

Лабораторная работа по курсу "Вычислительные методы линейной алгебры"

Максимальное количество баллов: 15.

1. Задание 1: построение LU -разложения матрицы, метод Гаусса для решения СЛАУ.

- (a) (1 балл) Необходимо написать программу `make_lu`, которая
- считывает из заданного текстового файла (вида "Amat*.m", где * – это какое-то число) матрицу A ;
 - проверяет, является ли матрица A квадратной и строго регулярной (если эти условия не выполнены, то работа программы завершается с сообщением об ошибке);
 - строит LU -разложение матрицы по формулам из лекции 7 (То есть $A = LU$, L – нижняя треугольная матрица с единицами на главной диагонали, U – верхняя треугольная матрица.);
 - результаты расчетов записывает в двух новых текстовых файлах вида "Lmat*.m" и "Umat*.m", где * – это то же число, которое было указано в названии файла с матрицей A .
- (b) (1 балл) Необходимо улучшить функцию, написанную в предыдущем пункте. Добавить возможность перестановки строк матрицы A так, чтобы на каждой итерации алгоритма поиска LU -разложения выбирался наибольший ведущий член. Новая функция `make_lu_upgrade` должна производить следующие действия:
- считывает из заданного текстового файла (вида "Amat*.m", где * – это какое-то число) матрицу A ;
 - проверяет, является ли матрица A квадратной и строго регулярной (если эти условия не выполнены, то работа программы завершается с сообщением об ошибке);
 - строит LU -разложение матрицы по формулам из лекции 7, с выбором наибольшего ведущего члена a (при этом производится перестановка строк матрицы A при помощи матрицы перестановки P ; в результате должно получиться $PA = LU$);
 - результаты расчетов записывает в трёх новых текстовых файлах вида "Pmat*.m", "LmatUp*.m" и "UmatUp*.m", где * – это то же число, которое было указано в названии файла с матрицей A .
- (c) (2 балла) Необходимо написать программу `lu_gauss`, которая
- считывает из заданного текстового файла (вида "Amat*.m", где * – это какое-то число) матрицу A ;
 - считывает из заданного текстового файла (вида "bvec*.m", где * – это какое-то число (то же, что и для матрицы A)) вектор b ;
 - проверяется, что число строк матрицы A совпадает с числом элементов вектора b ;
 - проверяет, было ли ранее подсчитано LU -разложение (то есть есть ли уже готовые файлы "Lmat*.m" и "Umat*.m"), либо улучшенная версия этого разложения (то есть есть ли уже готовые файлы "Pmat*.m", "LmatUp*.m" и "UmatUp*.m"). Если подходящих файлов нет, то запускается функция подсчета LU -разложения (программа `make_lu_upgrade` или `make_lu`);
 - из соответствующих файлов считываются матрицы L , U (и возможно P);
 - при помощи считанных матриц решается СЛАУ $Ax = b$;
 - результат решения – вектор x записывается в отдельный текстовый файл вида "xvec*.m", где * – это то же число, что было указано в матрице A .
- (d) (2 балла) Улучшить функции `make_lu`, `make_lu_upgrade`, `lu_gauss` так, чтобы можно было в матрицах и векторах использовать не только вещественные, но и комплексные числа.
- (e) (1 балл) Запустить функцию `lu_gauss` для матрицы A и вектора b очень большого размера (файлы `Amat5.m`, `bvec5.m`). Нужно 1) замерить и прислать время работы программы; 2) прислать получившийся файл `xvec5.m`.

Замечание. Строгую регулярность матрицы A можно не проверять предварительно. Достаточно вставить проверку на корректность расчетов (чтобы не получилось деление на ноль) на каждой итерации алгоритма.

Тестовые примеры:

- Файлы `Amat1.m`, `Lmat1.m`, `Umat1.m`. Матрицы размера 2×2 . Можно использовать для тестирования функции `make_lu`.
- Файлы `Amat2.m`, `LmatUp2.m`, `UmatUp2.m`, `Pmat2.m`. Матрицы размера 3×3 . Можно использовать для тестирования функции `make_lu_upgrade`.
- Файлы `Amat3.m`, `bvec3.m`, `xvec3.m`. Матрица размера 3×3 и вектор длины 3. Можно использовать для тестирования функции `lu_gauss`.
- Файлы `Amat4.m`, `LmatUp4.m`, `UmatUp4.m`, `Pmat4.m`. Матрицы размера 3×3 . Можно использовать для изучения формата записи матриц с комплексными числами и для тестирования функции `make_lu_upgrade`.

