# Лабораторная работа №1 по предмету Методы Вычислений

Автор: Томашевич Константин, Вариант 7.

## Часть 1

С помощью обычного rand() было бы довольно сложно и костыльно генерировать десятичные дроби з 13 знаками после запятой, RAND\_MAX же ~32000, да и rand() не такой уж и рандомный, как хотелось бы. Поэтому я решил использовать генератор случайных чисел Mersenne Twister, который используется в GMP. Код его я, конечно же, стырил, ибо конкретно он не часть лабы. Добивался наличия хотя бы 13 значащих цифр после запятой я таким образом:

1. gen <- random number in [-4, 4].
2. If fractional of gen \* 10^13 lower than 10^-4 then gen <- gen + 10^-13.
3. return gen.

## Часть 2

Пожалуй, наименее интересная часть лабораторной работы. Метод Гаусса-Жордана пишется довольно просто и топорно, но работает при этом чрезвычайно быстро :sarcasm:. Вот зачем нужно было выдумывать всякие гмресы с арнольди, если можно просто потратить всего в раз в 35 (в случае 256х256 матриц) больше времени и просто решить ? Впрочем, недавно учёные нашли Илью, который умеет считать за . Возможно, это и есть будущее алгебры, только Илья пока что отказывается делиться этим секретным алгоритмом.

Теперь по делу: число обусловленности для всех матриц колеблется в интервале [4.5; 5], что не так уж и плохо выглядит в общем случае, но вот если вспомнить, что , всё выглядит уже не так радужно. Но, к счастью, непреодолимое желание приблизиться к числу обусловленности испытывала только евклидова норма разницы реального решения и решения через МНК, с остальными как-то обошлось.

Заметим, что средний элемент матрицы таки совсем не ноль. Но мы же генерировали её рандомно! Нас обманули и мерсен твистер не работает? Таки не обманули, просто диагонали всё-таки заполнялись не рандомно (диагональное доминирование делали же), вот и не ноль.

## Часть 3

Наконец-то мы перешли к реальному решению СЛАУ. И вот у нас метод Гаусса с выбором максимального элемента по всей матрице. По правде говоря, с написанием этого метода я запарился больше всего, ибо тут и порядок решений нужно в конце восстановить, и постоянно перекидывать строки и столбцы в процессе. Запарно и дебажить пришлось.

Благодаря тому, что матрицы мы, всё-таки, генерируем довольно хорошие для этого алгоритма, евклидова норма разности решений колеблется в промежутке , что довольно таки точно. Для эксперимента я попробовал отключить выбор максимального элемента по матрице, в результате чего максимальная норма поднялась до , то есть в 1000 раз! Тем самым я убедился, что такая противная операция, как выбор максимального элемента по матрице, не просто имеет право на существование, а ещё и даёт очень даже хороший результат. Но всё-таки насколько попаболь была её дебажить...

Теперь рассмотрим время выполнения, оно составляет в среднем 41мс на матрицах 256х256, что почти в 3.5 раза быстрее поиска обратной матрицы по Гауссу-Жордану. Но стоп, мы же выполняем почти те же операции! Да ещё и играем со строками и колонками (и шрифтами), строки-то можно поменять местами за О(1), а вот столбцы только за О(n). Чего тогда так быстро? А штука просто в том, что в Гаусе у нас справа один столбец b, а не целая матрица. То есть разница в скорости очевидна.

## Часть 4

После написания и дебага обычного Гаусса, написать было уже не настолько сложно. Да, я думал, что LU надо тоже делать в выбором максимального элемента по матрице. Но после адского Гаусса с той же болью уже не было так сложно.

Сначала поговорим про евклидову норму разности решений. Впрочем, про неё говорить, возможно, и не стоит, так как она до последней цифры совпадает с той же нормой у обычного Гаусса. И удивляться тут нечему – так как мы делаем фактически то же самое, только «запоминаем» ходы Гаусса в объединённую LU-матрицу (так-как свойства матриц L и U позволяют совместить их в одну для хранения) и перестановки строк и столбцов в матрицы и (которые в памяти можно хранить просто как вектора перестановки), потом просто повторяем все нужные действия с данным вектором b.

А вот что насчёт времени? строится за 40мс, на 1мс быстрее, чем решается СЛАУ па Гауссу. Почему? Но ведь при построении мы не делаем обратного хода. На решение СЛАУ с построенным разложением уходит всего 1мс, что довольно логично, ведь мы просто мучаем один столбец b. Никакой магии, конечно, не происходит, ведь 40+1=41мс – тот же Гаусс. Зато можно решать потом много СЛАУ на очень большой скорости, благодаря уже полученному разложению.

