Методы решения СЛАУ большой размерности

М.Ю.Баландин Э.П.Шурина

1 августа 2000 г.

Аннотация

Документ является электронной версией одноименного учебного пособия, выпущенного в июне 2000 г. в издательстве Новосибирского Государственного Технического Университета (НГТУ). Исправлены замеченные в печатном издании опечатки и неточности.

Документ может свободно распространяться при условии отсутствия внесения изменений и неизвлечения коммерческой выгоды. При цитировании необходимо ссылаться на следующее издание:

Баландин М.Ю., Шурина Э.П.
Методы решения СЛАУ большой размерности. —
Новосибирск: Изд-во НГТУ, 2000. — 70 стр.

Связаться с авторами можно по следующим адресам электронной почты:

dolphin@gts.nsk.su (М.Ю.Баландин)
shurina@online.sinor.ru (Э.П.Шурина)

Пособие подготовлено в системе $\LaTeX$

Введение

Развитие вычислительной техники и вызванный этим процессом переход к более сложным (трехмерным, в произвольных геометрических областях) моделям в виде систем дифференциальных уравнений в частных производных и их дискретным аналогам на неструктурированных сетках, привел к необходимости решения больших разреженных систем линейных алгебраических уравнений с матрицами нерегулярной структуры.

Наиболее эффективными и устойчивыми среди итерационных методов решения таких систем уравнений являются так называемые проекционные методы, и особенно тот их класс, который связан с проектированием на подпространства Крылова. Эти методы обладают целым рядом достоинств: они устойчивы, допускают эффективное распараллеливание, работу с различными строчными (столбцовыми) форматами и предобусловливателями разных типов.

В данном учебном пособии рассматриваются форматы хранения разреженных матриц и реализация операций над ними (в частности, матрично-векторное умножение), предобусловливание (неполное LU-разложение), классические итерационные и проекционные методы на подпространствах Крылова. Приведены основные теоретические обоснования, рассмотрены системы с симметричными и несимметричными матрицами, приведены конкретные примеры. Для желающих использовать при реализации методов готовое программное обеспечение в приложении даны соответствующие Internet-адреса.

Используемые обозначения и соглашения

Основной рассматриваемой задачей будет являться решение СЛАУ

$$\mathbf{A}x = b \quad (1)$$

с квадратной невырожденной матрицей $\mathbf{A}$ и ненулевым вектором b размерности n , составленными из действительных коэффициентов.

В силу невырожденности матрицы у системы (1) существует некоторое точное решение $x_* = \mathbf{A}^{-1}b$.

Невязкой системы (1), соответствующей вектору x , будем называть вектор $r = b - \mathbf{A}x = \mathbf{A}(x_* - x)$. Здесь вектор $\vartheta = x_* - x$ есть ошибка решения.

Линейный оператор A , заданный матрицей $\mathbf{A}$, действует в пространстве $\mathcal{R}^n$. Введем в нем скалярное произведение

$$(x, y) = \sum_{j=1}^n x_j y_j. \quad (2)$$

Для действительного случая сопряженный оператор A^* определяется матрицей $\mathbf{A}^T$, так что $(\mathbf{A}x, y) = (x, \mathbf{A}^T y)$. Иногда также используется удобным использование скалярного $\mathbf{A}$ -произведения

$$(x, y)_{\mathbf{A}} = (\mathbf{A}x, y) = (x, \mathbf{A}^T y) = y^T \mathbf{A}x = x^T \mathbf{A}^T y. \quad (3)$$

На основе скалярных произведений (2) и (3) введем в $\mathcal{R}^n$ векторные нормы

$$\begin{aligned} \|x\|_2 &= \sqrt{(x, x)}; \\ \|x\|_{\mathbf{A}} &= \sqrt{(\mathbf{A}x, x)} = \sqrt{(x, x)_{\mathbf{A}}} \end{aligned}$$

(евклидова норма и $\mathbf{A}$ -норма соответственно).

Если явно не оговорено обратное, то будет предполагаться, что матрица $\mathbf{A}$ не обладает свойством симметрии и положительной определенности ($\mathbf{A}$ -норма, очевидно, применима лишь к таким матрицам¹, иначе она вообще не является нормой).

¹Часто называемым SPD-матрицами (Symmetric and Positively Defined).

Для невырожденной $\mathbf{A}$ симметризованная матрица $\mathbf{A}^T\mathbf{A}$ является положительно определенной, и тогда для евклидовой и $\mathbf{A}$ -нормы справедливо соотношение

$$\begin{aligned}\|x - x_*\|_{\mathbf{A}^T\mathbf{A}}^2 &= (\mathbf{A}^T\mathbf{A}(x - x_*), (x - x_*)) = \\ &= (\mathbf{A}(x - x_*), \mathbf{A}(x - x_*)) = (r_x, r_x) = \|r_x\|_2^2.\end{aligned}\quad (4)$$

Собственные значения матрицы $\mathbf{A}$ (которые, вообще говоря, являются комплексными) будем обозначать $\lambda_j(\mathbf{A})$. Максимальный из всех модулей $|\lambda_j(\mathbf{A})|$ есть спектральный радиус матрицы.

Вектор e_k будем обозначать k -й единичный орт:

$$e_k = (0, \dots, 0, 1_k, 0, \dots, 0)^T.$$

Глава 1

Хранение и обработка разреженных матриц

1.1. Форматы хранения

Определение 1.1.1 Портретом разреженной матрицы A называется множество пар индексов (i, j) , таких что $a_{ij} \neq 0$:

$$P_A \stackrel{\text{def}}{=} \{(i, j) \mid a_{ij} \neq 0\}.$$

Можно выделить следующие три случая:

1. матрицы с несимметричным портретом, т.е. матрицы, у которых

$$\exists (i, j) \in P_A : (j, i) \notin P_A.$$

Такие матрицы несимметричны и в обычном смысле $A \neq A^T$;

2. несимметричные матрицы с симметричным портретом, у которых

$$(i, j) \in P_A \iff (j, i) \in P_A, \quad (1.1)$$

хотя в общем случае $a_{ij} \neq a_{ji}$;

3. симметричные матрицы, у которых $a_{ij} = a_{ji}$ (очевидно, (1.1) для них выполняется).

Разреженную матрицу можно представить графом со множеством вершин $\{1, 2, \dots, n\}$, тогда P_A будет являться множеством ребер этого графа. В случае симметричного портрета такой граф будет неориентированным, в случае несимметричного — ориентированным.

Очевидно, что для симметричных матриц достаточно хранить только один из треугольников (верхний или нижний). Для несимметричных матриц с симметричным портретом необходимо хранение всех ненулевых элементов, однако схему их размещения опять же достаточно задать лишь для одного из треугольников. Следует также принимать во внимание тот факт, что генерируемые в МКО и МКЭ глобальные матрицы СЛАУ всегда имеют ненулевую главную диагональ.

Распространенным способом хранения несимметричных матриц произвольной структуры является CSR¹ [6]. В нем разреженная матрица A хранится с использованием следующих массивов:

- `aelem`, который содержит все ненулевые элементы матрицы A , перечисленные в строчном порядке;
- `jptr`, который содержит столько же элементов, сколько `aelem` и для каждого из них указывает, в каком столбце находится данный элемент;
- `iptr`, который хранит число элементов, равное увеличенной на единицу размерности СЛАУ. Его i -й элемент указывает, с какой позиции в массивах `aelem` и `jptr` начинается i -я строка матрицы. Соответственно `iptr[i+1]-iptr[i]` равно числу ненулевых элементов в i -й строке. Последний элемент `iptr[n+1]` равен числу элементов в `aelem`, увеличенному на единицу.²

ПРИМЕР. Проиллюстрируем сказанное на следующей разреженной матрице:

$$A = \begin{pmatrix} 9 & 0 & 0 & 3 & 1 & 0 & 1 \\ 0 & 11 & 2 & 1 & 0 & 0 & 2 \\ 0 & 1 & 10 & 2 & 0 & 0 & 0 \\ 2 & 1 & 2 & 9 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 12 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 8 & 0 \\ 2 & 2 & 0 & 0 & 3 & 0 & 8 \end{pmatrix}. \quad (1.2)$$

Для нее $n = 7$ и массивы имеют вид:

$$\begin{aligned} \text{aelem} &= [\underbrace{9, 3, 1, 1}_{4(1)}, \underbrace{11, 2, 1, 2, 1, 10, 2, 2, 1, 2, 9, 1, 1, 1, 12, 1, 8}_{1(21)}, \underbrace{2, 2, 3, 8}_{4(22)}] \\ \text{jptr} &= [\underbrace{1, 4, 5, 7}_{4(1)}, \underbrace{2, 3, 4, 7}_{4(5)}, \underbrace{2, 3, 4}_{3(9)}, \underbrace{1, 2, 3, 4, 5}_{5(12)}, \underbrace{1, 4, 5, 7}_{4(17)}, \underbrace{6}_{1(21)}, \underbrace{1, 2, 5, 7}_{4(22)}] \\ \text{iptr} &= [1, 5, 9, 12, 17, 21, 22, 26] \end{aligned}$$

¹Compressed Sparse Row.

²Поскольку всегда `iptr[1]=1`, его можно и не хранить. В этом случае `iptr[i]` указывает на начало не i -й, а $(i + 1)$ -й строки.

(фигурными скобками сгруппированы элементы, принадлежащие одной и той же строке матрицы. Под скобками указано число сгруппированных элементов и порядковый номер первого из них в массиве).

Замечание 1. Возможно использование модификации CSR, в которой данные хранятся тем же образом и в тех же массивах, однако ненулевые элементы матрицы перечисляются не по строкам, а по столбцам. Данный вариант известен как CSC.³

Замечание 2. Элементы, принадлежащие одной строке, могут храниться как с упорядочиванием по столбцовому индексу, так и без упорядочивания. Это не влияет на алгоритм матрично-векторного умножения из §1.2, однако как будет показано уже в §1.3, в определенных условиях упорядочивание способно дать выигрыш.

Для хранения матриц с симметричным портретом и ненулевой главной диагональю может быть применена та же CSR-схема, со следующими изменениями⁴:

- элементы главной диагонали хранятся отдельно в массиве `adiag`;
- вместо массива `aelem` используется массив `altr`, в котором таким же образом хранятся *только поддиагональные* элементы матрицы. Для него формируются массивы `jptr` и `iptr`;
- для *наддиагональных* элементов матрицы формируется массив `autr`, причем он хранит элементы не в строчном, а в *столбцовом* порядке. Для `autr` в силу симметрии портрета массивы `jptr` и `iptr` не формируются.

Для матрицы (1.2) указанные массивы должны выглядеть следующим образом:

$$\begin{aligned}
 \text{adiag} &= [9, 11, 10, 9, 12, 8, 8] \\
 \text{altr} &= [1, \underbrace{2, 1, 2}, \underbrace{1, 1}, \underbrace{2, 2, 3}] \\
 \text{autr} &= [2, \underbrace{3, 1, 2}, \underbrace{1, 1}, \underbrace{1, 2, 1}] \\
 \text{jptr} &= [2, \underbrace{1, 2, 3}, \underbrace{1, 4}, \underbrace{1, 2, 5}] \\
 \text{iptr} &= [1, 1, 1, 2, 5, 7, 7, 10]
 \end{aligned}$$

(Как и прежде, фигурными скобками сгруппированы элементы одной строки.)

