МЕХАНИКА ДЕФОРМИРУЕМОГО ТВЕРДОГО ТЕЛА

Научная статья
УДК 624.04
https://doi.org/10.24866/2227-6858/2025-1/43-55

Алгоритмы оптимизации в инженерном проектировании: сравнительный анализ методов модуля SciPy

Татьяна Львовна Дмитриева✉, **Александр Евгеньевич Ботхоев**

Иркутский национальный исследовательский технический университет,
Иркутск, Российская Федерация
✉ dmitrievat@istu.edu

Аннотация. Целью данной работы является анализ применения различных алгоритмов оптимизации для решения инженерных задач. Постановка проблемы оптимального проектирования представлена в форме нелинейного программирования как задача на условный экстремум. Рассмотрены классические методы оптимизации, входящие в библиотеку SciPy языка программирования Python. К исследованию приняты 3 метода безусловной минимизации: *Симплексный метод Нелдера – Мида*, *Метод Пауэлла*, *Метод Бройдена – Флетчера – Голдфарб – Шанно,* которые встраивались в авторский алгоритм оптимизации, реализующий переход исходной задачи к задаче на безусловный экстремум на основе некоторой модификации функции Лагранжа. Рассмотрен также отдельный модуль библиотеки SciPy решения задачи условной минимизации с использованием метода *SLSQP* (*Sequential Least Squares Quadratic Programming*). Для оценки эффективности исследуемых методов решена известная тестовая задача оптимизации консольной пластины. Выполнен анализ полученных решений по скорости сходимости и точности результатов. Выявлено, что все 3 метода на безусловный экстремум показали похожие результаты, где близкое к оптимальному решение было получено уже на третьей итерации поискового процесса с дальнейшей незначительной корректировкой на последующих итерациях, что является успешным результатом встраивания этих методов в авторский алгоритм оптимизации. Метод *SLSQP* показал менее устойчивую сходимость, т.к. близкое к оптимальному решение было получено только на 9-ой итерации. Таким образом, алгоритм на основе модифицированной функции Лагранжа, разработанный авторами, в сочетании с модулями безусловной минимизации библиотеки SciPy может быть рекомендован в дальнейшем для задач оптимизации большой размерности.

Ключевые слова: инженерная оптимизация, Python, методы условной минимизации, нелинейное программирование, функция Лагранжа
Для цитирования: Дмитриева Т.Л., Ботхоев А.Е. Алгоритмы оптимизации в инженерном проектировании: сравнительный анализ методов модуля SciPy // Вестник Инженерной школы Дальневосточного федерального университета. 2025. № 1(62). С. 43–55.

MECHANICS OF DEFORMABLE SOLIDS

Original article

Optimization algorithms in engineering design: a comparative analysis of SciPy module methods

Tatyana L. Dmitrieva✉, **Alexander E. Botkhoev**

Irkutsk National Research Technical University, Irkutsk, Russian Federation
✉ dmitrievat@istu.edu

Abstract. The aim of this study is to analyze the application of various optimization algorithms for solving engineering problems. The formulation of the optimal design problem is presented in the form of nonlinear programming as a constrained extremum problem. Classical optimization methods inclu-

ded in the SciPy library of the Python programming language are considered. The study focuses on three unconstrained minimization methods: *the Nelder – Mead simplex method, Powell's method, and the Broyden – Fletcher – Goldfarb – Shanno (BFGS) method*, which were integrated into the authors' optimization algorithm. This algorithm transforms the original problem into an unconstrained extremum problem based on a modified Lagrangian function. Additionally, a separate module of the SciPy library for constrained minimization using *the Sequential Least Squares Quadratic Programming (SLSQP)* method is examined. To evaluate the efficiency of the studied methods, a well-known benchmark optimization problem of a cantilever plate is solved. The obtained solutions are analyzed in terms of convergence rate and accuracy. It is found that all three unconstrained minimization methods produced similar results, with a near-optimal solution obtained as early as the third iteration of the search process, followed by minor adjustments in subsequent iterations. This demonstrates the successful integration of these methods into the authors' optimization algorithm. The *SLSQP* method exhibited less stable convergence, as a near-optimal solution was obtained only by the ninth iteration. Thus, the algorithm based on the modified Lagrangian function, developed by the authors, in combination with the unconstrained minimization modules of the SciPy library, can be recommended for further use in large-scale optimization problems.

***Keywords*:** engineering optimization, Python, conditional minimization methods, nonlinear programming, Lagrange function

For citation: Dmitrieva T.L., Botkhoev A.E. Optimization algorithms in engineering design: a comparative analysis of SciPy module methods. *FEFU: School of Engineering Bulletin*, 2025, no. 1(62), pp. 43–55. (In Russ.).

Введение

Поиск оптимальных решений в проектировании инженерных объектов охватывает широкий спектр направлений, связанных с конструктивными, технологическими, эксплуатационными вопросами [1–4], сюда также можно отнести проблемы надёжности, долговечности и множество других аспектов. Между тем, несмотря на определённый опыт, накопленный в области разработки высокоэффективных алгоритмов и программ оптимизации инженерных объектов, следует отметить, что в российской практике специализированное программное обеспечение, реализующее процедуры оптимизации, не находит такого массового применения, как программные комплексы инженерного анализа. В большей мере это связано с рядом факторов, затрудняющих разработку и практическое использование подобных программ. Популярные в отечественном проектировании зарубежные программные комплексы, такие как *ANSYS* и *NASTRAN*, включающие наиболее развитые модули оптимизации, не всегда доступны проектировщику и требуют от пользователя выбора определённого метода, используемого на протяжении вычислительного процесса, что не всегда обеспечивает результативность решения [5]. Для возможной верификации, а в некоторых случаях и для достижения результатов, целесообразней выполнять расчёты с использованием различных поисковых методов оптимизации, гибко настраивая при этом вычислительный процесс.

