Задание 1:
решение систем линейных уравнений

Москва
2023

1 О решении линейных систем

Линейные системы необходимо решать в огромном числе самых разных задач. Мы будем работать с системами вида

$$Ax = b, \quad \text{где} \;\; A \in \mathbb{R}^{n\times n}, \;\; x, b \in \mathbb{R}^n, \tag{1}$$

то есть рассматривать системы с квадратной матрицей. Эти системы у нас будут возникать в результате дискретизации уравнений в частных производных. В таких задачах A обычно является *разреженной*: число ненулевых элементов составляет, как правило, $\mathcal{O}(n)$. В случае «хороших» уравнений и дискретизаций может выполняться также $A = A^T > 0$, но это бывает далеко не всегда.

1.1 Итерационные методы

Методы решения линейных систем делятся на два типа: прямые и итерационные. Среди итерационных особую популярность имеют методы на подпространствах Крылова, среди которых:

- *Метод сопряженных градиентов* (CG). Работает для $A = A^T > 0$, имеет трехчленные рекуррентные соотношения, благодаря чему не требует много памяти;
- *Обобщенный метод минимальных невязок* (GMRES). Ослабляет ограничения на матрицу, но требует хранения базиса подпространства, за счет чего требует много памяти. Для устранения проблемы может перезапускаться каждые m итераций (GMRES(m));
- *Стабилизированный метод бисопряженных градиентов* (BiCGStab или Bi-CGSTAB). Имеет трехчленные рекуррентные соотношения, как и CG, но не имеет тех ограничений на матрицу. Крайне распространен на практике
- Многие другие методы.

Эти методы хорошо подходят для систем с разреженными матрицами (почему?).

1.2 Переобуславливатели

Сходимость крыловских методов определяется числом обусловленности A. Для ускорения сходимости применяется так называемое *(левое) переобуславливание* – приведение системы (1) к виду

$$BAx = Bb, \tag{2}$$

где матрица B называется *переобуславливателем*, и у матрицы BA должно быть улучшенное число обусловленности. Конечно, лучшим переобуславливателем была бы обратная матрица $B = A^{-1}$, тогда итерационные методы сходились бы за 1 итерацию, но это неоправданно дорогостоящий переобуславливатель. На практике надо искать баланс между силой переобуславливателя и трудоемкостью его вычисления.

Популярными переобуславливателями общего назначения (типа «черный ящик») являются всевозможные методы на основе неполной факторизации, например, ILU – неполное LU-разложение, где часть элементов отбрасывается по тем или иным признакам (вхождение в портрет матрицы A^{k+1} – ILU(k), отброс по малости элемента – ILUT и др.)

1.3 CSR-формат для записи разреженных матриц

Поскольку матрицы разреженные, то хранить n^2 чисел в памяти не имеет смысла. Используется специальный формат записи CSR (compressed sparse row). Подробнее с ним можно ознакомиться в [1].

2 Задание

Каждому студенту дается два набора матриц в формате CSR и два метода решения линейных систем. Задание состоит из двух частей. Первая следует классической методичке по СВТ [2], вторая является новой.

2.1 Часть 1: GMRES + ILU(k) из SPARSKIT

Ю. Саад – создатель метода GMRES, классической книги по итерационным решателям [3] и библиотеки SPARSKIT [4], содержащей различные итерационные методы и переобуславливатели. Необходимо установить SPARSKIT, написать код на Фортране, который подгружает матрицу, генерирует правую часть, строит переобуславливатель ILU(k) и запускает GMRES. Нужно замерить время построения переобуславливателя и время совершения итераций, а также число итераций.

- Матрицы: по адресу `https://old.inm.ras.ru/vtm/svt/matr.tgz`
- Правая часть: $b_i = sin(i)$

В отчет необходимо поместить таблицу, аналогичную первой таблице на стр. 16 в методичке [2], а также выводы по сложности вида «время построения переобуславливателя – $\mathcal{O}(n^{\alpha})$», «время итераций – $\mathcal{O}(n^{\beta})$», «число итераций – $\mathcal{O}(n^{\gamma})$»

Варианты заданий отличаются числом k в ILU(k). По списку группы это число распределено как 0, 1, 2, 0, 1, 2 и т.д.

2.2 Часть 2: BiCGStab + ILU2 из INMOST

Во второй части используется развивающаяся в ИВМ РАН платформа INMOST [5, 6]. Эта платформа представляет широкий набор средств для численного решения уравнений на неструктурированных сетках в параллельном режиме. С большей частью функционала мы познакомимся позже, а для начала освоим линейные решатели. Необходимо писать код на `C++` и использовать `CMake` (разберем на паре).

Решателем по умолчанию в INMOST является BiCGStab. В качестве переобуславливателя используем ILU2 – переобуславливатель на основе неполного LU-разложения, отбрасывающий элементы по малости (параметр `drop_tolerance`).

Линейные системы существуют в виде 3 архивов с матрицами и правыми частями в формате CSR, номера архивов распределены так же по списку группы как 0, 1, 2 и т.д. Системы из архивов получены в ходе дискретизации уравнений упругости методом виртуальных элементов [7] на последовательностях измельчающихся сеток разных типов.

Необходимо:

1. Написать код, который загружает матрицу и правую часть, строит переобуславливатель ILU2 и запускает BiCGStab;
2. Аналогично части 1 составить табличку с временами построения переобуславливателя и решения системы, повторить это для значений `drop_tolerance` 0.1, 0.01, 0.001;

3. Воспользовавшись функцией рисования портрета матрицы (`DrawMatrix`) из репозитория [8], нарисовать портрет одной из матриц своего набора и вставить в отчет.

3 Отчетность

Все материалы (код и отчет) должны быть загружены в репозиторий (как я понимаю, у всех есть гитхаб). Отчет должен быть сделан в LaTeXв жестко не регламентированной, но структурированной и читаемой форме.

Список литературы

[1] https://ru.wikipedia.org/wiki/Разреженная_матрица

[2] https://www.inm.ras.ru/wp-content/uploads/library/Monographies/yuv-kapyrin-svt-prak.pdf

[3] Saad Y. Iterative methods for sparse linear systems. – Society for Industrial and Applied Mathematics, 2003.

[4] https://www-users.cse.umn.edu/ saad/software/SPARSKIT/.

[5] https://github.com/INMOST-DEV/INMOST

[6] Vassilevski Y. et al. Parallel finite volume computation on general meshes. – New York : Springer International Publishing, 2020.

[7] https://arxiv.org/ftp/arxiv/papers/1311/1311.0932.pdf

[8] https://github.com/INMOST-DEV/INMOST-Graphics