³Compressed Sparse Column.

⁴Такая схема иногда обозначается CSIR — Compressed Sparse (lower triangle) Row. Встречается также термин «Skyline Format» [12].

В случае, когда матрица симметрична, массивы altr и autr для нее совпадают, поэтому достаточно хранить только один из них.

1.2. Матрично-векторное умножение

Умножение разреженной матрицы на вектор $z := Ax$ наиболее просто реализуется для формата CSR. Поскольку матрица хранится по строкам, производится последовательный перебор элементов массива aelem, которые умножаются на коэффициент вектора x с индексом, взятым из соответствующей позиции массива jptr; результат добавляется к коэффициенту вектора z , соответствующему текущей сканируемой строке:

```

Входные данные: x; aelem, jptr, iptr; n
Результат: z=Ax
для i от 1 до n {
    z[i]:=0
    для j от iptr[i] до iptr[i+1]-1 {
        z[i]:=z[i]+x[jptr[j]]*aelem[j]
    }
}

```

При использовании отдельного хранения главной диагонали и треугольников схема умножения для нижнего треугольника остается той же. Для верхнего треугольника строчные и столбцовые индексы меняются местами, а элементы главной диагонали учитываются отдельно:

```

Входные данные: x; adiaq, altr, autr, jptr, iptr; n
Результат: z=Ax
для i от 1 до n { z[i]:=x[i]*adiaq[i] }
для i от 1 до n {
    для j от iptr[i] до iptr[i+1]-1 {
        z[i]:=z[i]+x[jptr[j]]*altr[j]
        z[jptr[j]]:=z[jptr[j]]+x[i]*autr[j]
    }
}

```

Если поменять местами обращения к массивам altr и autr, то вместо $z := Ax$ будет вычислено произведение $z := A^T x$.

Этот же алгоритм может быть применен и для случая симметричных матриц. Единственное необходимое изменение заключается в использовании не двух массивов altr и autr, а какого-нибудь одного из них.

1.3. Симметричность портрета и учет краевых условий

Известно, что при МКО и МКЭ-моделировании физических процессов краевые условия первого рода (условия Дирихле) учитываются путем обнуления внедиагональных элементов тех строк глобальной матрицы, которые соответствуют узлам сетки на границе, при этом диагональные элементы приравниваются к 1, а соответствующие элементы вектора правой части приравниваются к значениям, взятым из краевых условий.

Подобная процедура нарушает симметричность матрицы. Избежать этого можно за счет вычеркивания из СЛАУ соответствующей строки и столбца (поскольку значение решения в узле известно из краевого условия, ненулевые элементы вычеркиваемого столбца можно учесть в виде поправок к вектору правой части). Однако соответствующий учет изменяет размерность и портрет матрицы и реализуется достаточно сложно.

Сохранить симметричный портрет матрицы можно более простым способом, если ввести в него некоторое количество нулевых элементов, дополняющих портрет до симметричного. Легко видеть, что все позиции таких элементов принадлежат портрету P_A исходной матрицы до учета краевых условий.

Так, обнуление внедиагональных элементов третьей строки матрицы (1.2) приведет ее к несимметричной матрице $\hat{A}$, в которой позиции элементов, дополняющих портрет до симметричного, обведены:

$$\hat{A} = \begin{pmatrix} 9 & 0 & 0 & 3 & 1 & 0 & 1 \\ 0 & 11 & 2 & 1 & 0 & 0 & 2 \\ 0 & \boxed{0} & 1 & \boxed{0} & 0 & 0 & 0 \\ 2 & 1 & 2 & 9 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 12 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 8 & 0 \\ 2 & 2 & 0 & 0 & 3 & 0 & 8 \end{pmatrix}.$$

Для формата CSR обнуление внедиагональных элементов k -й строки матрицы описывается следующим простым циклом:

<pre> для i от iptr[k] до iptr[k+1]-1 { если jptr[i]=k то aelem[jptr[k]]:=1 иначе aelem[jptr[k]]:=0 } </pre>
--

Для случая отдельного хранения верхнего треугольника, нижнего треугольника и диагонали процедура несколько сложнее:

```

adiag[k]:=1
для i от iptr[k] до iptr[k+1]-1 {
    altr[jptr[k]]:=0
}
для i от 1 до n {
    найти такое j (iptr[i]<j<iptr[i+1]) что jptr[j]=k
    если найдено
        то    autr[j]:=0
}

```

Очевидно, если элементы матрицы внутри строк упорядочены по столбцовым индексам (см. замечание 2, §1.1), операция поиска может быть реализована более оптимально.

Поскольку на практике известно множество D индексов тех узлов сетки, которые лежат на границе области с заданными условиями Дирихле, эту процедуру можно обобщить:

```

для всех k из множества D {
    adia[k]:=1
    для i от iptr[k] до iptr[k+1]-1 {
        altr[i]:=0
        если jptr[i] ∈ D
            то    autr[i]:=0
    }
}

```

1.4. Прямой и обратный ход по разреженным треугольным матрицам

Хранение матрицы СЛАУ в формате CSIR оказывается удобным для тех случаев, когда используется предобусловливание с разложением на множители треугольной структуры, например LU-факторизация⁵ (см. §2.2).

Поскольку после выполнения факторизации⁶ треугольные матрицы L и U имеют те же портреты, что нижний и верхний треугольники матрицы A , для них достаточно хранить только сами элементы матриц и возможно (в зависимости от модификации LU) диагональные элементы; портреты будут полностью определяться массивами $iptr$ и $jptr$.

⁵Incomplete Lower-Upper triangles decompozition

⁶О программной реализации LU-декомпозиции см. также §2.4.

Пусть для матрицы A , заданной массивами $adiag$, $altr$, $autr$, $jptr$ и $iptr$, найдена факторизация $A \approx LU$, в которой L задается массивами $lptr$ и $ldiag$, а матрица U — массивами $uutr$ и $udiag$.

Тогда, если дан некоторый вектор f , прямой ход для системы $Lz = f$ может быть выполнен следующим образом:

```

Входные данные: f; ldiag, lptr, jptr, iptr; n
Результат: z=L-1f
Побочные эффекты: изменяется массив f
для i от 1 до n {
    для j от iptr[i] до iptr[i+1]-1 {
        f[i]:=f[i]-z[jptr[j]]*lptr[j]
    }
    z[i]:=f[i]/ldiag[i]
}

```

Как видно, классическая процедура прямого хода реализуется без каких-либо существенных изменений, лишь с поправкой на специальный способ хранения матрицы.

Массив $uutr$, в отличие от $lptr$, хранит элементы матрицы U не в строчном, а в столбцовом порядке. Поэтому обратный ход для системы $Uz = f$ должен быть организован по-другому; соответствующая процедура имеет вид

```

Входные данные: f; udiag, uutr, jptr, iptr; n
Результат: z=U-1f
Побочные эффекты: изменяется массив f
для i от n до 1 с шагом -1 {
    z[i]:=f[i]/udiag[i]
    для j от iptr[i] до iptr[i+1]-1 {
        f[jptr[j]]:=f[jptr[j]]-z[i]*uutr[j]
    }
}

```

Глава 2

Предобусловливание. Неполное LU-разложение

2.1. Предобусловливание

Пусть M — некоторая невырожденная матрица размерности n . Домножив (1) на M^{-1} , получим систему

$$M^{-1}Ax = M^{-1}b, \quad (2.1)$$

которая в силу невырожденности M имеет то же точное решение x_* . Введя обозначения $\hat{A} = M^{-1}A$ и $\hat{b} = M^{-1}b$, запишем (2.1) в виде

$$\hat{A}x = \hat{b}. \quad (2.2)$$

Хотя (2.2) алгебраически эквивалентна (1), спектральные характеристики матрицы $\hat{A}$ отличаются от характеристик матрицы A , что, вообще говоря, ведет к изменению скорости сходимости численных методов для (2.2) по отношению к (1) в конечной арифметике.

Процесс перехода от (1) к (2.2) с целью улучшения характеристик матрицы для ускорения сходимости к решению называется *предобусловливанием*¹, а матрица M — *матрицей предобусловливателя*.

Из (2.1) сразу же вытекает важное требование: *матрица M должна быть близка к матрице A* . (Выбор $M = A$ сразу же приводит (1) к виду $Ix = A^{-1}b$, однако не имеет практического смысла, так как требует нахождения A^{-1} , что, по существу и сводится к решению (1)). Вторым естественным требованием является требование легкой вычислимости матрицы M .

Нахождение произведения $M^{-1}A$ для вычисления $\hat{A}$ в общем случае ведет к изменению портрета ($P_A \neq P_{\hat{A}}$). Поэтому на практике используется другой подход.

¹Иногда *переобусловливанием*.

Невязка $\hat{r}$ системы (2.2) связана с невязкой r системы (1) очевидным соотношением

$$M\hat{r} = r, \quad (2.3)$$

которое справедливо и для матрично-векторных произведений $z = \mathbf{A}q$ и $\hat{z} = \hat{\mathbf{A}}q$:

$$\hat{z} = \hat{\mathbf{A}}q = M^{-1}\mathbf{A}q \implies M\hat{z} = \mathbf{A}q = z. \quad (2.4)$$

Это позволяет вместо явного перехода от (1) к (2.2) вводить в схемы методов корректирующие шаги для учета влияния предобуславливающей матрицы². Пример такого подхода будет приведен в §5.2.

Из (2.4) следует еще одно условие: структура матрицы предобуславливателя должна допускать легкое и быстрое решение «обратных к предобуславливателю» систем вида $M\hat{z} = z$.

Таким образом:

- M должна быть по возможности близка к $\mathbf{A}$;
- M должна быть легко вычислима;
- M должна быть легко обратима.

Замечание. Выбор в качестве M некоторой диагональной матрицы удовлетворяет двум последним требованиям из трех перечисленных и, по существу, сводит предобуславливание к масштабированию СЛАУ. Как известно, в ряде случаев масштабирование способно значительно ускорить процесс решения.

Описанное выше предобуславливание иногда называется *левым*, так как домножение матрицы СЛАУ на матрицу предобуславливателя производится слева. Другой возможный подход основан на переходе от (1) к системе

$$\mathbf{A}M^{-1}y = b, \quad (2.5)$$

у которой точное решение y_* связано с точным решением x_* исходной СЛАУ соотношением

$$x_* = M^{-1}y_*. \quad (2.6)$$

Предобуславливание (2.5) реализуется путем выполнения вместо матрично-векторных умножений $z := \mathbf{A}q$ двойных умножений $z := \mathbf{A}(M^{-1}q)$; кроме того, при достижении требуемой точности осуществляется пересчет решения в соответствии с (2.6).

Подобная схема предобуславливания называется *правой*.

²В некоторых случаях, когда матрица M имеет простую структуру (например, диагональную — см. ниже замечание), переход от $\mathbf{A}$ и b к $\hat{\mathbf{A}}$ и $\hat{b}$ может выполняться и явно. Подобное предобуславливание, чтобы подчеркнуть что оно выполняется вне метода, иногда называют *внешним*.