Для решения узкоспециализированных задач возможна разработка собственных программных модулей на основе имеющихся в свободном доступе средств программирования. В этом случае разработчик может полностью автоматизировать собственные алгоритмы [6] либо использовать готовые библиотеки, например, такие как библиотека *SciPy* языка программирования *Python* [7]. Модули *Python* могут быть встроены в различные вычислительные комплексы инженерного анализа, обеспечивая удобный интерфейс для проведения инженерных расчётов и оптимизации, не нарушая при этом стилистического оформления программы [8]. В настоящей статье выполнена автоматизация нескольких алгоритмов оптимизации с использованием модуля *scipy.optimize* библиотеки *SciPy* [9].

Материалы и методы

Рассмотрим реализацию алгоритмов оптимизационного поиска применительно к тестовой задаче оптимизации консольного *стержня* переменной толщины (рис. 1*а*) под действием сосредоточенного момента *M* на свободном конце.

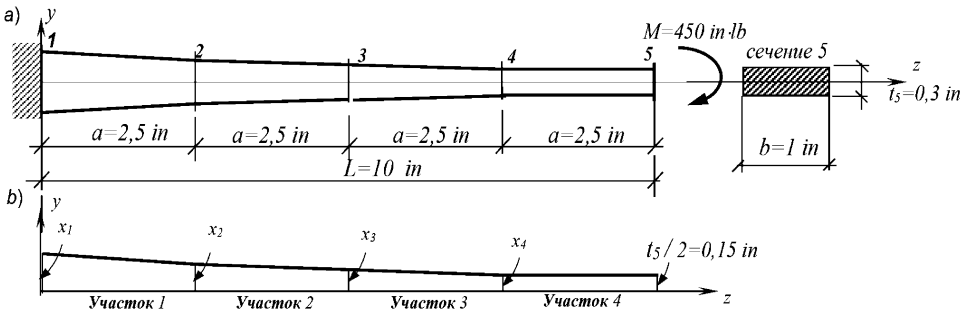

Рис. 1. Оптимизируемый консольный стержень

Fig. 1. Optimizable console beam

Требуется подобрать значения толщин в сечениях 1–4 из условия минимума объёма консоли. Приняты ограничения:

– на максимальные напряжения $\sigma_{max} \le 30000$ *psi;*

– на максимальное перемещение $\Delta_{max} \le 0{,}5$ *in.*

Варьировались значения полутолщин X_i в узловых сечениях 1–4, показанных на рис. 1*b*. Толщина консоли в месте приложения нагрузки фиксирована (t_5=0,3 *in*). Шаг сечений $a=L/4$. Модуль упругости $E = 10^7$ *psi*.

Целевая функция *f(x)* представляет собой объём консоли:

$$f(x) = 2.5 \cdot (x_1 + \frac{t_5}{2} + 2 \cdot \sum_{i=2}^{4} x_i). \quad (1)$$

В силу того что нижний предел для всех x_i принят 0,15 *in*, ограничения на напряжения $\sigma_{max} \le 30000$ *psi* выполнялись автоматически, поэтому к расчёту принято только одно ограничение на перемещение в сечении 5, что существенно упростило алгоритм решения задачи:

$$g_1(x) = \frac{\Delta_5(x)}{0{,}5} - 1 \le 0. \quad (2)$$

Здесь значение перемещения Δ_5 вычислялось при помощи интеграла Мора:

$$\Delta_5(x) = \frac{M}{E} \cdot \sum_{i=1}^{4} \int_{l_i}^{l_{i+1}} \frac{\bar{M}(z) \cdot 12}{(2t_i(x,z))^3} dz, \quad (3)$$

где $\bar{M}(z)$ – функция изгибающего момента на единичном состоянии

$$\bar{M}(z) = 10 - z, \quad (4)$$

а пределы интегрирования по участкам приняты с учётом того, что l_i – привязка i-го сечения к началу координат (l_5=4a). Таким образом,

$$l_i = (i-1) \cdot a, \;\; i = 1..5.$$

Соответственно, толщина стержня в произвольном сечении для i-го участка была рассчитана через параметры варьирования x_i.

$$t_i(x,z) = x_i - \frac{x_i - x_{i+1}}{a} \cdot (z - l_i), \; i = 1,\ldots,4 \quad (5)$$

Математическая модель задачи в форме нелинейного программирования представляет задачу минимизации объёма консоли на заданном ограничении:

$$\text{найти} \quad min\, f(x) = 2.5 \cdot (x_1 + \frac{t_5}{2} + 2 \cdot \sum_{i=2}^{4} x_i) \quad (6)$$

$$\text{при} \quad g_1(x) = \frac{M}{0{,}5 \cdot E} \cdot \sum_{i=1}^{4} \int_{l_i}^{l_{i+1}} \frac{(10-z) \cdot 12}{(2t_i(x,z))^3} \, dz - 1 \le 0.$$

Программная реализация алгоритма была выполнена на языке *Python* с подключением библиотеки *SciPy* и входящим в неё модулем *scipy.optimize*. Исходный код этой библиотеки открыт, она распространяется бесплатно по лицензии BSD. Использована среда разработки *Spyder*, которая позволяет выполнять расчёт интерактивно с анализом и визуализацией данных.

Для исследования было выбрано несколько методов:

- метод условной минимизации *SLSQP;*
- методы безусловной минимизации *Нелдера – Мида, Пауэлла и BFGS.*

Приведём основные шаги настройки среды и ввода исходных данных.

1. Импортируем в среду разработки spyder модули: *numpy* – для работы с массивами*; matplotlib* – для построения графиков функций, *scipy.optimize* – для оптимизации*; scipy.integrate* – для нахождения интегралов (рис. 2).

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import minimize
from scipy.integrate import quad
```

Рис. 2. Импорт библиотек

Fig. 2. Importing libraries

2. Задаём исходные данные (рис. 3).

