Отчет

**Метод вращений Якоби с выбором в качестве обнуляемого элемента максимального по модулю среди внедиагональных элементов.**

* n – размерность матрицы
* m – размерность блока матрицы, которая печатается на экран
* eps - точность нахождения собственных значений матрицы
* k –
  + 1,2,3,4 – номера формул, которая заполняет исходную матрицу
  + 0 – необходим 5-ый аргумент (имя файла)
* filename – имя файлы, откуда считывается матрица

**Описание метода:**

Выделяем память на матрицу размера n\*n и заполняем ее либо по формуле, либо считываем ее из файла (в зависимости от аргументов командной строки).

Находим максимальный по модулю внедиагональный элемент (в силу симметричности матрицы относительно диагонали ищем в верхней треугольной матрице (при умножении на ортогональную матрицу симметричность сохраняется)) – пусть этот элемент находится в i-ой строке и в j-ом столбце. Если модуль найденного элемента больше точности из командной строки Выполняем умножение исходной матрицы справа на ортогональную матрицу специального вида – на главной диагонали стоят 1, в i-ой строке в i-ом столбце стоит cos(x), в i-ой строке в j-ом столбце стоит -sin(x), в j-ой строке в i-ом столбце стоит sin(x), в j-ой строке в j-ом столбце стоит cos(x), где x – угол поворота в плоскости (i, j) такой, чтобы занулить рассматриваемых внедиагональный элемент (и, соответственно, симметричный ему) – находится из тригонометрических уравнений. Фактически умножение описанных матриц эквивалентно изменению лишь двух столбцов с номерами i и j. После этого умножаем полученную матрицу слева на ту же ортогональную матрицу (описанную выше), только транспонированную (она же является обратной). Последнее умножение эквивалентно изменению лишь двух строк предыдущей матрицы с номерами i и j. Таким образом, в результате выполнения обоих умножений мы получаем матрицу, отличающуюся от исходной матрицы лишь элементами, стоящими в двух горизонтальных и двух вертикальных полосах с номерами i и j.s

Описанный шаг метода Якоби проделываем многократно, пока модуль максимального внедиагонального элемента текущей матрицы больше точности, введенной из командной строки.

Также находятся две норма невзяки:

* невязку в первом инварианте: модуль разности следа исходной матрицы и суммы собственных значений
* невязку во втором инварианте: модуль разности длины исходной матрицы как вектора размера *n*2 и корня из суммы квадратов собственных значений

Приведу результаты времени и нормы невязки для моей программы:

|  |  |  |  |
| --- | --- | --- | --- |
| ввод | время, secs | невязка в первом инварианте | невязка во втором инварианте |
| ./a.out 50 3 1e-2 2 | 6.560e-03 | 5.684e-14 | 1.274e-03 |
| ./a.out 100 3 1e-2 2 | 5.290e-02 | 1.421e-13 | 4.049e-03 |
| ./a.out 200 3 1e-2 2 | 4.345e-01 | 1.832e-13 | 1.066e-02 |
| ./a.out 1000 3 1e-2 2 | 5.731e+01 | 6.821e-13 | 3.876e-02 |

Рост времени происходит пропорционально n3:

100/50 = 2 🡨🡪 5.290e-02/6.560e-03 = 8.064 ~ 23

200/50 = 4 🡨🡪 4.345e-01/6.560e-03 = 66.235 ~ 43

1000/50 = 20 🡨🡪 5.731e+01/6.560e-03 = 8736.280 ~ 203