## Часть 5

Метод квадратного корня, полученный путём поиска производной от разложения Холецкого (образно говоря), позволяет довольно эффективно ускорить вычисление решения СЛАУ, матрица коэффициентов которого симметрична. Принцип очень даже прост: находим разложение Холецкого , дальше тоже, что и с LU. D тоже удобно хранить как вектор, кстати. И разложение можно также пускать на конвейер, как и LU.

Точность получилась такой же, как и у Гаусса, а именно евклидова норма находится в промежутке . Но тут следует заметить, что при поиске разложения Холецкого, в отличии от того же поиска LU, никаких лайфхаков по типу выбора максимального элемента по матрице не используется, так как это просто нарушило бы симметричную структуру матрицы. Но, тем не менее, разложение Холецкого и без этого хорошо справляется с точностью решения.

По времени решения метод квадратного корня также показал себя достаточно хорошо: 26мс против 41мс Гауссовских. Ну что, шах и мат? Нет, ведь метод квадратного корня специально создан для быстрого решения СЛАУ с **симметричными** матрицами коэффициентов, с другими его и провести, собственно, невозможно. Так что со своим делом этот метод справляется очень даже хорошо, а для всего остального можно и Гаусса использовать, при желании.

## Часть 6

Перейдём к методу релаксаций. С его доказательством я не особо знаком, ибо «на лекциях не обучался» (и в гимназиях тоже). Вследствие вышеописанных факторов чрезвычайной академической неграмостности этот чрезвычайно интересный метод решения СЛАУ кажется мне магией. А ещё его и написать было очень даже проще, не то что долбанный Гаусс с выбором максимального элемента.

Сначала рассмотрим точность решения через евклидову норму разности: это будет промежуток . Собственно, поразительно точно для итерационного метода. Они же созданы для быстрого и не совсем точного получения решения путём жертвоприношения конспектов одногрупников.

Перейдём к вопросу времени. И вот тут ещё один шокирующий факт: всего 9 долбанных миллисекунд! Что это такое? Разрыв шаблонов и победа неформалов? Может где-то на фоне JS-сники с радостными воплями пишут свою ОС, которая будет «лучше линукса»? Они-то может и пишут, но с методом релаксации другая фигня. А именно штука в том, что он работает только на специфическом классе матриц (как и все классические итерационные методы), а сгенерированные по данному нам правилу матрицы вообще близки к идеальному случаю для этого метода, что вполне себе объясняет такой крутой результат. То есть метод релаксации, конечно, крут, но он именно что «широко известен(применяем) в узких кругах».

## Часть 7

Теперь к методу отражений, он метод Хаусхолдера. Этот метод показался мне чрезвычайно интересным с той стороны, что он и не итерационный, и не является ещё одной модификацией поднадоевшего метода Гаусса.

С точностью решения, к сожалению, получилось не очень хорошо: евклидова норма разности решений , что выглядит угрожающе по сравнению с тем же Гауссом. Мне кажется, что проблема в том, что Гауссу наша матрица очень даже по душе, а вот методу отражений она может быть и не очень удобной. Даже если она нормальная для метода отражений, она всё ещё остаётся чрезвычайно удобной для Гаусса. То есть у нас получается, что «нормальная» для метода отражений матрица была им решена худше, чем Гауссом «идеальная» для Гаусса матрица. Тут достаточно сложно сравнивать.

С временем тоже вышло как-то проблемно: 49мс против 41мс у Гаусса. Что тут можно сказать? У метода отражений две основные задачи: построить QR и закрыть критические точки Гаусса. QR он строит, матрицы в общем случае (а не «идеальном для Гаусса») тоже должен портить значительно меньше. В данном случае у нас не получилось доказать, что Хаусхолдер лучше, но мне кажется, что это именно из-за правила генерации матрицы. А может и просто я метод отражений криво написал.

## Часть 8

Метод наименьших квадратов прост как палка, но и палка раз в год стреляет. Впрочем, МНК, возможно, и реже. Фича, так сказать. Сам по себе метод чрезвычайно идейно прост и понятен, поэтому и в интернете есть его множество вариаций. Я даже встретил в интернете на сайте одного физического факультета версию, где сразу был выведен Х. И там предлагали считать по этой формуле (с поиском обратной, конечно же)! Но, к сожалению, в нашей действительности так может делать только студент Илья, который находит обратную матрицу за О(1). Менее гениальным студентам, думаю, лучше всего будет использовать метод квадратного корня, так как заведомо симметрична.

Точность решения у МНК описывается только нецензурной лексикой и евклидовой нормой: , что просто ну \*ец как жесточайше. Но на то он и метод наименьших квадратов: просто и без вкуса. К слову, с другими ядрами (например Гауссом или Хаусхолдером) ситуация была ещё плачевней. Также надо заметить, что в моём варианте берётся довольно большое количество столбцов из сгенерированной A, то есть МНК тут используется, на мой взгляд, немного не по назначению.