```
# Исходные данные
E = 10000000
L = 10
M = 450
t5 = 0.3
a = L / 4
# Начальные значения
x = np.array([0.4, 0.3, 0.2, 0.18])
```

Fig. 3. Input of initial data

Рис. 3. Ввод исходных данных

```
def f(x):
    return 2.5 * (x[0] + t5 / 2 + 2 * (x[1] + x[2] + x[3]))
def t1(x, z):
    return x[0] - (x[0] - x[1]) / a * (z - l[0])
def t2(x, z):
    return x[1] - (x[1] - x[2]) / a * (z - l[1])
def t3(x, z):
    return x[2] - (x[2] - x[3]) / a * (z - l[2])
def t4(x, z):
    return x[3] - (x[3] - t5) / a * (z - l[3])
def M1(z):
    return (10 - z)
# Интегрирование по интервалам
def integrand1(z, x):
    return M1(z) * 12 / (2 * t1(x, z))**3
def integrand2(z, x):
    return M1(z) * 12 / (2 * t2(x, z))**3
def integrand3(z, x):
    return M1(z) * 12 / (2 * t3(x, z))**3
def integrand4(z, x):
    return M1(z) * 12 / (2 * t4(x, z))**3 def delta(x):
    integral1, _ = quad(integrand1, l[0], l[1], args=(x,))
    integral2, _ = quad(integrand2, l[1], l[2], args=(x,))
    integral3, _ = quad(integrand3, l[2], l[3], args=(x,))
    integral4, _ = quad(integrand4, l[3], l[4], args=(x,))
    return M / E * (integral1 + integral2 + integral3 + integral4)
def g(x):
    return (delta(x) / 0.5 - 1)
```

Рис. 4. Определение целевой функции f(x) и функции ограничения g(x)

Fig. 4. Definition of the objective function f(x) and the constraint function g(x)

А. Решение методом условной минимизации. Приведём последовательность операций алгоритма решения задачи условной оптимизации (рис. 5).

Рис. 5. Блок-схема алгоритма решения условной задачи

Fig. 5. Flowchart of the algorithm for solving the constrained problem

Выход из программы может произойти по одной из следующих причин:

1. *Сходимость по параметрам:* относительная разница между значениями параметров варьирования на соседних итерациях меньше заданного порога (*по умолчанию* 10^{-6}).

2. *Сходимость по функции цели:* разница между значениями целевой функции на соседних итерациях меньше заданного порога (*по умолчанию не задано*). То есть дальнейшее улучшение значения функции маловероятно.

3. *Сходимость по градиенту:* если норма градиента целевой функции становится меньше заданного порога (*по умолчанию* 10^{-6}). Означает, что решение приблизилось к стационарной точке.

4. *Исчерпание лимита итераций:* если количество итераций превысит максимальный порог (*по умолчанию 1000*).

5. *Прочие причины*: численные ошибки или вычислительные проблемы.

Рассмотрим обращение к методу *SLSQP (Sequential Least Squares Quadratic Programming).* Здесь на каждой итерации применяется последовательная квадратичная аппроксимация целевой и ограничительных функций на основе метода наименьших квадратов [10]. Используем команду *minimize*, где параметр *bounds* учитывает границы переменных *x*; *constraints* включает функцию ограничения *g(x); callback* помогает отслеживать алгоритм, а *options* – дополнительные параметры (рис. 6).