- матрицы $\mathbf{L}$ и $\mathbf{U}$ являются нижнетреугольной и верхнетреугольной соответственно;
- $P_{\mathbf{L}} \subset P_{\mathbf{A}}$ и $P_{\mathbf{U}} \subset P_{\mathbf{A}}$;
- $\forall (i, j) \in P_{\mathbf{A}} : [\mathbf{LU}]_{ij} = [\mathbf{A}]_{ij}$;
- $P_{\mathbf{A}} \cap P_{\mathbf{R}} = \emptyset$.

Тогда приближенное представление $\mathbf{A} \approx \mathbf{LU}$ называется *неполной LU-факторизацией матрицы $\mathbf{A}$* или коротко ее *LU-разложением*³.

(Последние два требования, по существу, означают, что на множестве индексов $P_{\mathbf{A}}$ матричное произведение $\mathbf{LU}$ должно точно воспроизводить элементы $\mathbf{A}$.)

Для нахождения матриц $\mathbf{L}$ и $\mathbf{U}$ будем генерировать их построчно. Предположим, что первые $(k - 1)$ строк уже найдены и необходимо найти k -ю. Запишем в блочном виде первые k строк разложения (2.10):

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{a}_{21}^T & \mathbf{a}_{22}^T \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11} & \mathbf{0} \\ \mathbf{l}_{21}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ \mathbf{0} & \mathbf{u}_{22}^T \end{pmatrix} + \begin{pmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} \\ \mathbf{r}_{21}^T & \mathbf{r}_{22}^T \end{pmatrix}, \quad (2.11)$$

где $\mathbf{l}_{21}$, $\mathbf{u}_{22}$, $\mathbf{r}_{21}$ и $\mathbf{r}_{22}$ — некоторые векторы. Выполнив действия над матрицами в правой части (2.11), получим

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{a}_{21}^T & \mathbf{a}_{22}^T \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11}\mathbf{U}_{11} + \mathbf{R}_{11} & \mathbf{L}_{11}\mathbf{U}_{12} + \mathbf{R}_{12} \\ \mathbf{l}_{21}^T\mathbf{U}_{11} + \mathbf{r}_{21}^T & \mathbf{l}_{21}^T\mathbf{U}_{12} + \mathbf{u}_{22}^T + \mathbf{r}_{22}^T \end{pmatrix}.$$

Из равенства матриц следует, что искомые векторы $\mathbf{l}_{21}$ и $\mathbf{u}_{22}$ должны удовлетворять условиям:

$$\mathbf{l}_{21}^T\mathbf{U}_{11} + \mathbf{r}_{21}^T = \mathbf{a}_{21}^T; \quad (2.12)$$

$$\mathbf{u}_{22}^T + \mathbf{r}_{22}^T = \mathbf{a}_{22}^T - \mathbf{l}_{21}^T\mathbf{U}_{12}. \quad (2.13)$$

Решив эти системы, можно найти коэффициенты k -х строк матриц разложения $l_{k1}, \dots, l_{k,k-1}, u_{kk}, \dots, u_{kn}$ ⁴.

Определим l_{kj} из (2.12) в предположении, что $l_{k1} \dots l_{k,j-1}$ уже найдены⁵. Согласно ранее сформулированным условиям, если $a_{kj} = 0$,

³Чтобы подчеркнуть тот факт, что в данной постановке задачи в портрет не вводятся новые ненулевые элементы, такую факторизацию иногда называют факторизацией с нулевым заполнением.

⁴Запись (2.11) подразумевает, что $l_{kk} \equiv 1$.

⁵Поскольку матрица $\mathbf{L}$ нижнетреугольна, для ее искомого коэффициента всегда $j < k$.

то сразу $l_{kj} = 0$. В противном же случае $r_{kj} = 0$ и (2.12) можно записать в виде⁶

$$\sum_{i=1}^j l_{ki}u_{ij} = \sum_{i=1}^{j-1} l_{ki}u_{ij} + l_{kj}u_{jj} = a_{kj}. \quad (2.14)$$

Это позволяет вычислить l_{kj} следующим образом⁷:

$$l_{kj} = \frac{1}{u_{jj}} \left(a_{kj} - \sum_{i=1}^{j-1} l_{ki}u_{ij} \right). \quad (2.15)$$

Аналогичными рассуждениями с учетом $l_{jj} \equiv 1$ из (2.13) можно получить выражение для u_{kj} ⁸:

$$u_{kj} = a_{kj} - \sum_{i=1}^{k-1} l_{ki}u_{ij} \quad (2.16)$$

(для тех случаев, когда $a_{kj} \neq 0$, иначе сразу $u_{kj} = 0$).

На основании формул (2.15) и (2.16) можно записать следующий алгоритм LU-разложения:

Для $k = \overline{1, n}$
 Для $j = \overline{1, k-1}$
 Если $(k, j) \notin P_A$
 то $l_{kj} := 0$
 иначе $l_{kj} := \frac{1}{u_{jj}} \left(a_{kj} - \sum_{i=1}^{j-1} l_{ki}u_{ij} \right)$
 увеличить j
 $l_{kk} := 1$
 Для $j = \overline{k, n}$
 Если $(k, j) \notin P_A$
 то $u_{kj} := 0$
 иначе $u_{kj} := a_{kj} - \sum_{i=1}^{k-1} l_{ki}u_{ij}$
 увеличить j
 увеличить k

⁶С учетом того, что для $i > j$ элементы верхнетреугольной матрицы U равны нулю.

⁷Напомним, что строки матриц L и U с 1-й по k -ю предполагаются уже построенными. Так как $j < k$, то все коэффициенты матрицы U , присутствующие в (2.14), уже определены.

⁸Теперь для $i > k$ элементы нижнетреугольной матрицы L равны нулю.

ПРИМЕР. Для матрицы (1.2) приведенный алгоритм дает следующие матрицы L и U (с точностью три знака):

$$L = \begin{pmatrix} 1 & & & & & & & \\ 0 & 1 & & & & & & \\ 0 & 0.091 & 1 & & & & & \\ 0.222 & 0.091 & 0.185 & 1 & & & & \\ 0.111 & 0 & 0 & 0.085 & 1 & & & \\ 0 & 0 & 0 & 0 & 0 & 1 & & \\ 0.222 & 0.182 & 0 & 0 & 0.235 & 0 & 1 & \end{pmatrix}; \quad (2.17)$$

$$U = \begin{pmatrix} 9 & 0 & 0 & 3 & 1 & 0 & 1 & \\ & 11 & 2 & 1 & 0 & 0 & 2 & \\ & & 9.818 & 1.909 & 0 & 0 & 0 & \\ & & & 7.889 & 0.778 & 0 & 0 & \\ & & & & 11.823 & 0 & 0.889 & \\ & & & & & 8 & 0 & \\ & & & & & & & 7.205 \end{pmatrix}. \quad (2.18)$$

Соответствующая им матрица ошибки разложения R равна

$$R = A - LU = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.182 & \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.404 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & -0.364 & -0.848 & 0 & 0 & 0 & \end{pmatrix}.$$

Замечание. Элементы матриц L и U , полученных в результате LU -разложения, не совпадают с соответствующими элементами матриц L_A и U_A , полученных при полной LU -факторизации. В этом легко убедиться, сравнивая (2.8) и (2.9) с (2.17) и (2.18).

Очевидно, что матрица LU удовлетворяет всем трем требованиям к матрице предобусловливателя, сформулированным в §2.1. Действительно, она приближает матрицу A , так как на множестве индексов P_A точно воспроизводит ее; она легко вычисляется по приведенному выше несложному алгоритму; наконец, она легко обратима, так как является произведением двух треугольных матриц.

Таким образом, выбор $M = LU$ является достаточно хорошим способом предобусловливания.

2.3. Симметричный случай

В случае, когда матрица A симметрична, из равенств $A = LU$ и $A = A^T$ для треугольных матриц L и U следует, что $L = U^T$. Фактори-

зация (2.10) в этом случае принимает вид

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T + \mathbf{R} = \mathbf{U}^T\mathbf{U} + \mathbf{R}. \quad (2.19)$$

Очевидно, однако, что для нахождения диагональных элементов такой матрицы $\mathbf{L}$ требуются операции извлечения квадратного корня. Чтобы избежать этого, вместо факторизации (2.19) используется несколько отличающийся вариант

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T + \mathbf{R}, \quad (2.20)$$

где главная диагональ $\mathbf{L}$ состоит из единичных элементов, а $\mathbf{D}$ — диагональная матрица.

Для нахождения коэффициентов $\mathbf{L}$ и $\mathbf{D}$ применяется такой же подход, как и в §2.2, поэтому сразу приведем алгоритм без пояснений:

<p>Для $k = \overline{1, n}$ Для $j = \overline{1, k-1}$ Если $(k, j) \notin P_{\mathbf{A}}$ то $l_{kj} := 0$ иначе $l_{kj} := \frac{1}{d_j} \left(a_{kj} - \sum_{i=1}^{j-1} d_i l_{ik} l_{jk} \right)$ увеличить j $d_k := a_{kk} - \sum_{i=1}^{k-1} l_{ki}^2 d_i^2$ увеличить k</p>

ПРИМЕР. Для симметричной матрицы

$$\mathbf{A} = \begin{pmatrix} 9 & 0 & 3 & 0 \\ 0 & 8 & 0 & 1 \\ 3 & 0 & 11 & 1 \\ 0 & 1 & 1 & 9 \end{pmatrix}$$

приведенный алгоритм дает (с точностью три знака):

$$\mathbf{L} = \begin{pmatrix} 1 & & & \\ 0 & 1 & & \\ 0.333 & 0 & 1 & \\ 0 & 0.125 & 0.091 & 1 \end{pmatrix}; \quad \mathbf{D} = \text{diag}(9, 8, 11, 8.784).$$

Соответствующая им матрица ошибки разложения $\mathbf{R} = \mathbf{A} - \mathbf{L}\mathbf{D}\mathbf{L}^T$ равна нулю.

2.4. О программной реализации LU-предобусловливания

Одним из условий при построении LU-разложения было соответствие портретов $P_L \subset P_A$ и $P_U \subset P_A$. При этом одна из матриц является верхне-, а другая нижнетреугольной, так что их портреты не совпадают, за исключением диагональных элементов. Однако алгоритм разложения построен так, что $l_{kk} \equiv 1$ и, следовательно, явно хранить диагональные элементы L необходимости нет.

Таким образом, $P_{L+U} \subset P_A$ и разложение лучше всего хранить как еще одну матрицу с тем же портретом что и A — вследствие совпадения портретных массивов они хранятся только один раз, а к массиву элементов `aelem` (для CSIR — `altr`, `autr`, `adiag`) добавляется массив `luelem` (для CSIR — массивы `ltr`, `utr`, `luddiag`). Это эквивалентно хранению объединенной матрицы разложения

$$B = L + U - I,$$

диагональ которой считается отнесенной к матрице U .

Для нахождения LU-факторизации формат хранения CSIR оказывается более удобным по сравнению с CSR. Как можно видеть из §2.2, в алгоритме для вычисления коэффициентов l_{kj} и u_{kj} используются скалярные произведения «строка-столбец» вида

$$\sum_{i=1}^{\max\{i,k\}-1} l_{ki} u_{ij}, \quad (2.21)$$

в которых доступ к элементам нижнетреугольной матрицы L производится по строкам, а к элементам верхнетреугольной матрицы U — по столбцам. Для CSIR порядок доступа совпадает с порядком хранения для обеих матриц, тогда как для CSR — только для матрицы L . Для постолбцового доступа к элементам U необходимы операции поиска, что существенно замедляет расчеты⁹.

