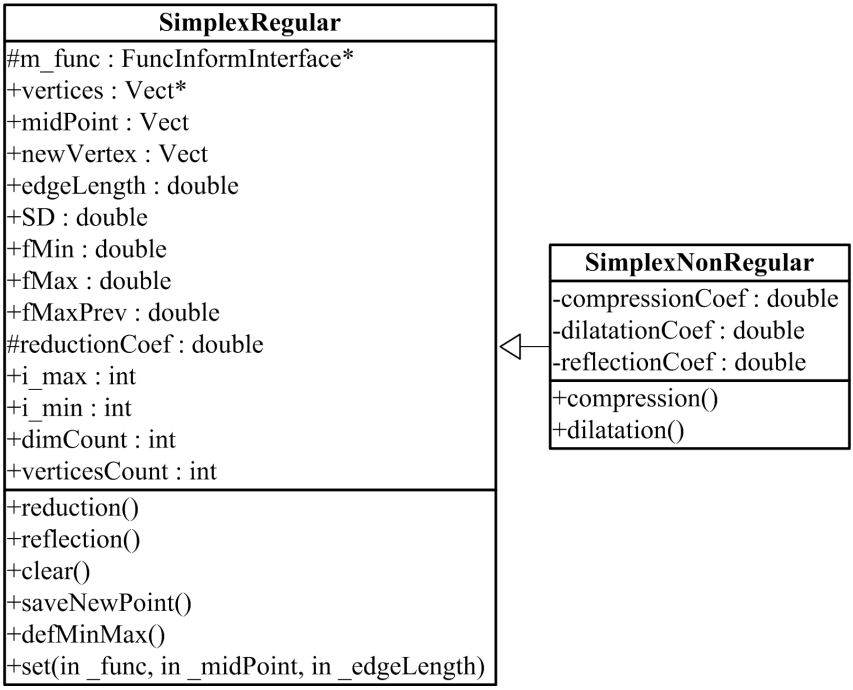


Рис. 8.1. Базовая структура `SimplexRegular`, реализующая работу с регулярным симплексом, и производная структура `SimplexNonRegular`, реализующая работу с нерегулярным симплексом

ЛИТЕРАТУРА

Основная литература

1. *Аттетков А. В., Галкин С. В., Зарубин В. С.* Методы оптимизации. М.: Изд-во МГТУ им. Н. Э. Баумана, 2003. 440 с.

Дополнительная литература

2. *Аоки М.* Введение в методы оптимизации. Основы и приложения нелинейного программирования. М.: Наука, 1977. 344 с.

3. *Аттетков А. В., Зарубин В. С., Канатников А. Н.* Введение в методы оптимизации. М.: Финансы и статистика; ИНФРА-М, 2008. 272 с.

4. *Банди Б.* Методы оптимизации. Вводный курс. М.: Радио и связь, 1998. 128 с.

5. *Васильев Ф. П.* Численные методы решения экстремальных задач. М.: Наука, 1988. 552 с.

6. *Дамбраускас А. П.* Симплексный поиск. М.: Энергия, 1979. 176 с.

7. *Поляк Б. Т.* Введение в оптимизацию. М.: Наука, 1983. 384 с.

8. *Химмельблау Д.* Прикладное нелинейное программирование. М.: Мир, 1975. 534 с.

9. *Мину М.* Математическое программирование. Теория и алгоритмы. М.: Наука, 1990. 487 с.

10. *Ганшин Г. С.* Методы оптимизации и решение уравнений. М.: Наука, 1987. 128 с.

11. *Зангвилл У. И.* Нелинейное программирование. Единый подход. М.: Сов. радио, 1973. 312 с.

12. *Измаилов А. Ф., Солодов М. В.* Численные методы оптимизации. М.: Физматлит, 2003. 304 с.

13. *Карманов В. Г.* Математическое программирование. М.: Физматлит, 2001. 264 с.

14. *Моисеев Н. Н., Иванилов Ю. П., Столярова Е. М.* Методы оптимизации. М.: Наука, 1978. 351 с.

15. *Полак Э.* Численные методы оптимизации. Единый подход. М.: Мир, 1974. 376 с.

16. *Канатников А. Н., Крищенко А. П.* Линейная алгебра. М.: Изд-во МГТУ им. Н. Э. Баумана, 2002. 336 с.

17. *Канатников А. Н., Крищенко А. П., Четвериков В. Н.* Дифференциальное исчисление функций многих переменных. М.: Изд-во МГТУ им. Н. Э. Баумана, 2000. 456 с.

18. *Карпенко А. П.* Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой. М.: Изд-во МГТУ им. Н. Э. Баумана, 2014. 500 с.

19. *Комарцова Л. Г., Максимов А. В.* Нейрокомпьютеры. М.: Изд-во МГТУ им. Н. Э. Баумана, 2004. 400 с.

20. *Хайкин С.* Нейронные сети: полный курс. М.: Издательский дом «Вильямс», 2006. 1104 с.

21. *Хофер Э., Лундерштедт Р.* Численные методы оптимизации. М.: Машиностроение, 1981. 192 с.

22. *Пахомов Б. И.* C/C++ и MS Visual C++ 2008. СПб.: БХВ-Петербург, 2009. 624 с.

23. *Шилдт Г.* Полный справочник по C++. М.: Издательский дом «Вильямс», 2006. 800 с.

24. *Элджер Дж.* C++: Библиотека программиста. СПб.: Питер, 1999. 320 с.

25. *Гамма Э., Хелм Р., Джонсон Р., Влиссидес Д.* Приемы объектно-ориентированного проектирования. Паттерны проектирования. СПб.: Питер, 2010. 366 с.

26. *Буч Г., Рамбо Д., Джекобсон А.* Язык UML. Руководство пользователя. СПб.: Питер, 2004. 430 с.

27. *Васильев А. Н.* Mathematica. Практический курс с примерами решения прикладных задач. СПб.: КОРОНА-ВЕК, 2008. 448 с.

28. *Балдин Е. М.* Компьютерная типография LaTex. СПб.: БХВ-Петербург, 2008 г. 304 с.