```
def constraint(x):
    return -g(x)
constraints = {'type': 'ineq', 'fun': constraint}
bounds = [(0.15, 0.5), (0.15, 0.5), (0.15, 0.5), (0.15, 0.5)]
res = minimize(f, x, bounds=bounds, constraints=constraints,
method='SLSQP', callback=callback, options={'disp': True})
x = res.x
print(x)
```

Рис. 6. Использование метода SLSQP для минимизации функции f(x)

Fig. 6. Using the SLSQP method for minimizing the function f(x)

Б. Включение методов безусловной минимизации. Решение задачи (6) может быть сведено к задаче на безусловный экстремум с использованием функции Лагранжа F_L, которая для одного ограничения имеет вид:

$$F_L(x) = f(x) + Y \cdot g_1(x), \quad (7)$$

где Y – множитель Лагранжа. Однако в работе [1] отмечено, что функция (7) применима для задач выпуклого программирования. Для преодоления этого недостатка исследована модификация функции Лагранжа F_P, которая существенно расширяет область исследования:

$$F_P(x) = F_L(x) + 0.5 \cdot k \cdot g_1(x)^2. \quad (8)$$

Таким образом, решение представляет собой итерационный процесс, где на каждой итерации t минимизируется функция $F_p(x)$, а затем переопределяется двойственная переменная Y и штрафной коэффициент k.

$$Y^{t+1} = \max(0, Y^t + k \cdot g(x)).$$
$$k = \frac{Y}{\Delta Z_{max}} + k_{min}. \quad (9)$$

Здесь ΔZ_1 – величина сдвига ограничения в допустимую область; δ – булевая переменная, определяемая из условия: если $g_1 + \Delta Z_1 > 0$, то $\delta = 1$, иначе $\delta = 0$.

Рассмотрим используемые при этом методы безусловной минимизации.

I. *Симплексный метод Нелдера – Мида* относится к методам прямого поиска. Устойчив для решения задач с негладкими функциями, где вычисление производных затрудни тельно, а иногда и невозможно [10], но менее эффективен для задач большой размерности [11], так как требует большого числа обращений к функциям ограничений.

II. *Метод Пауэлла* (также прямой метод [12, 13]) минимизирует функцию на основе поиска по двум сопряжённым направлениям. Хорошо работает для нелинейных функций [14], однако не всегда эффективен для задач со сложным характером функций ограничений.

III. *Метод Бройдена – Флетчера – Голдфарба – Шанно (BFGS)* – квазиньютоновский метод [15], основанный на аппроксимации обратной матрицы Гессе для направления поиска. Существуют модификации метода: с ограниченным использованием памяти (*L-BFGS*) [16], с использованием памяти в многомерном кубе (*L-BFGS-B*) [17, 18] для решения нелинейных большеразмерных задач. Эффективен для задач с хорошо определяемыми производными.

Покажем блок-схему алгоритма, использующего эти методы [1] (рис. 7).

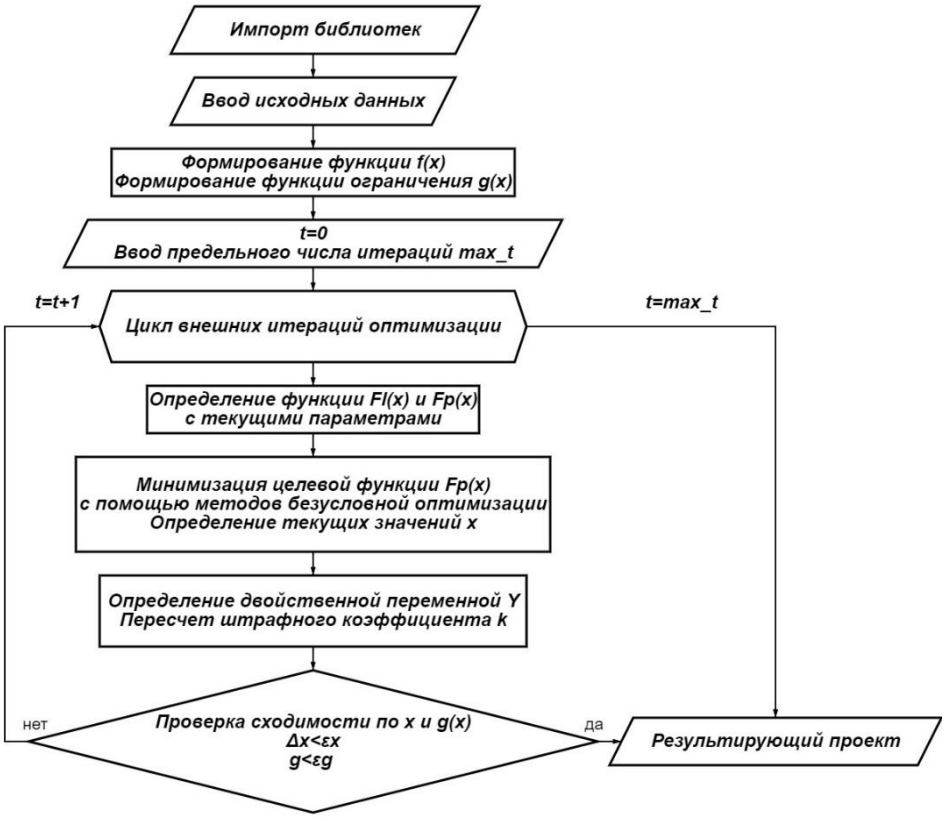

Рис. 7. Блок-схема алгоритма решения условной задачи с использованием методов безусловной минимизации

Fig. 7. Flowchart of the algorithm for solving the constrained problem using unconstrained minimization methods

Приведём пошаговое выполнение автоматизации процесса оптимизации.

1. Вводим значения исходных данных.

```
y = 0
k=4
delta_Zmax = 0.3
kmin = 4
max_t = 12
εx=1e-6
εg=1e-6
```

Рис. 8. Ввод данных

Fig. 8. Input of data

2. Формируем функцию Лагранжа F_L и модифицированную функцию F_P, которая затем минимизируется на заданном интервале варьирования (рис. 9).

