версия от 4 ноября 2022 г. 23:52

Численные методы, осень 2022
Задание 4 [Число обусловленности. Числа с плавающей точкой и вычислительная устойчивость]
Всего баллов: 37 Срок сдачи: 18 ноября

РЕКОМЕНДОВАННАЯ ЛИТЕРАТУРА

- Лекции 12–19, 20–23 из [1]
- Лекции 6–7 из [2]

УПРАЖНЕНИЯ

1. (3) Предложите вычислительно устойчивый способ вычислить функцию $f(x, a) = \sqrt{x + a} - \sqrt{x}$ при положительных x и a.

2. (2) Вычислите $\mathcal{C} = \tan(10^{100})$ с помощью модуля `mpmath`, предназначенного для арифметики произвольной точности. Пример использования:

```
from mpmath import *
mp.dps = 64        # точность (число десятичных цифр)
mp.pretty = True
+pi                # pi - переменная из mpmath
```

   Чему равно относительное число обусловленности при вычислении $\mathcal{C} = \mathcal{C}(10^{100})$? Сколько цифр нужно хранить в памяти при промежуточных вычислениях, чтобы получить $\mathcal{C}$ с точностью в 7 значащих цифр?

3. (4) Реализуйте функцию `solve_quad(b, c)`, возвращающую корни приведённого квадратного уравнения $x^2 + bx + c = 0$. Корни могут повторяться или быть комплексными.

   Когда вам покажется, что функция работает, запустите её на следующих пяти тестах. Добейтесь того, чтобы она правильно работала на каждом из них.

```
tests = [{'b': 4.0,   'c': 3.0},
         {'b': 2.0,   'c': 1.0},
         {'b': 0.5,   'c': 4.0},
         {'b': 1e10,  'c': 3.0},
         {'b': -1e10, 'c': 4.0}]
```

4. (5) Рассмотрите многочлен

$$w(x) = \prod_{r=1}^{20}(x-r) = \sum_{i=0}^{20} a_i x^i$$

и исследуйте число обусловленности его корней, выступающих в роли функций от коэффициентов a_i. Проведите эксперимент: случайным образом измените коэффициенты и найдите новые корни с помощью алгоритма из `numpy`. Коэффициенты изменяйте по правилу $a_i \to n_i a_i$, где n_i подчиняются нормальному распределению с математическим ожиданием, равным 1, и дисперсией, равной $\exp(-10)$. Проведите 100 таких экспериментов и изобразите результаты на одном графике вместе с корнями исходного многочлена.

   Оцените по одному из экспериментов абсолютное и относительное число обусловленности корней многочлена как функций его коэффициентов.

5. (10) Рассмотрим задачу наименьших квадратов — $Ax \approx b$:

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 1.00001 \\ 1 & 1.00001 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 0.00001 \\ 4.00001 \end{bmatrix}$$

   - Формально решение можно найти как

$$x = (A^T A)^{-1} A^T b. \tag{1}$$

     Вычислите его по этой формуле аналитически.

   - Вычислите (1) с помощью `numpy`, используя числа одинарной и двойной точности; сравните результат с аналитическим.

   - Помимо формулы (1), реализуйте решение, основанное на сингулярном разложении. Какой способ вычислительно более стабильный?

- Решите эту же задачу с помощью `np.linalg.lstsq`. Какой алгоритм использует эта функция?
- Какие четыре числа обусловленности, относящиеся к этой задаче, упоминаются в теореме 18.1 из [1]? (Возможно, их требуется вычислить — прим. пер.). Приведите примеры таких δb и δA, при которых приблизительно достигаются оценки на $\|\delta x\|$, даваемые числами обусловленности.

6. (7) Пусть

$$A = \begin{bmatrix} \epsilon & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

- Аналитически найдите LU-разложение матрицы A с применением выбора главного элемента и без него.
- Объясните, почему при $|\epsilon| \ll 1$ мы можем неправильно оценить множители L и U в арифметике конечной точности.

7. (6) Пусть функция $f(n, \alpha)$ определена следующим образом:

$$f(n, \alpha) = \frac{1}{n} - \alpha f(n-1, \alpha)$$
$$f(0, \alpha) = \ln(1 + 1/\alpha)$$

Вычислите $f(20, 0.1)$ и $f(20, 10)$ с помощью арифметики обычной (двойной) точности. Теперь сделайте то же самое в арифметике произвольной точности:

```python
from mpmath import mp, mpf
mp.dps = 64 # precision (in decimal places)
f = mp.zeros(1, n)
f[0] = mp.log(1 + 1/mpf(alpha))
for i in range(1, n):
    f[i] = 1/mpf(i) - mpf(alpha) * f[i-1]
```

Постройте в единицах машинного эпсилон график относительной разности между точными и приближёнными результатами как функции от n. Сделайте это при $\alpha = 0.1$ и при $\alpha = 10$. Машинный эпсилон можно получить как `np.finfo(float).eps`.
Как бы вы стали вычислять $f(30, 10)$ без арифметики произвольной точности?

[1] L. N. Trefethen and D. Bau III, *Numerical linear algebra*, Vol. 50 (Siam, 1997).

[2] E. E. Tyrtyshnikov, *A brief introduction to numerical analysis* (Springer Science & Business Media, 2012).