При использовании разреженных строчных форматов внешние циклы по перебору индексов элементов матрицы в алгоритме из §2.2, очевидно, должны быть переписаны. Построчный доступ к элементам матрицы L выполняется непосредственно, для построчного доступа к элементам U необходимы дополнительные операции

⁹С другой стороны, как будет показано ниже, из-за построчной генерации матриц операции поиска необходимы и при использовании CSIR, однако в этом случае на каждую сумму вида (2.21) нужен только один поиск, тогда как для CSR операция поиска должна производиться для каждого слагаемого суммы. Можно показать, что вычислительная сложность для CSR на порядок выше, чем для CSIR.

поиска. Общая схема нахождения LU -факторизации выглядит следующим образом¹⁰:

```
для k от 1 до n {
    для j от iptr[k] до iptr[k+1]-1 {
        вычислить ltr[j] по формуле для  $l_{k,jptr[j]}$ 
    }
    вычислить ludiag[k] по формуле для  $u_{k,k}$ 
    для j от k+1 до n {
        найти такое q ( $iptr[j] \leq q < iptr[j+1]$ ) что  $jptr[q]=k$ 
        если найдено
            то вычислить utr[q] по формуле для  $u_{k,j}$ 
    }
}
```

Для решения треугольных СЛАУ с матрицами L и U (см. формулу (2.4)) могут быть использованы алгоритмы из §1.4.

¹⁰Снова (ср. §1.3) упорядоченность элементов внутри строки способна дать некоторый выигрыш при поиске.

Глава 3

Классические итерационные методы и релаксация

Исторически первые итерационные методы основывались на циклическом *покомпонентном* изменении вектора решения, осуществляемом таким образом, чтобы обнулить соответствующий коэффициент вектора невязки и тем самым уменьшить его норму. Подобная методика уточнения решения получила название *релаксации*.

Хотя в настоящее время такие методы в их классической формулировке уже практически не применяются, существуют определенные классы задач [4], для которых разработаны их модификации, хорошо себя зарекомендовавшие. Кроме того, как будет показано в §3.3, эти методы могут быть применены (и применяются) не в качестве самостоятельного средства решения СЛАУ, а для предобуславливания¹.

3.1. Методы Якоби и Гаусса-Зейделя

Пусть матрица A системы (1) такова, что ее главная диагональ не содержит нулевых элементов. Представим ее в виде разности трех матриц:

$$A = D - E - F, \quad (3.1)$$

где матрица D содержит только диагональные элементы A ; матрица E — только поддиагональные; матрица F — только наддиагональные (см. рис. 3.1)

Система (1) тогда может быть записана в виде

$$Dx - Ex - Fx = b.$$

¹См. также §4.4, где идея методов, основанных на расщеплении, используется для построения подпространств Крылова.

что в векторной форме эквивалентно

$$x_{k+1} = (\mathbf{D} - \mathbf{E})^{-1}\mathbf{F}x_k + (\mathbf{D} - \mathbf{E})^{-1}b. \quad (3.6)$$

Такая схема называется методом Гаусса-Зейделя.

Выражение (3.5) получается из (3.2) заменой x_k на x_{k+1} при матрицах $\mathbf{D}$ и $\mathbf{E}$. Если вместо этой пары матриц взять $\mathbf{D}$ и $\mathbf{F}$, то получится похожая схема

$$x_{k+1} = (\mathbf{D} - \mathbf{F})^{-1}\mathbf{E}x_k + (\mathbf{D} - \mathbf{F})^{-1}b, \quad (3.7)$$

которая называется обратным методом Гаусса-Зейделя.

Еще одной очевидной модификацией является симметричный метод Гаусса-Зейделя, который заключается в циклическом чередовании (3.6) и (3.7) на соседних итерациях.

Нетрудно заметить, что выражения (3.3), (3.6), (3.7) могут быть записаны в виде

$$\mathbf{K}x_{k+1} = \mathbf{R}x_k + b, \quad (3.8)$$

где матрицы $\mathbf{K}$ и $\mathbf{R}$ связаны соотношением

$$\mathbf{A} = \mathbf{K} - \mathbf{R}. \quad (3.9)$$

Подобное представление матрицы $\mathbf{A}$ называется *расщеплением*, а методы вида (3.8) — *методами, основанными на расщеплении*. Очевидно, матрица $\mathbf{K}$ должна быть невырожденной и легко обратимой.

Условия сходимости методов (3.8) устанавливает следующая теорема:

Теорема 3.1.1 *Вычислительная схема (3.8) сходится при любом начальном приближении x_0 тогда и только тогда, когда матрицы $\mathbf{K}$ и $\mathbf{R}$ удовлетворяют условию $\max_j |\lambda_j(\mathbf{K}^{-1}\mathbf{R})| < 1$.*

Доказательство. В силу соотношения (3.9) для точного решения x_* системы (1) справедливо

$$\mathbf{K}x_* = \mathbf{R}x_* + b.$$

Почленно вычитая из этого равенства (3.8), получим

$$\mathbf{K}(x_* - x_{k+1}) = \mathbf{R}(x_* - x_k), \quad (3.10)$$

где векторы в левой и правой частях есть ошибки решения ϑ_{k+1} и ϑ_k соответственно.

Очевидно, сходимость схемы (3.8) к точному решению эквивалентна сходимости $\vartheta_k \rightarrow 0$ при $k \rightarrow \infty$. Из (3.10) имеем

$$\vartheta_{k+1} = \mathbf{K}^{-1}\mathbf{R}\vartheta_k = (\mathbf{K}^{-1}\mathbf{R})^{k+1}\vartheta_0. \quad (3.11)$$

Следовательно, для сходимости (3.8) необходимо и достаточно выполнения условия $(\mathbf{K}^{-1}\mathbf{R})^k \rightarrow 0$ при $k \rightarrow \infty$, откуда и вытекает утверждение теоремы. $\square$

Замечание 1. Из (3.11) видно, что чем меньше присутствующая в условии величина $\max_j |\lambda_j(\mathbf{K}^{-1}\mathbf{R})|$, тем быстрее сходимость метода.

Замечание 2. Матрицы $\mathbf{K}$ и $\mathbf{R}$ в (3.8) могут, вообще говоря, зависеть от номера итерации ($\mathbf{K} = \mathbf{K}_k$, $\mathbf{R} = \mathbf{R}_k$)³. В этом случае условия $\max_j |\lambda_j(\mathbf{K}_k^{-1}\mathbf{R}_k)| < 1$ достаточно для гарантированной сходимости, однако оно перестает быть необходимым.

3.2. Ускорение сходимости релаксационных методов

Из теоремы 3.1.1 следует, что скорость сходимости методов, основанных на расщеплении, непосредственно связана со спектральным радиусом матрицы $\mathbf{K}^{-1}\mathbf{R}$; с другой стороны, выбор $\mathbf{K}$ ограничен требованием легкой обратимости.

Одним из распространенных способов улучшения сходимости является введение параметра. Пусть ω — некоторое вещественное число. Рассмотрим вместо (1) масштабированную систему

$$\omega\mathbf{A}x = \omega b, \quad (3.12)$$

и вместо (3.1) воспользуемся представлением

$$\omega\mathbf{A} = (\mathbf{D} - \omega\mathbf{E}) - (\omega\mathbf{F} + (1 - \omega)\mathbf{D}), \quad (3.13)$$

где матрицы $\mathbf{D}$, $\mathbf{E}$ и $\mathbf{F}$ имеют тот же смысл, что и в (3.1).

Тогда на основании (3.12) и (3.13) можно построить итерационную схему, похожую на метод Гаусса-Зейделя,

$$(\mathbf{D} - \omega\mathbf{E})x_{k+1} = [\omega\mathbf{F} + (1 - \omega)\mathbf{D}]x_k + \omega b. \quad (3.14)$$

Схема (3.14) называется *методом последовательной верхней релаксации* (SOR⁴) Для нее

$$\begin{aligned} \mathbf{K}_{\text{SOR}}(\omega) &= \mathbf{D} - \omega\mathbf{E}; \\ \mathbf{R}_{\text{SOR}}(\omega) &= \omega\mathbf{F} + (1 - \omega)\mathbf{D}. \end{aligned}$$

Выбор параметра ω , минимизирующего спектральный радиус $\rho(\omega) = \max_j |\lambda_j(\mathbf{K}_{\text{SOR}}^{-1}(\omega)\mathbf{R}_{\text{SOR}}(\omega))|$ является, вообще говоря, достаточно сложной проблемой. Однако для многих классов матриц (например, связанных с конечно-разностными аппроксимациями уравнений в частных производных [4, 5]) такая задача исследована и оптимальные значения ω известны.

³Такие итерационные схемы называются *нестационарными*.

⁴Successive OverRelaxation.

Замечание. Записав (3.14) в скалярной форме, нетрудно получить, что в SOR решение удовлетворяет соотношению

$$x_i^{(k+1)} = \omega x_i^{\text{GS}} + (1 - \omega)x_i^{(k)} \quad i = \overline{1, n},$$

где x_i^{GS} — итерация метода Гаусса-Зейделя, определяемая формулой (3.5). Действительно, при $\omega = 1$ (3.13) совпадает с (3.1).

Выражение (3.13) остается тождеством, если в нем поменять местами матрицы $\mathbf{E}$ и $\mathbf{F}$. Такая перестановка дает обратный метод последовательной верхней релаксации:

$$(\mathbf{D} - \omega\mathbf{F})x_{k+1} = [\omega\mathbf{E} + (1 - \omega)\mathbf{D}]x_k + \omega b. \quad (3.15)$$

Последовательное применение прямого и обратного методов SOR дает симметричный метод последовательной верхней релаксации (SSOR⁵)

$$\begin{aligned} (\mathbf{D} - \omega\mathbf{E})x_{k+1/2} &= [\omega\mathbf{F} + (1 - \omega)\mathbf{D}]x_k + \omega b; \\ (\mathbf{D} - \omega\mathbf{F})x_{k+1} &= [\omega\mathbf{E} + (1 - \omega)\mathbf{D}]x_{k+1/2} + \omega b. \end{aligned}$$

3.3. Связь с предобуславливанием

Ранее было показано, что методы Якоби и Гаусса-Зейделя, а также SOR и SSOR могут быть представлены в виде (3.8). Другой формой этого выражения является

$$x_{k+1} = \mathbf{G}x_k + f, \quad (3.16)$$

которое получается из (3.8) домножением левой и правой частей на $\mathbf{K}^{-1}$.

Проведя аналогию с (3.1) и (3.2), нетрудно увидеть, что (3.16) можно рассматривать как итерационную схему, построенную для решения СЛАУ

$$(\mathbf{I} - \mathbf{G})x = f, \quad (3.17)$$

в которой

$$\begin{aligned} f &= \mathbf{K}^{-1}b; \\ \mathbf{G} &= \mathbf{K}^{-1}\mathbf{R} = \mathbf{K}^{-1}(\mathbf{K} - \mathbf{A}) = \mathbf{I} - \mathbf{K}^{-1}\mathbf{A}. \end{aligned}$$

Таким образом, система (3.17) эквивалентна системе

$$\mathbf{K}^{-1}\mathbf{A}x = \mathbf{K}^{-1}b,$$

⁵Symmetric SOR.