По каждой написанной программе необходимо прислать исходный код. Кроме того, в пункте (е) нужно прислать получившийся файл `xvec5.m` и (в отдельном файле `'task_1e.txt'`) указать время работы программы на данном тесте, а также указать, какая из функций (`make_lu` или `make_lu_upgrade`) была использована.

2. Задание 2: построение разложения Холецкого симметричной матрицы.

- (a) (1 балл) Необходимо написать программу `make_chol`, которая
- считывает из заданного текстового файла (вида `"Amat*.m"`, где `*` – это какое-то число) матрицу A ;
 - проверяет, является ли матрица A квадратной, симметричной, положительно определённой (если эти условия не выполнены, то работа программы завершается с сообщением об ошибке);
 - строит разложение Холецкого матрицы по формулам из лекции 7 (То есть $A = CC^*$, C – нижняя треугольная матрица с положительной главной диагональю.);
 - результаты расчетов записывает в новом текстовом файле вида `"Cmat*.m"`, где `*` – это то же число, которое было указано в названии файла с матрицей A .
- (b) (2 балла) Необходимо написать программу `chol_gauss`, которая
- считывает из заданного текстового файла (вида `"Amat*.m"`, где `*` – это какое-то число) матрицу A ;
 - считывает из заданного текстового файла (вида `"bvec*.m"`, где `*` – это какое-то число (то же, что и для матрицы A)) вектор b ;
 - проверяется, что число строк матрицы A совпадает с числом элементов вектора b , что матрица A является симметричной и положительно определённой;
 - проверяет, было ли ранее подсчитано разложение Холецкого (то есть есть ли уже готовый файл `"Cmat*.m"`). Если подходящего файла нет, то запускается функция подсчета разложения Холецкого (программа `make_chol`).
 - из соответствующего файла считывается матрица C ;
 - при помощи считанной матрицы решается СЛАУ $Ax = b$;
 - результат решения – вектор x записывается в отдельный текстовый файл вида `"xvec*.m"`, где `*` – это то же число, что было указано в матрице A .
- (c) (1 балл) Запустить функцию `chol_gauss` для матрицы A и вектора b очень большого размера (файлы `Amat8.m`, `bvec8.m`). Нужно 1) замерить и прислать время работы программы; 2) прислать получившийся файл `xvec8.m`.

Тестовые примеры:

- Файлы `Amat6.m`, `Cmat6.m`. Матрицы размера 5×5 . Можно использовать для тестирования функции `make_chol`.
- Файлы `Amat7.m`, `bvec7.m`, `xvec7.m`. Матрица размера 3×3 и вектор длины 3. Можно использовать для тестирования функции `chol_gauss`.

По каждой написанной программе необходимо прислать исходный код. Кроме того, в пункте (с) нужно прислать получившийся файл `xvec8.m` и (в отдельном файле `'task_2c.txt'`) указать время работы программы на данном тесте.

3. Задание 3: построение QR -разложения матрицы.

- (a) (1 балл) Необходимо написать программу `make_qr`, которая
- считывает из заданного текстового файла (вида `"Amat*.m"`, где `*` – это какое-то число) матрицу A ;
 - проверяет, является ли матрица A квадратной (если это условие не выполнено, то работа программы завершается с сообщением об ошибке);
 - строит QR -разложение матрицы по формулам из лекции 8 (модифицированный алгоритм Грама-Шмидта, с дополнительными действиями для борьбы с потерей ортогональности);
 - результаты расчетов записывает в двух новых текстовых файлах вида `"Qmat*.m"` и `"Rmat*.m"`, где `*` – это то же число, которое было указано в названии файла с матрицей A .
- (b) (2 балла) Необходимо написать программу `qr_gauss`, которая
- считывает из заданного текстового файла (вида `"Amat*.m"`, где `*` – это какое-то число) матрицу A ;
 - считывает из заданного текстового файла (вида `"bvec*.m"`, где `*` – это какое-то число (то же, что и для матрицы A)) вектор b ;
 - проверяется, что число строк матрицы A совпадает с числом элементов вектора b , что матрица A является квадратной и невырожденной;
 - проверяет, было ли ранее подсчитано QR -разложение (то есть есть ли уже готовые файлы `"Qmat*.m"`, `"Rmat*.m"`). Если подходящего файла нет, то запускается функция подсчета QR -разложения (программа `make_qr`).