```
for iter in range(max_t):
    print("iter =", t)
    # Пересчет параметров
    delta_Z = y / k
    deltan = 1 if g(x) + delta_Z > 0 else 0
    def FL(x):
        return f(x) + y * deltan * g(x)
    def FP(x):
        return FL(x) + 0.5 * g(x)**2 * deltan * k
# Минимизируем функцию Fp
res = minimize(FP, x, method='Powell', bounds=bounds)
x = res.x
print(x)
# Пересчет параметра y и k
    y = max(0, y + k * g(x))
    k = y / delta_Zmax + kmin
# Проверка условий выхода
    if x_change < εx:
        print("Exit condition met. X<εx")
        break
    # Проверка условий выхода g(x)
    if abs(g_val) < εg:
        print("Exit condition met. g<εg")
        break
```

Рис. 9. Цикл минимизации функции FP

Fig. 9. Cycle of minimizing the function FP

Для минимизации функции *Fp* используем команду *minimize*, в параметре *method* указываем используемый метод. Параметр *bounds* учитывает границы *x*. Окончанием цикла является достижение предельного количества итераций (*max_t*) или удовлетворения условия выхода (рис. 9).

Обсуждение результатов

Использование модуля условной минимизации *SLSQP*. Сходимость алгоритма была получена на 12-й итерации (рис. 10).

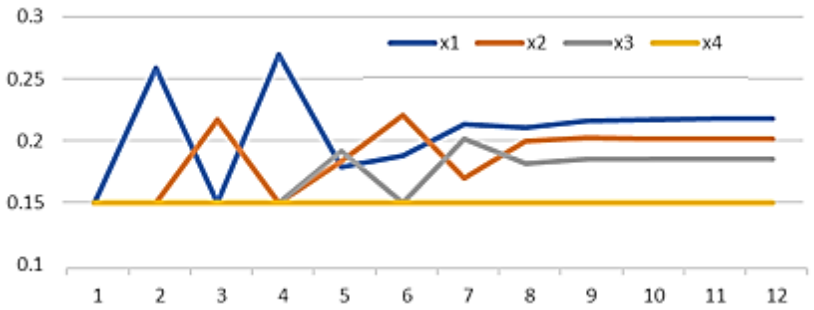

Рис. 10. График значений x (метод SLSQP)

Fig. 10. Plot of x values (SLSQP method)

Здесь заданная точность вычислений (10^{-8}) проверялась соотношением параметров ***x*** на соседних итерациях. Изменения целевой и ограничительной функций показаны на рис. 11.

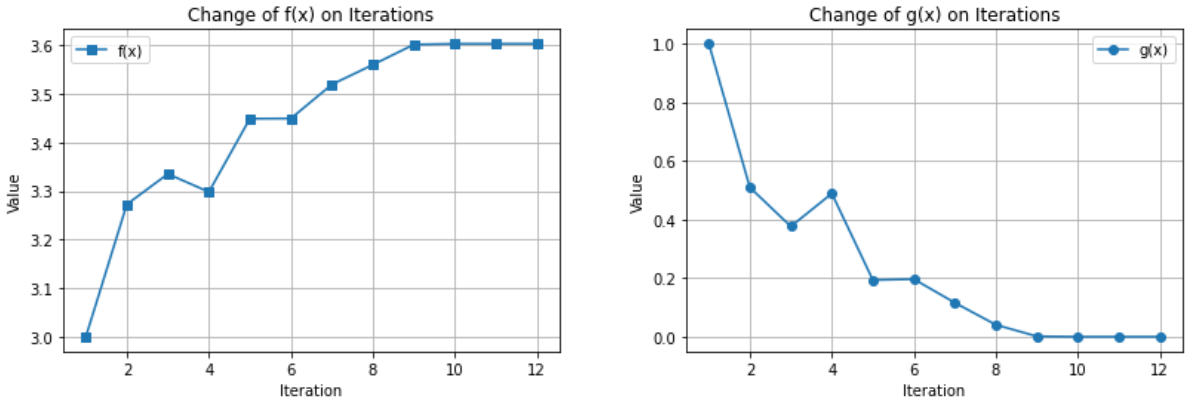

Рис. 11. Графики функций f(x) и g(x) (метод SLSQP)

Fig. 11. Plots of functions f(x) and g(x) (SLSQP method)

Использование методов безусловной минимизации на основе метода модифицированной функции Лагранжа *Fp*.

Отметим 2 преимущества такого подхода:

1. Возможна настройка алгоритма путём назначения параметров, влияющих на сходимость, таких как *kmin*, *ΔZmax* (см. выражения (8–9)).

2. Есть возможность следить за ходом решения, а также осуществлять выход из цикла (рис. 9) на основе проверки сходимости по значениям параметров варьирования на соседних итерациях, а также выполнения заданных ограничительных условий с требуемой точностью.

$$\frac{|x_{i+1} - x_i|}{x_i} < \varepsilon_x, \quad |g(x)| < \varepsilon_g. \tag{10}$$

Симплексный алгоритм Нелдера – Мида. Метод показал достаточно точный результат на 9-й итерации (рис. 12).

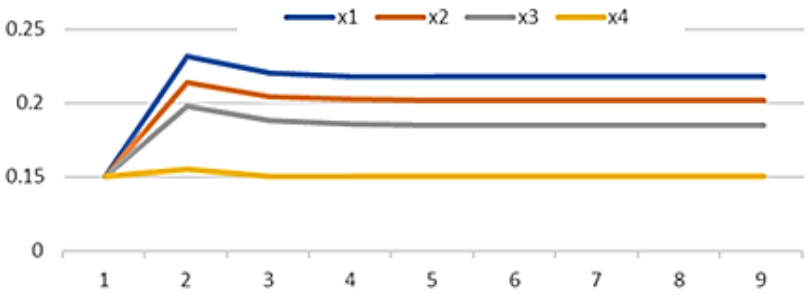

Рис. 12. График значений x (алгоритм Нелдера – Мида)

Fig. 12. Plot of x values (Nelder – Mead algorithm)

Отметим, что здесь присутству.т также внутренние итерации. По ходу приближения к оптимальному результату количество обращений к функции *Fp* на внешних итерациях *t* уменьшается с 209 до 70 на 3 и 9 итерациях соответственно.

Метод Пауэлла показал стабильную работу. Однако ввиду линейного поиска скорость достаточно низкая.

Алгоритм Бройдена – Флетчера – Голдфарба – Шанно (BFGS) показал быструю сходимость и высокую точность в поиске оптимального решения.

Сравнительный анализ используемых методов. Методы безусловной минимизации показали близкие результаты. Уже на 4-й итерации получено решение, близкое к оптимальному, которое в дальнейшем доводится до требуемой точности. На рис. 13, 14 показаны изменения целевой и ограничительных функций на итерациях для этих методов.

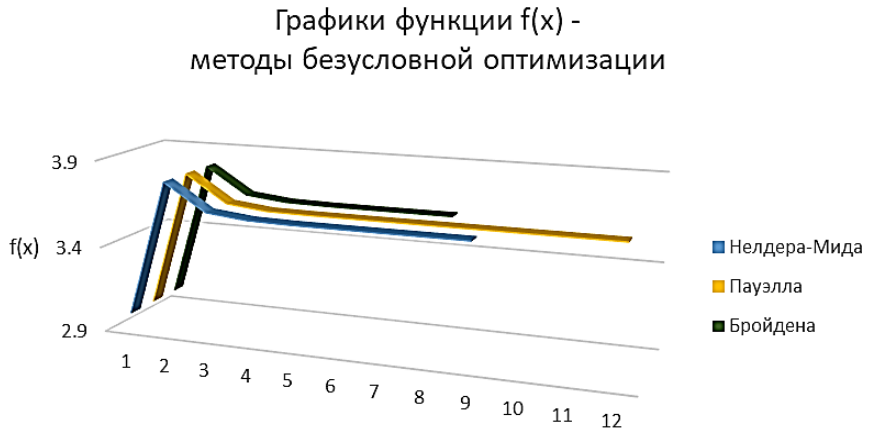

Рис. 13. Графики функций f(x) (методы безусловной оптимизации)

Fig. 13. Plots of functions f(x) (unconstrained optimization methods)

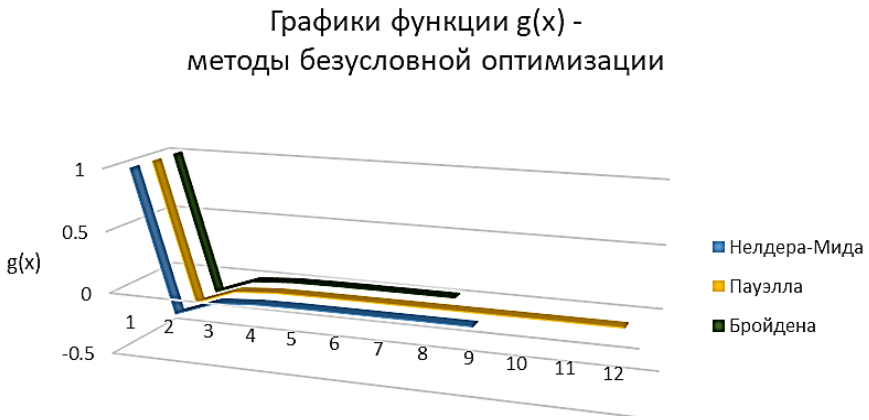

Рис. 14. Графики функций g(x) (методы безусловной оптимизации)

Fig. 14. Plots of functions g(x) (unconstrained optimization methods)

Выделим методы, показавшие наилучшие результаты, по критериям:

- по числу внешних итераций: метод *L-BFGS-B;*
- по числу обращений к целевой функции: метод *Powell;*
- по времени счёта: метод *L-BFGS-B;*
- по точности вычисления: алгоритм *Нелдера – Мида*, метод *L-BFGS-B.*

Сравнительные числовые значения приведены в таблице 1.