т.е. предобусловленной СЛАНУ (1) с матрицей предобусловливания $\mathbf{K}$. Для рассмотренных в данной главе методов эти матрицы равны⁶:

$$\begin{aligned}\mathbf{K}_J &= \mathbf{D}; \\ \mathbf{K}_{GS} &= \mathbf{D} - \mathbf{E}; \\ \mathbf{K}_{SOR} &= \frac{1}{\omega}(\mathbf{D} - \omega\mathbf{E}); \\ \mathbf{K}_{SSOR} &= \frac{1}{\omega(2-\omega)}(\mathbf{D} - \omega\mathbf{E})\mathbf{D}^{-1}(\mathbf{D} - \omega\mathbf{F}).\end{aligned}$$

Скалярные множители $\frac{1}{\omega}$ и $\frac{1}{\omega(2-\omega)}$ при практических реализациях SOR и SSOR-предобусловливания обычно не учитываются, так как дают лишь общее масштабирование, практически не влияющее на скорость сходимости.

⁶Выражения для $\mathbf{K}_{GS}$ и $\mathbf{K}_{SOR}$ приведены для прямых методов. Для обратных методов матрицы $\mathbf{E}$ и $\mathbf{F}$ меняются местами.

Глава 4

Проекционные методы. Подпространства Крылова

4.1. Общий подход к построению проекционных методов

Рассмотрим систему $Ax = b$ и сформулируем для нее следующую задачу. Пусть заданы некоторые два подпространства $\mathcal{K} \subset \mathcal{R}^n$ и $\mathcal{L} \subset \mathcal{R}^n$. Требуется найти такой вектор $x \in \mathcal{K}$, который обеспечивал бы решение исходной системы, «оптимальное относительно подпространства $\mathcal{L}$ », т.е. чтобы выполнялось условие

$$\forall l \in \mathcal{L} : (Ax, l) = (b, l) ,$$

называемое условием Петрова-Галеркина. Сгруппировав обе части равенства по свойствам скалярного произведения и заметив что $b - Ax = r_x$, это условие можно переписать в виде

$$\forall l \in \mathcal{L} : (r_x, l) = 0 , \tag{4.1}$$

т.е. $r_x = b - Ax \perp \mathcal{L}$. Такая задача называется задачей проектирования решения x на подпространство $\mathcal{K}$ ортогонально к подпространству $\mathcal{L}$.

В более общей постановке задача выглядит следующим образом. Для исходной системы (1) известно некоторое приближение x_0 к решению x_* . Требуется уточнить его поправкой $\delta_x \in \mathcal{K}$ таким образом, чтобы $b - A(x_0 + \delta_x) \perp \mathcal{L}$. Условие Петрова-Галеркина в этом случае можно записать в виде

$$\forall l \in \mathcal{L} : (r_{x_0 + \delta_x}, l) = ((b - Ax_0) - A\delta_x, l) = (r_0 - A\delta_x, l) = 0 .$$

Пусть $\dim \mathcal{K} = \dim \mathcal{L} = m$. Введем в подпространствах $\mathcal{K}$ и $\mathcal{L}$ базисы $\{v_j\}_{j=1}^m$ и $\{w_j\}_{j=1}^m$ соответственно. Нетрудно видеть, что (4.1) вы-

полняется тогда и только тогда, когда

$$\forall j (1 \leq j \leq m) : (r_0 - \mathbf{A}\delta_x, w_j) = 0. \quad (4.2)$$

При введении для базисов матричных обозначений $\mathbf{V} = [v_1|v_2|\dots|v_m]$ и $\mathbf{W} = [w_1|w_2|\dots|w_m]$ можно записать $\delta_x = \mathbf{V}y$ где $y \in \mathcal{R}^m$ — вектор коэффициентов. Тогда (4.2) может быть записано в виде

$$\mathbf{W}^T(r_0 - \mathbf{A}\mathbf{V}y) = 0, \quad (4.3)$$

откуда $\mathbf{W}^T\mathbf{A}\mathbf{V}y = \mathbf{W}^Tr_0$ и

$$y = (\mathbf{W}^T\mathbf{A}\mathbf{V})^{-1} \mathbf{W}^Tr_0. \quad (4.4)$$

Таким образом, решение должно уточняться в соответствии с формулой

$$x_1 = x_0 + \mathbf{V}(\mathbf{W}^T\mathbf{A}\mathbf{V})^{-1} \mathbf{W}^Tr_0, \quad (4.5)$$

из которой сразу вытекает важное требование: в практических реализациях проекционных методов подпространства $\mathcal{K}$ и $\mathcal{L}$ и их базисы должны выбираться так, чтобы матрица $\mathbf{W}^T\mathbf{A}\mathbf{V}$ либо была малой размерности, либо имела простую структуру, удобную для обращения.

Из (4.3) также вытекает соотношение

$$\mathbf{V}y = \mathbf{A}^{-1}r_0 = \mathbf{A}^{-1}(b - \mathbf{A}x_0) = x_* - x_0,$$

т.е. $\mathbf{V}y = \delta_x$ представляет собой проекцию на подпространство $\mathcal{K}$ разности между точным решением и начальным приближением.

Пусть имеется набор пар подпространств $\langle \mathcal{K}_i, \mathcal{L}_i \rangle_{i=1}^q$ таких, что $\mathcal{K}_1 \oplus \mathcal{K}_2 \oplus \dots \oplus \mathcal{K}_q = \mathcal{R}^n$ и $\mathcal{L}_1 \oplus \mathcal{L}_2 \oplus \dots \oplus \mathcal{L}_q = \mathcal{R}^n$. Тогда очевидно, что последовательное применение описанного ранее процесса ко всем таким парам приведет к решению x_q , удовлетворяющему исходной системе $\mathbf{A}x = b$. Соответственно, в общем виде алгоритм любого метода проекционного класса может быть записан следующим образом:

Начало

Выбор пары подпространств $\mathcal{K}$ и $\mathcal{L}$

Построение для $\mathcal{K}$ и $\mathcal{L}$ базисов $\mathbf{V} = [v_1|v_2|\dots|v_m]$

и $\mathbf{W} = [w_1|w_2|\dots|w_m]$

$$r := b - \mathbf{A}x$$

$$y := (\mathbf{W}^T\mathbf{A}\mathbf{V})^{-1} \mathbf{W}^Tr$$

$$x := x + \mathbf{V}y$$

Продолжать до достижения сходимости

4.2. Случай одномерных подпространств $\mathcal{K}$ и $\mathcal{L}$

Наиболее простой ситуацией является случай, когда пространства $\mathcal{K}$ и $\mathcal{L}$ одномерны. Пусть $\mathcal{K} = \text{span}\{v\}$ и $\mathcal{L} = \text{span}\{w\}$. Тогда (4.5) принимает вид

$$x_{k+1} = x_k + \gamma_k v_k, \quad (4.6)$$

причем коэффициент γ_k легко находится из условия ортогональности $r_k - \mathbf{A}(\gamma_k v_k) \perp w_k$:

$$(r_k - \gamma_k \mathbf{A}v_k, w_k) = (r_k, w_k) - \gamma_k (\mathbf{A}v_k, w_k) = 0,$$

откуда $\gamma_k = (r_k, w_k) / (\mathbf{A}v_k, w_k)$.

ПРИМЕР 1. Выберем $v_k = w_k = r_k$. Тогда (4.6) примет вид

$$x_{k+1} = x_k + \frac{(r_k, r_k)}{(\mathbf{A}r_k, r_k)} r_k. \quad (4.7)$$

Поскольку выражение в знаменателе представляет собой квадратичную форму $r_k^T \mathbf{A} r_k$, сходимость процесса гарантирована, если матрица $\mathbf{A}$ является симметричной и положительно определенной.

Данный метод есть метод наискорейшего спуска¹; можно показать что на каждой итерации в нем минимизируется функционал $\Phi(x) = \|x - x_*\|_{\mathbf{A}}^2$ в направлении $-\nabla\Phi(x)$.

Действительно, в силу положительной определенности $\mathbf{A}$, квадратичная форма $(x - x_*)^T \mathbf{A} (x - x_*) = \Phi(x)$ достигает своего минимума (равного нулю) при $x = x_*$ и строго выпукла. При этом

$$\begin{aligned} \Phi(x) &= (\mathbf{A}(x - x_*), x - x_*) = \\ &= (\mathbf{A}x, x) - (\mathbf{A}x, x_*) - (b, x) + (b, x_*) = \\ &= x^T \mathbf{A} x - x_*^T \mathbf{A} x - b^T x + b^T x_*. \end{aligned}$$

В силу симметричности матрицы $\mathbf{A}$ справедливо $x_*^T \mathbf{A} x = b^T (\mathbf{A}^{-1}) \mathbf{A} x = b^T x$, и функционал равен

$$\Phi(x) = x^T \mathbf{A} x - 2b^T x + b^T x_*, \quad (4.8)$$

а его градиент $\nabla\Phi(x) = 2\mathbf{A}x - 2b = -2r_x$; следовательно, $-\nabla\Phi(x) \uparrow r_x$.

Покажем теперь, что выбор $\gamma = \|r_x\| / (r_x^T \mathbf{A} r_x)$ доставляет минимум $\Phi(x)$ в этом направлении. Подставим в (4.8) выражение $x + \gamma r$; последним слагаемым $b^T x_*$ при этом можно пренебречь, так как оно постоянно и не влияет на процесс минимизации. Снова будем учитывать симметрию матрицы $\mathbf{A}$

$$f(\gamma) = \Phi(x + \gamma r) = (x + \gamma r)^T \mathbf{A} (x + \gamma r) - 2b^T (x + \gamma r) =$$

¹Англ. Steepest Descent Method — SDM.

$$\begin{aligned}
&= x^T \mathbf{A}x + 2\gamma x^T \mathbf{A}r + \gamma^2 r^T \mathbf{A}r - 2b^T x - 2\gamma b^T r = \\
&= \left[x^T \mathbf{A}x - b^T x \right] - b^T x + 2\gamma \left[x^T \mathbf{A}r - b^T r \right] + \gamma^2 r^T \mathbf{A}r = \\
&= -(r+b)^T x - 2\gamma r^T r + \gamma^2 r^T \mathbf{A}r.
\end{aligned}$$

Найдем $\min_{\gamma} f(\gamma)$; учитывая выпуклость $\Phi(x)$, для этого достаточно найти стационарную точку $f(\gamma)$:

$$f'(\gamma) = 2\gamma r^T \mathbf{A}r - 2r^T r = 0 \implies \gamma = \frac{r^T r}{r^T \mathbf{A}r} = \frac{(r, r)}{(\mathbf{A}r, r)},$$

что совпадает с (4.7)

Замечание. В практических задачах метод наискорейшего спуска обладает достаточно медленной сходимостью, соответствующий анализ приведен в [4, 3].