Со временем выполнения тоже есть некоторые проблемы: это 27мс, что многовато для такой фиговой нормы. Я это объяснил бы тем же: по-моему, с моим вариантом (140 столбцов) МНК не очень дружит, то есть задача поставлено неудобно для применения МНК. Может и я, конечно, что-то наговнокодил. Ну или метод вообще очень плохой.

## Часть 9

Мне очень понравилась идея GMRES – как метод отражений решает СЛАУ совсем не по-гауссовски, так и GMRES, хоть он и итеративный, действует по совершенно отличной от Гаусса-Зейделя и релаксаций методом. При этом метод кажется довольно изящным и красивым. Свежие и новые идеи затмевают молодые умы, короче.

Что насчёт точность? Евклидова норма даёт достаточно большой разброс: , заметим, что её среднее значение -- , то есть результат довольно точный, за исключением редких случаев. С точностью метода релаксаций, конечно, не сравниться, зато использовать GMRES можно абсолютно для всех матриц, для которых можно использовать классического Гаусса, что уже делает GMRES довольно крутым. И ведь такая точность достигнута с использованием адски неточного МНК!

Но основной плюс GMRES’а в его скорости: даже на пространстве Крылова он справился с СЛАУ с указанной точностью за всего 6 миллисекунд! В полтора раза быстрее релаксаций, для которых эта матрица почти что идеальна, хоть и точность пострадала. Такой скоростью GMRES обязан хорошо продуманной схемой вычислений, в которой не так уж и много приходиться пересчитывать.

## Часть 10

GMRES – сам по себе уже круто, а с использованием алгоритма Арнольди ещё круче. Тут у нас и все вектора для выражения Х попарно ортогональны, что даёт возможность сделать намного меньше итераций. Да и подаваемое на МНК выражение значительно проще, чем в обычном GMRES: тут и в b почти одни нули, и матрица коэффициентов гейзенбергова. Всё круто, короче.

Точность улучшенная Арнольди версия даёт действительно лучшую: евклидова норма находиться в промежутке , причём её среднее значение -- . То есть по точности уже всё лучше. Неужели ортогональность векторов так помогла? Штука в том, что в этой версии алгоритма мы подаём МНК значительно более простую матрицу с совсем уж простым решением. Уж гейзенбергову матрицу с почти нулевым столбцом b очень сложно попортить, даже для МНК.

Но основная цель использования алгоритма Арнольди – повышение скорости вычисления за счёт выкидывания «мусорных» итераций, то есть тех итераций, где фактический базис подпространства, через которое мы выражаем Х, не увеличивается. И это было достигнуто: GMRES с алгоритмом Арнольди работает в среднем в полтора раза быстрее на сгенерированых в матрицах. В среднем он тратил 4мс против 6мс у простого GMRES. Хз почему, но мне просто доставляет этот алгоритм решения СЛАУ.

## Таблица результатов

|  |  |
| --- | --- |
|  | Метод Гаусса-Жордана |
| Максимальное число обусловленности | 4.9973427950896072 |
| Минимальное число обусловленности | 4.5600691315216730 |
| Средний элемент матрицы | 1.6766354119699938 |
| Среднее время выполнения, мс | 152 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Гаусс | LU | Холецкий |
| Максимальная норма | 0.0000000000000396 | 0.0000000000000396 | 0.0000000000000397 |
| Минимальная норма | 0.0000000000000301 | 0.0000000000000301 | 0.0000000000000286 |
| Средняя норма | 0.0000000000000336 | 0.0000000000000336 | 0.0000000000000331 |
| Среднее время выполнения, мс | 41 | 40 (build), 1(solve) | 26 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Релаксации | Хаусхолдер | МНК |
| Максимальная норма | 0.0000000000000216 | 0.0000000000002913 | 2.5890708815126460 |
| Минимальная норма | 0.0000000000000154 | 0.0000000000000688 | 1.8309969488103850 |
| Средняя норма | 0.0000000000000185 | 0.0000000000000939 | 2.2629771119112743 |
| Среднее время выполнения, мс | 9 | 49 | 27 |

|  |  |  |
| --- | --- | --- |
|  | GMRES | GMRES+Arnoldi |
| Максимальная норма | 0.0001088217298520 | 0.0001088185085487 |
| Минимальная норма | 0.0000014093526386 | 0.0000005336945648 |
| Средняя норма | 0.0000096058395496 | 0.0000058194810650 |
| Среднее время выполнения, мс | 6 | 4 |

## Список литературы