- из соответствующего файла считываются матрицы Q , R ;
 - при помощи считанных матриц решается СЛАУ $Ax = b$;
 - результат решения – вектор x записывается в отдельный текстовый файл вида "xvec*.m", где * – это то же число, что было указано в матрице A .
- (с) (1 балл) Запустить функцию `qr_gauss` для матрицы A и вектора b очень большого размера (файлы `Amat11.m`, `bvec11.m`). Нужно 1) замерить и прислать время работы программы; 2) прислать получившийся файл `xvec11.m`.
- (d) (2 балла) Реализовать функцию `qr_algo`, которая производит приближённый поиск собственных значений матрицы при помощи QR -разложения (см. материал лекции 11). (Обратите внимание, что собственные значения могут быть **комплексными** – в программе надо предусмотреть арифметические действия с комплексными числами.) Функция
- считывает из заданного текстового файла (вида "Amat*.m", где * – это какое-то число) матрицу A ;
 - проверяется, что матрица A является квадратной;
 - при помощи функции `make_qr` строится матрица \hat{A} (количество итераций определяется из условия малости элементов, которые в пределе должны стать нулевыми);
 - результат решения – вектор λ из собственных значений записывается в отдельный текстовый файл вида "lvec*.m", где * – это то же число, что было указано в матрице A .

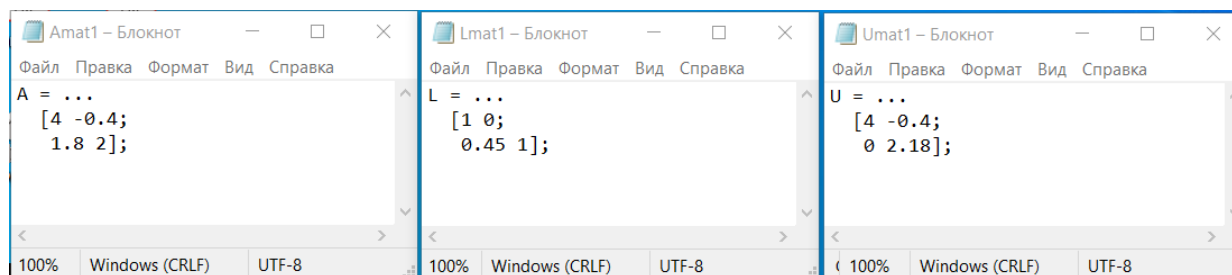
Тестовые примеры:

- Файлы `Amat9.m`, `Qmat9.m`, `Rmat9.m`. Матрицы размера 3×3 . Можно использовать для тестирования функции `make_qr`.
- Файлы `Amat10.m`, `bvec10.m`, `xvec10.m`. Матрица размера 3×3 и вектор длины 3. Можно использовать для тестирования функции `qr_gauss`.
- Файлы `Amat10.m`, `lvec10.m` можно использовать для тестирования функции `qr_algo`.

По каждой написанной программе необходимо прислать исходный код. Кроме того, в пункте (с) нужно прислать получившийся файл `xvec11.m` и (в отдельном файле 'task_3c.txt') указать время работы программы на данном тесте.

Общие замечания:

- Перед тем, как начинать программировать конкретные алгоритмы рекомендуется написать вспомогательные функции для чтения и записи в текстовые файлы матриц. Эти функции (одни и те же) далее могут быть использованы в каждом из заданий. Каждая матрица записывается в отдельном текстовом файле, например, вида "Amat.m". Пример:



Здесь представлены три файла с матрицами размера 2×2 . Первая строка файла всегда содержит имя матрицы, знак равенства и три точки. Конструкция с числами матрицы заключается внутри квадратных скобок. Каждая строка матрицы задаётся отдельно, в одной строке файла. Между числами строки указываются символы-разделители 'пробел'. Символ ';' указывает на конец строки (переход к следующей строке). Этот же символ указывается в самом конце матрицы, после квадратной скобки. При записи действительных чисел в качестве символа, отделяющего целую часть числа от дробной, используется 'точка' ('.').

- В программах **запрещено использовать готовые функции**, позволяющие строить разные разложения матриц, обращать матрицы и т.п. То есть все вычисления должны быть произведены только за счет использования элементарных операций.