ПРИМЕР 2. Выберем $v_k = \mathbf{A}^T r_k$ и $w_k = \mathbf{A}v_k$. Формула (4.6) принимает вид

$$\begin{aligned}
x_{k+1} &= x_k + \frac{(r_k, \mathbf{A}\mathbf{A}^T r_k)}{(\mathbf{A}\mathbf{A}^T r_k, \mathbf{A}\mathbf{A}^T r_k)} \mathbf{A}^T r_k = \\
&= x_k + \frac{(\mathbf{A}^T r_k, \mathbf{A}^T r_k)}{(\mathbf{A}^T \mathbf{A} \mathbf{A}^T r_k, \mathbf{A}^T r_k)} \mathbf{A}^T r_k. \tag{4.9}
\end{aligned}$$

Данный метод есть *метод наискорейшего уменьшения невязки*²; условием его сходимости является невырожденность матрицы $\mathbf{A}$.

Сравнивая (4.9) и (4.7), нетрудно убедиться что метод наискорейшего уменьшения невязки совпадает с методом наискорейшего спуска, примененным к системе $\mathbf{A}^T \mathbf{A}x = \mathbf{A}^T b$. Тогда на основании соотношения (4) можно утверждать, что в методе (4.9) на каждом шаге минимизируется функционал $\Phi(x) = \|b - \mathbf{A}x\|_2^2$ по направлению $-\nabla \Phi(x)$ ³.

ПРИМЕР 3. Если положить $\mathcal{K} = \mathcal{L}$ и на различных итерациях в качестве вектора v_k циклически с повторением выбирать единичные орты $e_1, e_2, \dots, e_n, e_1, \dots$ то получится рассмотренный в §3.1 метод Гаусса-Зейделя⁴.

ПРИМЕР 4. Выбор на k -й итерации $v_k = w_k = \mathbf{A}^T e_k$ дает ABS-класс методов [1]. Чтобы для различных итераций выполнялось условие $\mathcal{K}_i \perp \mathcal{K}_j$, возможно либо хранение всех v_k с их ортогонализацией по мере нахождения (схема Хуанга), либо пересчет матрицы $\mathbf{A}$ (схема Абрамова). Первый вариант ведет к увеличению расходов

²Англ. Residual norm Steepest Descent — RnSD.

³Это утверждение можно получить и непосредственным анализом функционала, однако выкладки получаются значительно длиннее, чем в Примере 1.

⁴Обратный порядок выбора соответствует обратному методу Гаусса-Зейделя.

памяти для реализации алгоритма, второй — к изменению заполненности матрицы. Следовательно, данные методы пригодны лишь для решения плотных и/или небольших систем; с другой стороны, их единственным условием сходимости является существование решения как такового, а потому данный класс пригоден для СЛАУ с вырожденными и неквадратными матрицами.

Непосредственно из определения векторов v_k следует, что в данных методах на k -й итерации поправка к решению вычисляется из условия обращения k -го уравнения в тождество.

В простейшем случае, если предполагать, что матрица A квадратная и невырожденная, вычислительная схема имеет следующий вид (приведен вариант с пересчетом матрицы, метод ABR1ORT):

<p>Выполнять для $i = \overline{1, n}$</p> $x := x + \frac{b_i}{\ a_i\ _2^2} a_i^T$ <p>Выполнять для $j = \overline{i+1, n}$</p> $\alpha := \frac{(a_i, a_j)}{\ a_i\ _2^2}$ $a_j := a_j - \alpha a_i$ $b_j := b_j - \alpha b_i$ <p>увеличить j</p> <p>увеличить i</p>

4.3. Два важных выбора подпространств

В предыдущем параграфе были рассмотрены методы наискорейшего спуска (см. пример 1), в котором подпространства $\mathcal{K}$ и $\mathcal{L}$ были связаны соотношением $\mathcal{K} = \mathcal{L}$, и наискорейшего уменьшения невязки (см. пример 2), основанный на соотношении $\mathcal{L} = A\mathcal{K}$. Сами подпространства являлись одномерными, в качестве базиса $\mathcal{K}$ выступал вектор невязки, и было показано, что в этих случаях задача проектирования эквивалентна задаче минимизации функционалов.

Как оказывается, подобные утверждения справедливы и в гораздо более общих случаях, которые имеют важное значение при построении более сложных и эффективных методов.

Теорема 4.3.1 Если матрица A симметрична и положительно определена, то задача проектирования решения СЛАУ (1) на любое подпространство $\mathcal{K}$ ортогонально к нему самому⁵ является эквивалентной задаче минимизации функционала $\Phi_1(x) = \|x - x_*\|_A^2$ на пространстве $\mathcal{K}$.

⁵Т.е. ортогонально к пространству $\mathcal{L} = \mathcal{K}$.

Доказательство. По условию теоремы $\mathcal{K} = \mathcal{L}$, а следовательно, $\mathbf{V} = \mathbf{W}$. Выражение для функционала Φ_1 было уже найдено (см. (4.8)), а сам функционал по свойствам нормы является строго выпуклым. Таким образом, сформулированная в условии задача минимизации сводится к нахождению

$$y = \arg \min_y \Phi_1(x_0 + \mathbf{V}y).$$

Рассмотрим эту задачу. В силу выпуклости достаточно найти стационарную точку функционала $\Psi(y) = \Phi_1(x_0 + \mathbf{V}y)$, т.е. решить систему $\nabla \Psi(y) = 0$. Пренебрегая постоянным слагаемым в (4.8), имеем

$$\begin{aligned} \Psi(y) &= (x_0 + \mathbf{V}y)^T \mathbf{A}(x_0 + \mathbf{V}y) - 2b^T(x_0 + \mathbf{V}y) = \\ &= (x_0^T \mathbf{A} - b^T)x_0 - b^T x_0 + 2(x_0^T \mathbf{A} - b^T)\mathbf{V}y + y^T(\mathbf{V}^T \mathbf{A} \mathbf{V})y = \\ &= y^T(\mathbf{V}^T \mathbf{A} \mathbf{V})y - r_0^T x_0 - b^T x_0 - 2r_0^T \mathbf{V}y. \end{aligned}$$

Градиент этого функционала равен

$$\nabla \Psi(y) = 2\mathbf{V}^T \mathbf{A} \mathbf{V}y - 2\mathbf{V}^T r_0.$$

Приравнивая его к нулю, получим

$$y = (\mathbf{V}^T \mathbf{A} \mathbf{V})^{-1} \mathbf{V}^T r_0,$$

что в точности совпадает с выражением для y из формулы (4.4), если в ней положить $\mathbf{V} = \mathbf{W}$. $\square$

Теорема 4.3.2 Для произвольной невырожденной матрицы $\mathbf{A}$ задача проектирования решения СЛАУ (1) на любое подпространство $\mathcal{K}$ ортогонально к подпространству $\mathcal{L} = \mathcal{A}\mathcal{K}$ является эквивалентной задаче минимизации функционала $\Phi_2(x) = \|r_x\|_2^2$ на пространстве $\mathcal{K}$.

Доказательство. Подставив в формулу (4.4) соотношение для базисов $\mathbf{W} = \mathbf{A}\mathbf{V}$, получим

$$y = (\mathbf{V}^T \mathbf{A}^T \mathbf{A} \mathbf{V})^{-1} \mathbf{V}^T \mathbf{A}^T r_0.$$

Это означает, что рассматриваемая ситуация эквивалентна выбору $\mathcal{L} = \mathcal{K}$ для симметризованной системы $\mathbf{A}^T \mathbf{A}x = \mathbf{A}^T b$. Учитывая соотношение (4) и применяя к такой системе предыдущую теорему, получаем сформулированное в условии утверждение. $\square$

4.4. Подпространства Крылова

При построении и реализации проекционных методов важную роль играют так называемые подпространства Крылова, часто выбираемые в качестве $\mathcal{K}$.

Определение 4.4.1 Подпространством Крылова размерности m , порожденным вектором v и матрицей $\mathbf{A}$ называется линейное пространство

$$K_m(v, \mathbf{A}) \stackrel{\text{def}}{=} \text{span} \{v, \mathbf{A}v, \mathbf{A}^2v, \dots, \mathbf{A}^{m-1}v\}. \quad (4.10)$$

В качестве вектора v обычно выбирается невязка начального приближения r_0 ; тогда выбор подпространства $\mathcal{L}$ и способ построения базисов подпространств полностью определяет вычислительную схему метода⁶.

К идее использования подпространств Крылова можно прийти, например, следующим образом.

При построении релаксационных методов (см. §3.1) использовалось представление матрицы $\mathbf{A}$ в виде $\mathbf{A} = \mathbf{D} - \mathbf{E} - \mathbf{F}$. Было также показано, что методы Якоби и Гаусса-Зейделя являются частными случаями класса методов, основанного на расщеплении $\mathbf{A}$ в виде разности (3.9) двух матриц $\mathbf{K}$ и $\mathbf{R}$. Тогда исходная система (1) может быть записана в виде

$$\mathbf{K}x = b + \mathbf{R}x = b + (\mathbf{K} - \mathbf{A})x,$$

что позволяет построить итерационный процесс

$$\mathbf{K}x_{k+1} = \mathbf{K}x_k + (b - \mathbf{A}x_k),$$

или, что то же самое,

$$x_{k+1} = x_k + \mathbf{K}^{-1}r_k. \quad (4.11)$$

Выберем $\mathbf{K} = \mathbf{I}$ и $\mathbf{R} = \mathbf{I} - \mathbf{A}$, тогда процесс (4.11) будет сведен к виду

$$x_{k+1} = x_k + r_k, \quad (4.12)$$

откуда следует

$$x_k = x_0 + r_0 + r_1 + \dots + r_{k-1}. \quad (4.13)$$

Умножив обе части (4.12) слева на $(-\mathbf{A})$ и прибавив к ним b , получим

$$b - \mathbf{A}x_{k+1} = b - \mathbf{A}x_k - \mathbf{A}r_k = r_k - \mathbf{A}r_k,$$

⁶Иногда, впрочем, могут использоваться эквивалентные подходы, основанные на теоремах 4.3.1 и 4.3.2. См., напр., построение метода GMRES в §5.3.

что позволяет найти выражение для невязки на k -ой итерации через невязку начального приближения:

$$r_k = (\mathbf{I} - \mathbf{A})r_{k-1} = (\mathbf{I} - \mathbf{A})^k r_0. \quad (4.14)$$

После подстановки (4.14) в (4.13) получаем

$$x_k = x_0 + \left[\sum_{j=0}^{k-1} (\mathbf{I} - \mathbf{A})^j \right] r_0.$$

т.е. $\delta_x \in \text{span} \{r_0, \mathbf{A}r_0, \dots, \mathbf{A}^{k-1}r_0\} = K_k(r_0, \mathbf{A})$.

Непосредственно из определения вытекают следующие свойства подпространств Крылова. Пусть q — полином, такой что $q(\mathbf{A})v = 0$, причем имеет степень $\deg q = \mu$, минимальную среди всех таких полиномов. Тогда

- $\forall m (m \geq \mu) : K_m(v, \mathbf{A}) = K_\mu(v, \mathbf{A})$.
Более того, K_μ инвариантно относительно $\mathbf{A}$.
- $\forall m : \dim(K_m) = m \iff m \leq \mu$.

Кроме того, из (4.14) следует что в методах, использующих подпространства Крылова, невязка на k -ой итерации выражается через начальную невязку некоторым матричным полиномом (см. приложение В).