Более высокая точность в результатах *SLSQP* метода обусловлена заложенным в нём критерием сходимости в невязках ограничений (10^{-8}), что потребовало большего числа итераций. Но процесс сходимости здесь показывает больший разброс значений. Результаты, близкие к оптимальным, получены только на 9-й итерации (рис. 12).

Таблица 1 / Table 1

Сравнение методов оптимизации
Comparison of optimization methods

Наименование	Метод оптимизации			
	Метод SLSQP	Алгоритм Нелдера – Мида	Метод Пауэлла	Метод L-BFGS-B
Число итераций	12	9	12	8
Значение функции f(x)	3,603311763	3,603270055	3,603391127	3,603328235
Значение функции g(x)	$1{,}98\cdot10^{-9}$	$4{,}77\cdot10^{-5}$	$-3{,}561\cdot10^{-5}$	$-5{,}291\cdot10^{-5}$
X_1, in	0,217456172	0,217603166	0,217658348	0,217641701
X_2, in	0,201915294	0,201882202	0,201827761	0,20189474
X_3, in	0,185018973	0,184970226	0,184960716	0,184950056
X_4, in	0,15	0,15	0,150060575	0,15
Время, с	0,023076296	0,56085562	0,723393678	0,24882292

Использование метода модифицированных функций Лагранжа показало более устойчивую монотонную сходимость. При этом параметры kmin и ΔZmax влияют на скорость сходимости, т.к. меняют кривизну функции Fp. Так, если принять kmin=1 и ΔZmax=0,1, сходимость на основе метода L-BFGS-B достигается за 4 итерации (рис. 15).

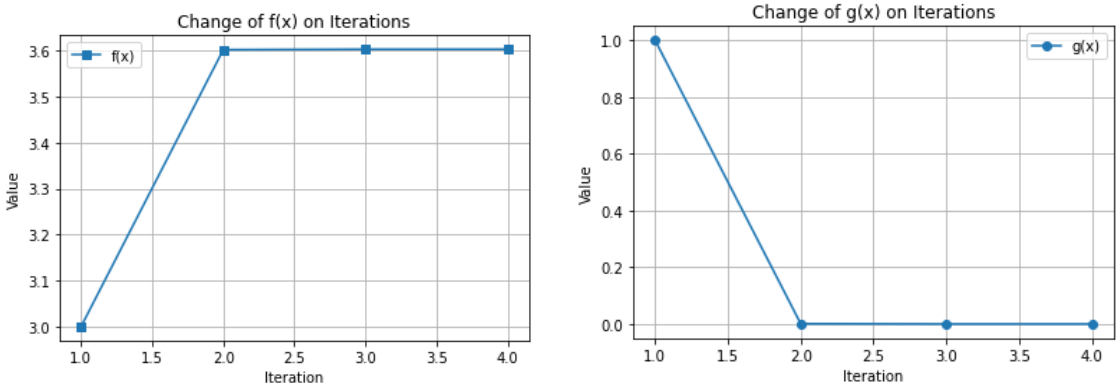

Рис. 15. Графики функций f(x) и g(x) (метод L-BFGS-B)
Fig. 15. Plots of functions f(x) and g(x) (*L-BFGS-B method)*

Данная задача рассматривалась ранее в верификационном отчёте ПК *ANSYS Mechanical* [19], где конструкция рассчитывалась как пластина, работающая в условиях плоского напряжённого состояния, где, соответственно, имела место погрешность в решении задачи КЭ анализа. В таблице 2 приведено сравнение с результатами, полученными в ПК *ANSYS*.

Таблица 2 / Table 2

Сравнение результатов расчёта с решением в ПК Ansys
Comparison of calculation results with the solution in the Ansys PC

		Объём, in^3	ε (%)	Максимальное перемещение, in	ε (%)	Число итераций
Аналитическое решение		3,600		0,500		
ANSYS Метод аппроксимации подзадачи		3,616	0,444	0,499	0,15	12
ANSYS		3,609	0,25	0,501	0,13	17
SciPy (методы)	*SLSQP*	3,6033117	0,092	0,500000001	$2\cdot10^{-7}$	12
	Нелдера – Мида	3,60327	0,093	0,500023868	$4{,}7\cdot10^{-3}$	9
	Пауэлла	3,6033911	0,094	0,499982194	$3{,}5\cdot10^{-3}$	12
	L-BFGS-B	3,6033282	0,093	0,499997356	$5{,}3\cdot10^{-4}$	8

Заключение

Выбор конкретного метода оптимизации зависит от вида задачи, требований к точности и скорости сходимости [20]. Методы безусловной оптимизации с применением модифицированной функции Лагранжа показали приблизительно похожие по сходимости результаты – на 4-й итерации было получено близкое к оптимальному решение, которое корректировалось на последующих итерациях. Главным преимуществом данного подхода является возможность регулировать процесс сходимости.

Создание подобных программных разработок открывает новые перспективы для инженеров и исследователей в области оптимизации конструкций различного назначения. В силу простоты задачи трудно говорить о преимуществах того или иного метода. Дальнейшее исследование предполагает решение этого примера с подключением модуля scipy.optimize на основе более сложной расчётной схемы консоли – как пластины в условиях плоского напряжённого состояния, что потребует подключения к расчёту модуля КЭ анализа.

ВКЛАД АВТОРОВ | CONTRIBUTION OF THE AUTHORS

Авторы сделали эквивалентный вклад в подготовку публикации. Т.Л. Дмитриева – формулировка исследовательской задачи, корректность постановки вычислительных экспериментов и интерпретация полученных результатов; А.Е. Ботхоев – реализация алгоритма оптимизации, проведение численных экспериментов и подготовка анализа полученных результатов по критериям точности и скорости сходимости.

The authors contributed equally to this article. T.L. Dmitrieva – formulation of the research problem, correctness of the computational experiments and interpretation of the obtained results; A.E. Botkhoev – implementation of the optimization algorithm, conducting numerical experiments and preparation of the analysis of the obtained results based on accuracy and convergence rate criteria.

КОНФЛИКТ ИНТЕРЕСОВ | DISCLOSURE

Авторы заявляют об отсутствии конфликта интересов.