4.5. Базис подпространства Крылова. Ортогонализация Арнольди

Для построения базиса в пространстве Крылова $K_m(v_1, \mathbf{A})$ можно применить следующий подход.

Найдем сначала векторы $w_1 = v_1$, $w_2 = \mathbf{A}w_1$, $w_3 = \mathbf{A}^2w_1 = \mathbf{A}w_2, \dots, w_m = \mathbf{A}^{m-1}w_1 = \mathbf{A}w_{m-1}$. По определению (4.10),

$$K_m(v_1, \mathbf{A}) = \text{span}\{w_1, w_2, \dots, w_m\}.$$

Перейдем теперь от $\{w_1, w_2, \dots, w_m\}$ к $\{v_1, v_2, \dots, v_m\}$, применив процедуру ортогонализации

$$v_{k+1} = w_{k+1} - \sum_{i=1}^k \alpha_i v_i \quad (4.15)$$

и затем пронормировав полученные векторы.

Предположим, что предыдущие k векторов уже построены, т.е.

$$\forall i, j (1 \leq i, j \leq k) : (v_i, v_j) = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases} \quad (4.16)$$

Тогда (4.15) можно записать как

$$v_{k+1} = \mathbf{A}v_k - \sum_{i=1}^k \alpha_i v_i.$$

Для выполнения условия ортогональности v_{k+1} ко всем предыдущим векторам умножим это равенство скалярно на v_j ($j \leq k$) и приравняем результат к нулю

$$(\mathbf{A}v_k, v_j) - \sum_{i=1}^k \alpha_i (v_i, v_j) = 0. \quad (4.17)$$

С учетом (4.16) отсюда легко получить выражение для коэффициентов α_j

$$\alpha_j = (\mathbf{A}v_k, v_j).$$

Описанный метод может быть оформлен в виде следующей процедуры, называемой ортогонализацией Арнольди⁷ [4].

Входные данные: v_1 , такой что $\|v_1\|_2 = 1$; $\mathbf{A}$; m
Выполнять для $j = \overline{1, m}$
 $w := \mathbf{A}v_j$
Выполнять для $i = \overline{1, j}$
 $h_{ij} := (w, v_i)$
 $w := w - h_{ij}v_i$
увеличить i
 $h_{j+1,j} := \|w\|_2$
Если $h_{j+1,j} = 0$, **то КОНЕЦ. Базис построен.**
 $v_{j+1} := \frac{w}{h_{j+1,j}}$
увеличить j

Замечание. Для коэффициентов ортогонализации здесь введена двойная индексация, с учетом которой внутренний цикл алгоритма алгебраически записывается как

$$h_{j+1,j}v_{j+1} = \mathbf{A}v_j - \sum_{i=1}^j h_{ij}v_i. \quad (4.18)$$

⁷Данный вариант построения базиса основан на модифицированной ортогонализации Грама-Шмидта, примененной к векторам $v_1, \dots, \mathbf{A}^{m-1}v_1$. Возможны и другие варианты, например ортогонализация Арнольди-Хаусхолдера [13].

Коэффициенты ортогонализации $h_{i,j}$ можно объединить в виде матрицы $\mathbf{H}$, дополнив в ней недостающие позиции нулями. При этом, как видно из алгоритма, для заданной размерности пространства m генерируется $m + 1$ векторов. Последний вектор v_{m+1} (возможно, нулевой) в матричном представлении означает расширение базиса $\mathbf{V}$ одним дополнительным столбцом, т.е.

$$\begin{aligned}\mathbf{V}_m &= [v_1|v_2|\dots|v_m]; \\ \mathbf{V}_{m+1} &= [v_1|v_2|\dots|v_m|v_{m+1}].\end{aligned}$$

Соответствующий вектору v_{m+1} коэффициент $h_{m+1,m}$ означает расширение матрицы $\mathbf{H}$ одной дополнительной строкой (возможно, нулевой).

Пусть $\bar{\mathbf{H}}_m$ — это $(m + 1) \times m$ матрица коэффициентов h_{ij} , дополненная последней строкой за счет $h_{m+1,m}$, а $\mathbf{H}_m$ — та же самая матрица без последней строки, имеющая размерность $m \times m$. Тогда непосредственно из описания алгоритма Арнольди и (4.18) следует что матрица $\mathbf{H}_m$ является матрицей в верхней форме Хессенберга и для нее справедливы соотношения

$$\mathbf{A}\mathbf{V}_m = \mathbf{V}_{m+1}\bar{\mathbf{H}}_m = \mathbf{V}_m\mathbf{H}_m + w_m\mathbf{e}_m^T; \quad (4.19)$$

$$\mathbf{V}_m^T\mathbf{A}\mathbf{V}_m = \mathbf{H}_m. \quad (4.20)$$

Кроме того, вследствие ортонормальности базиса $\{v_j\}$ имеет место равенство

$$\mathbf{V}^T v_k = \mathbf{e}_k. \quad (4.21)$$

ПРИМЕР. Пусть $m = 2$, а матрица $\mathbf{A}$ и вектор v_1 равны

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}; \quad v_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

Подпространство Крылова $K_2(v_1, \mathbf{A})$ есть

$$\begin{aligned}\text{span}\{v_1, \mathbf{A}v_1\} &= \{u \mid u = \alpha_1(0, 1, 0)^T + \alpha_2(1, 1, 0)^T, \alpha_i \in \mathcal{R}\} = \\ &= \{u \mid u = (\beta_1, \beta_2, 0)^T, \beta_i \in \mathcal{R}\}.\end{aligned}$$

Применение алгоритма Арнольди дает

$$\begin{array}{l}
v_1 = (0, 1, 0)^T \\
\text{при } j = 1 \quad w := \mathbf{A}v_1 = (1, 1, 0)^T \\
\quad \text{при } i = 1 \quad h_{11} := 1 \quad w := (1, 1, 0)^T - (0, 1, 0)^T = (1, 0, 0)^T \\
\quad h_{21} := \|w\|_2 = 1 \\
\quad v_2 := w = (1, 0, 0)^T \\
\text{при } j = 2 \quad w := \mathbf{A}v_2 = (1, 0, 0)^T \\
\quad \text{при } i = 1 \quad h_{12} := 0 \quad w := (1, 0, 0)^T \\
\quad \text{при } i = 2 \quad h_{22} := 1 \quad w := (1, 0, 0)^T - (1, 0, 0)^T = (0, 0, 0)^T \\
\quad h_{32} := \|w\|_2 = 0 \\
\quad v_3 := w = (0, 0, 0)^T
\end{array}$$

Результатом является базис из векторов $v_1 = (0, 1, 0)^T$ и $v_2 = (1, 0, 0)^T$, а во введенных ранее матричных обозначениях:

$$\begin{array}{l}
\mathbf{V}_{m+1} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}; \quad \mathbf{V}_m = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix}; \\
\bar{\mathbf{H}}_m = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 0 \end{pmatrix}; \quad \mathbf{H}_m = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}.
\end{array}$$

Непосредственным вычислением нетрудно проверить для этих матриц ранее приведенные соотношения (4.19) и (4.20).

4.6. Биортогонализация Ланцоша

Алгоритм ортогонализации Арнольди, предложенный в предыдущем параграфе, для построения каждого нового вектора v_k требует нахождения $(k-1)$ скалярных произведений и столько же операций линейного комбинирования. Однако, как оказывается, при отказе от требования ортогональности базиса в пользу некоторого более общего условия можно построить процедуру меньшей сложности.

Определение 4.6.1 Системы векторов $\{x_i\}_{i=1}^m$ и $\{y_i\}_{i=1}^m$ называются биортогональными, если скалярное произведение (x_i, y_i) обращается в ноль при $i \neq j$.

Очевидно, что в случае $x_i \equiv y_i$ условие биортогональности сводится к обычному условию ортогональности.

Теорема 4.6.1 Пусть векторы v_1 и w_1 таковы, что $(v_1, w_1) \neq 0$ и пусть системы векторов $\{v_i\}_{i=1}^m$ и $\{w_i\}_{i=1}^m$ определяются соотношениями:

$$v_{i+1} = \mathbf{A}v_i - \alpha_i v_i - \beta_i v_{i-1}, \quad v_0 \equiv 0; \quad (4.22)$$

$$w_{i+1} = \mathbf{A}^T w_i - \alpha_i w_i - \beta_i w_{i-1}, \quad w_0 \equiv 0; \quad (4.23)$$

$$\alpha_i = \frac{(\mathbf{A}v_i, w_i)}{(v_i, w_i)}; \quad (4.24)$$

$$\beta_i = \frac{(v_i, w_i)}{(v_{i-1}, w_{i-1})}, \quad \beta_1 \equiv 0. \quad (4.25)$$

Тогда

- системы $\{v_i\}_{i=1}^m$ и $\{w_i\}_{i=1}^m$ являются биортогональными;
- каждая из систем $\{v_i\}_{i=1}^m$ и $\{w_i\}_{i=1}^m$ является линейно независимой и образует базис в $K_m(v_1, \mathbf{A})$ и $K_m(w_1, \mathbf{A}^T)$ соответственно.

Доказательство. Первое утверждение теоремы доказывается по индукции. Действительно, пара векторов v_1 и w_1 удовлетворяет условию биортогонализации. Предположим теперь, что уже построены биортогональные наборы $\{v_1, \dots, v_i\}$ и $\{w_1, \dots, w_i\}$, и далее покажем, что для вектора v_{i+1} , определяемого соотношением (4.22), имеет место $(v_{i+1}, w_k) = 0$, $k = \overline{1, i}$.

Умножим (4.22) скалярно на w_k

$$(v_{i+1}, w_k) = (\mathbf{A}v_i, w_k) - \alpha_i(v_i, w_k) - \beta_i(v_{i-1}, w_k).$$

Если $k = i$, то по предположению индукции последнее скалярное произведение обращается в ноль и

$$\begin{aligned} (v_{i+1}, w_i) &= (\mathbf{A}v_i, w_i) - \alpha_i(v_i, w_i) = \\ &= (\mathbf{A}v_i, w_i) - \frac{(\mathbf{A}v_i, w_i)}{(v_i, w_i)}(v_i, w_i) = 0. \end{aligned}$$

Если же $k < i$, то

$$\begin{aligned} (v_{i+1}, w_k) &= (v_i, \mathbf{A}^T w_k) - \beta_i(v_{i-1}, w_k) = \\ &= (v_i, w_{k+1} + \alpha_k w_k + \beta_k w_{k-1}) - \beta_i(v_{i-1}, w_k) = \\ &= (v_i, w_{k+1}) + \alpha_k(v_i, w_k) + \beta_k(v_i, w_{k-1}) - \beta_i(v_{i-1}, w_k). \end{aligned}$$

По предположению индукции, при $k < i - 1$ все четыре скалярные произведения обращаются в ноль; при $k = i - 1$ равны нулю скалярные произведения во втором и третьем слагаемых, и тогда

$$\begin{aligned} (v_{i+1}, w_{i-1}) &= (v_i, w_i) - \beta_i(v_{i-1}, w_{i-1}) \\ &= (v_i, w_i) - \frac{(v_i, w_i)}{(v_{i-1}, w_{i-1})}(v_{i-1}, w_{i-1}) = 0. \end{aligned}$$

Аналогичным образом доказывается, что $(v_k, w_{i+1}) = 0$ для $k = \overline{1, i}$.