The authors declare no conflict of interest.

СПИСОК ИСТОЧНИКОВ

1. Kablukov A.V., Dmitrieva T.L. State of the problem of numerical modeling of optimal design of load bearing structures of linear pipeline systems. IOP Conf. Series: Materials Science and Engineering. 880 (2020) 012078. DOI: https://doi.org/10.1088/1757-899X/880/1/012078
2. Башин К.А., Торсунов Р.А., Семенов С.В. Методы топологической оптимизации конструкций, применяющиеся в аэрокосмической отрасли // Вестник ПНИПУ. Аэрокосмическая техника. 2017. № 51. С. 51–61. DOI: https://doi.org/10.15593/2224-9982/2017.51.05
3. Ma Y., Gao X., Liu C., Li J. Improved SQP and SLSQP algorithms for feasible path-based process optimisation. 2024. Т. 188. С. 108751.
4. Gong M. et al. An experimental study on local and global optima of linear antenna array synthesis by using the sequential least squares programming // Applied Soft Computing. 2023. Vol. 148. P. 110859.
5. Дмитриева Т.Л., Чан Л.Т.М. Оптимальное проектирование стальных конструкций с использованием ANSYS // International Journal for Computational Civil and Structural Engineering. 2014. Т. 10. С. 79–84. DOI: https://doi.org/10.22337/2587-9618
6. Дмитриева Т.Л. Программный комплекс «OPTIDEST» и его использование в задачах расчёта и оптимизации стальных конструкций // Вестник МГСУ. 2011. Т. 1, № 1. С. 100–105.
7. Gommers R. et al. scipy/scipy: SciPy 1.9.0 // Zenodo. 2022. P. 15.
8. Hill C. Optimization with scipy.optimize // Python for Chemists. Cambridge University Press, 2023. P. 247–259.
9. Мельникова В.А., Куприянова Ю.В. Применение специальных библиотек Python для изучения задач линейного программирования // Совершенствование качества образования: сборник статей XX(XXXVI) Всероссийской научно-методической конференции. Братск: Изд-во БрГУ, 2023. С. 146.
10. Певнева А.Г., Калинкина М.Е. Методы оптимизации. СПб: Университет ИТМО, 2020. С. 64.

11. Ревякин М.А. Метод Нелдера – Мида для решения задач нелинейного программирования // Приоритетные направления инновационной деятельности в промышленности. 2021. С. 78–80.
12. Лажаунинкас Ю.В., Кочегарова О.С. Одномерная оптимизация методом Пауэлла // Экономико-математические методы анализа деятельности предприятий АПК. 2022. С. 253–258.
13. Powell M.J.D. An efficient method for finding the minimum of a function of several variables without calculating derivatives // Computer Journal. 1964. vol. 7, № 2. P. 155–162.
14. Кляус К.М. Численные методы нелинейной оптимизации в задачах математического моделирования. Общая характеристика. 2021. С. 112.
15. Dwail H.H., Shiker M.A.K. Using trust region method with BFGS technique for solving nonlinear systems of equations // Journal of Physics: Conference Series. IOP Publishing, 2021. Vol. 1818, № 1. P. 012022.
16. Li L., Hu J. Fast-converging and low-complexity linear massive MIMO detection with L-BFGS method // IEEE Transactions on Vehicular Technology. 2022. Vol. 71, № 10. P. 10656–10665.
17. Pratama D.A. et al. Solving partial differential equations with hybridized physic-informed neural network and optimization approach: Incorporating genetic algorithms and L-BFGS for improved accuracy // Alexandria Engineering Journal. 2023. Vol. 77. P. 205–226.
18. Nocedal J., Wright S.J. Numerical optimization. Second edition. 2006. P. 526–573.
19. Ali K.S.A. et al. Analysis of composite leaf spring using ANSYS software // Materials Today: Proceedings. 2021. P. 2346–2351.
20. Богданова П.А., Сахаров Д.М., Васильева Т.В. Обзор методов многокритериальной оптимизации в задачах принятия решений // Инновационные аспекты развития науки и техники. 2021. № 6. С. 153–157.

REFERENCES

1. Kablukov A.V., Dmitrieva T.L. State of the problem of numerical modeling of optimal design of load bearing structures of linear pipeline systems. *IOP Conf. Series: Materials Science and Engineering*, 880 (2020) 012078. DOI: https://doi.org/10.1088/1757-899X/880/1/012078
2. Bashin K.A., Torsunov R.A., Semenov S.V. Topology optimization methods in aerospace industry. *Bulletin of PNRPU. Aerospace Engineering*, 2017, no. 4(51), pp. 51–61. (In Russ.). DOI: https://doi.org/10.15593/2224-9982/2017.51.05
3. Ma Y., Gao X., Liu C., Li J. Improved SQP and SLSQP algorithms for feasible path-based process optimization. 2024. Vol. 188. P. 108751.
4. Gong M. et al. An experimental study on local and global optima of linear antenna array syn-thesis by using the sequential least squares programming. *Applied Soft Computing*, 2023, vol. 148, p. 110859.
5. Dmitrieva T.L., Chan L.T.M. Optimal Design of Steel Structures Using ANSYS. *International Journal for Computational Civil and Structural Engineering*, 2014, vol. 10. pp. 79–84. (In Russ.). DOI: https://doi.org/10.22337/2587-9618
6. Dmitrieva T.L. Software package “OPTIDEST” and its use in problems of calculation and optimization of steel structures. *Bulletin of MGSU*, 2011, vol. 1, no. 1, pp. 100–105. (In Russ.).
7. Gommers R. et al. scipy/scipy: SciPy 1.9.0. *Zenodo*, 2022, p. 15.
8. Hill C. Optimization with scipy.optimize. *Python for Chemists*. Cambridge University Press, 2023. P. 247–259.
9. Melnikova V.A., Kupriyanova Yu.V. Application of special Python libraries for studying linear programming problems. *Improving the Quality of Education: Collection of Articles of the XX(XXXVI) All-Russian Scientific and Methodological Conference*. Bratsk: BrSU Publishing House, 2023. P. 146. (In Russ.).
10. Pevneva A.G., Kalinkina M.E. Optimization methods. St. Petersburg: ITMO University, 2020. P. 64. (In Russ.).
11. Revyakin M.A. Nelder – Mead method for solving nonlinear programming problems. *Priority Areas of Innovative Activity in Industry*. 2021. P. 78–80. (In Russ.).
12. Lazhauinkas Yu.V., Kochegarova O.S. One-dimensional optimization by the Powell method. *Economic and Mathematical Methods for Analyzing the Activities of Agricultural Enterprises*. 2022. P. 253–258. (In Russ.).
13. Powell M.J.D. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 1964, vol. 7, no. 2, pp. 155–162.

14. Klyaus K.M. Numerical methods of nonlinear optimization in mathematical modeling problems. General characteristics. 2021. P. 112. (In Russ.).
15. Dwail H.H., Shiker M.A.K. Using trust region method with BFGS technique for solving non-linear systems of equations. *Journal of Physics: Conference Series*. IOP Publishing, 2021, vol. 1818, no. 1, p. 012022.
16. Li L., Hu J. Fast-converging and low-complexity linear massive MIMO detection with L-BFGS method. *IEEE Transactions on Vehicular Technology*, 2022, vol. 71, no. 10, pp. 10656–10665.
17. Pratama D.A. et al. Solving partial differential equations with hybridized physic-informed neural network and optimization approach: Incorporating genetic algorithms and L-BFGS for improved accuracy. *Alexandria Engineering Journal*, 2023, vol. 77, pp. 205–226.
18. Nocedal J., Wright S.J. Numerical optimization. Second edition. 2006. P. 526–573.
19. Ali K.S.A. et al. Analysis of composite leaf spring using ANSYS software. *Materials Today: Proceedings*, 2021, pp. 2346–2351.
20. Bogdanova P.A., Sakharov D.M., Vasilyeva T.V. Review of multicriteria optimization methods in decision making problems. *Innovative Aspects of Science and Technology Development*, 2021, no. 6, pp. 153–157. (In Russ.).

ИНФОРМАЦИЯ ОБ АВТОРАХ | INFORMATION ABOUT THE AUTHORS

Дмитриева Татьяна Львовна – доктор технических наук, заведующий кафедрой механики и сопротивления материалов, Иркутский национальный исследовательский технический университет (Иркутск, Российская Федерация).
✉ dmitrievat@istu.edu; https://orcid.org/0000-0001-5460-4780
Tatiana L. Dmitrieva, Doctor of Engineering Sciences, Head of Mechanics and Strength of Materials, Irkutsk National Research Technical University (Irkutsk, Russian Federation).

Ботхоев Александр Евгеньевич – аспирант, Иркутский национальный исследовательский технический университет (Иркутск, Российская Федерация).
✉ botkhoev@ya.ru; https://orcid.org/0009-0007-3874-2770
Alexander E. Botkhoev, Postgraduate Student, Irkutsk National Research Technical University (Irkutsk, Russian Federation).

Статья поступила в редакцию / Received: 18.02.2025.
Доработана после рецензирования / Revised: 14.03.2025.
Принята к публикации / Accepted: 18.03.2025.