Чтобы доказать второе утверждение теоремы, заметим, что непосредственно из (4.22) следует $\text{span}\{v_1, \dots, v_m\} = K_m(v_1, \mathbf{A})$. Остается лишь показать линейную независимость векторов $v_1, \dots, v_m$. Предположим от противного, что существуют коэффициенты γ_k , для которых

$$\gamma_1 v_1 + \gamma_2 v_2 + \dots + \gamma_m v_m = 0.$$

Составляя скалярные произведения с векторами w_k , $k = \overline{1, m}$, получим

$$\gamma_k (v_k, w_k) = 0, \quad k = \overline{1, m},$$

а так как по ранее доказанной биортогональности $(v_k, w_k) \neq 0$ ⁸, то все коэффициенты γ_k должны быть нулевыми.

Аналогичные рассуждения для $w_1, \dots, w_m$ завершают доказательство теоремы. $\square$

Процедура построения векторов v и w согласно формулам (4.22)–(4.25) называется биортогонализацией Ланцоша.

Очевидно, что (4.22) является частным случаем (4.18), где из всей суммы оставлены только два последних слагаемых; точно так же (4.23) является частным случаем (4.18), записанной для матрицы $\mathbf{A}^T$ и вектора w_1 . При этом коэффициенты ортогонализации в (4.22) и (4.23) одинаковы.

Из сказанного следует, что можно записать аналог матричной формулы (4.20)

$$\mathbf{W}_m^T \mathbf{A} \mathbf{V}_m = \mathbf{T}_m, \quad (4.26)$$

где $\mathbf{T}_m$ — симметричная⁹ трехдиагональная матрица, элементы которой определяются соотношениями

$$[\mathbf{T}_m]_{ii} = \alpha_i(v_i, w_i); \quad (4.27)$$

$$[\mathbf{T}_m]_{i+1,i} = [\mathbf{T}_m]_{i,i+1} = \beta_{i+1}(v_i, w_i), \quad (4.28)$$

легко выводимыми из (4.22)–(4.23).

Замечание. Основным недостатком биортогонализации Ланцоша является возможность возникновения ситуации, когда $(v_i, w_i) = 0$; при этом продолжение процесса становится невозможным из-за неопределенности коэффициента β_{i+1} .

⁸Иначе способ определения коэффициентов β сделал бы невозможным построение всех следующих векторов.

⁹Симметричность следует из того, что для построения v_{i+1} и w_{i+1} используются одни и те же коэффициенты ортогонализации.

Глава 5

Методы крыловского типа

В §4.1 был построен класс проекционных методов, основой которого служит условие Петрова-Галеркина (4.1) и формула (4.5). В §4.2 были приведены примеры построения нескольких методов для достаточно простого случая одномерных подпространств $\mathcal{K}$ и $\mathcal{L}$.

Однако выбор $m = 1$, как правило, дает медленную сходимость и соответствующие методы оказываются пригодными лишь для СЛАУ небольшой размерности или специальных случаев. В настоящее время широкое распространение получили проекционные методы, которые в качестве $\mathcal{K}$ используют описанные в §4.4 подпространства Крылова размерности больше единицы. Такие методы называются методами крыловского типа¹. Для упрощения вычислений $(\mathbf{W}^T \mathbf{A} \mathbf{V})^{-1}$ из формулы (4.5) в них используются соотношения (4.19)–(4.21).

5.1. FOM: Метод полной ортогонализации

Наиболее характерным примером применения описанного в §4.1 подхода является метод полной ортогонализации, известный также как метод Арнольди или FOM².

Положим $\mathcal{K} = \mathcal{L}$ и будем проектировать искомое решение на подпространство Крылова $K_m(v_1, \mathbf{A})$, где в качестве v_1 выступает нормированный вектор начальной невязки

$$v_1 = r_0 / \beta, \quad \beta = \|r_0\|_2. \quad (5.1)$$

Тогда $\mathbf{V} = \mathbf{W}$, для построения базиса можно воспользоваться ортогонализацией Арнольди (§4.5), а матрица $\mathbf{W}^T \mathbf{A} \mathbf{V}$ в силу (4.20) будет

¹Англ. Krylov sequence methods.

²Full Orthogonalization Method.

$r_0 := b - Ax_0$
 $\beta := \|r_0\|_2; \quad v_1 := r_0/\beta$
Создать $(m \times m)$ -матрицу H_m ; **положить** $H_m = 0$
Создать m -вектор f ; **положить** $f = \beta e_1$
Для $j = \overline{1, m}$
 $w_j := Av_j$
 Для $i = \overline{1, j}$
 $h_{ij} := (w_j, v_i)$
 $w_j := w_j - h_{ij}v_i$
 увеличить i
 $h_{j+1,j} := \|w_j\|_2$
 Если $h_{j+1,j} = 0$
 то $m := j$
 Прервать цикл по j
 $v_{j+1} := w_j/h_{j+1,j}$
увеличить j
Для $i = \overline{2, m}$
 Обнулить $h_{i,i-1}$, **выполнив гауссовское исключение**
 для i -й строки системы $H_m y = f$
 относительно $(i - 1)$ -й
увеличить i
Найти y из верхнетреугольной системы $H_m y = f$
 $x_m := x_0 + \sum_{i=1}^m y_i v_i$

Рис. 5.1. FOM

иметь верхнюю хессенбергову форму и следовательно, окажется легко обратимой. Решение СЛАУ с такой матрицей сведется к $(m - 1)$ гауссовскому исключению строк и последующему обратному ходу по верхнетреугольной матрице.

Присутствующий в (4.5) сомножитель $\mathbf{W}^T r_0$ в силу (5.1) и (4.21) будет равен $\beta \mathbf{e}_1$. Таким образом, формула (4.5) будет сведена к

$$x_m = x_0 + \beta \mathbf{V} \mathbf{H}_m^{-1} \mathbf{e}_1. \quad (5.2)$$

На основании (5.2) и описанного в §4.5 алгоритма ортогонализации Арнольди, необходимого для построения базиса $\mathbf{V}$, метод FOM может быть записан как на рис.5.1.

В такой формулировке метод обладает серьезным недостатком: заранее неизвестно, какой должна быть размерность подпространства m , чтобы найденное решение x_m имело достаточную точность. Однако существует теорема, позволяющая оценить уменьшение нормы невязки без явного нахождения решения.

Теорема 5.1.1 *Норма вектора невязки $\|r_m\|_2$ в методе FOM удовлетворяет соотношению*

$$\|r_m\|_2 = h_{m+1,m} |y_m|. \quad (5.3)$$

Доказательство. Найдем невязку r_m , воспользовавшись свойством (4.19)

$$\begin{aligned} r_m &= b - \mathbf{A}(x_0 + \mathbf{V}_m y) = r_0 - \mathbf{A} \mathbf{V}_m y = \\ &= \beta v_1 - (\mathbf{V}_m \mathbf{H}_m + w_m \mathbf{e}_m^T) y. \end{aligned}$$

Подставим сюда выражения $y = \beta \mathbf{H}_m^{-1} \mathbf{e}_1$ и $w_m = h_{m+1,m} v_{m+1}$:

$$\begin{aligned} r_m &= \beta v_1 - \beta \mathbf{V}_m \mathbf{e}_1 - h_{m+1,m} (\mathbf{e}_m, y) v_{m+1} = \\ &= -h_{m+1,m} y_m v_{m+1}. \end{aligned}$$

С учетом того, что векторы v_j нормированы, а коэффициенты $h_{j+1,j}$ вычисляются как нормы векторов $\mathbf{e}_j$, следовательно, неотрицательны, отсюда и следует утверждение теоремы. $\square$

Проверка условия выхода $\|r\|_2 < \varepsilon$ в соответствии с (5.3) может быть достаточно просто введена в алгоритм FOM, если гауссовское исключение строк $\mathbf{H}$ проводить сразу же в цикле ортогонализации Арнольди (при этом коэффициенты исключения необходимо хранить в отдельном векторе, так как матрица $\mathbf{H}$ по мере нахождения базиса расширяется).

Однако на практике чаще применяется другой подход, гораздо более простой с точки зрения программной реализации. Его суть заключается в том, что заранее выбирается некоторая размерность подпространства m , относительно небольшая по сравнению с порядком СЛАУ. После того как решение x_m найдено, проверяется, является ли оно достаточно точным. Если необходимая точность еще не достигнута, то весь процесс повторяется с вектором x_m в качестве начального приближения:

Выбрать $m < n$ и начальное приближение x_0
Начало
 (алгоритм рис.5.1 для выбранного m)
 $x_0 := x_m$
Повторять, пока $\|b - Ax_0\|_2 > \varepsilon$

Такая схема носит название *перезапускаемого FOM*³ или FOM(m).

Замечание. При программной реализации FOM и FOM(m) основные затраты памяти приходятся на хранение базисных векторов $v_1, \dots, v_m$ подпространства Крылова; они составляют $O(mn)$ вещественных чисел.

Еще одним вариантом метода является *неполная ортогонализация*, когда вектор w_{j+1} ортогонализуется не ко всем ранее найденным $v_1, \dots, v_j$, а только к k предыдущим векторам $v_{j-m+1}, \dots, v_j$. Это позволяет уменьшить расходы памяти при программной реализации метода; соответствующая схема носит название IOM⁴ [13].

5.2. Предобусловливание в схеме метода

На примере построенного в предыдущем параграфе FOM покажем, как в схему проекционного метода может быть введено предобусловливание (2.1) без необходимости явного вычисления матричного произведения.

Перепишем алгоритм FOM в применении к системе (2.1) с матрицей предобусловливателя M . Очевидно, что в нем необходимо изменить только те фрагменты, где фигурирует матрица СЛАУ и вектор правой части:

³Англ. Restarted FOM.

⁴Incomplete Orthogonalization Method. Встречается также название Truncated FOM.

Построить матрицу предобусловливателя M
 $z := b - Ax_0$
Найти r_0 из системы $Mr_0 = z$
 $\beta := \|r_0\|_2; \quad v_1 := r_0/\beta$
Создать $(m \times m)$ -матрицу H_m ; положить $H_m = 0$
Создать m -вектор f ; положить $f = \beta e_1$
Для $j = \overline{1, m}$
 $z := Av_j$
 Найти w_j из системы $Mw_j = z$
 Для $i = \overline{1, j}$
 $h_{ij} := (w_j, v_i)$
 $w_j := w_j - h_{ij}v_i$
 увеличить i
 $h_{j+1,j} := \|w_j\|_2$
 Если $h_{j+1,j} = 0$
 то $m := j$
 Прервать цикл по j
 $v_{j+1} := w_j/h_{j+1,j}$
увеличить j
Для $i = \overline{2, m}$
 Обнулить $h_{i,i-1}$, выполнив гауссовское исключение
 для i -й строки системы $H_m y = f$
 относительно $(i-1)$ -й
увеличить i
Найти y из верхнетреугольной системы $H_m y = f$
 $x_m := x_0 + \sum_{i=1}^m y_i v_i$

Рис. 5.2. Предобусловленный FOM

$r_0 := \mathbf{M}^{-1}(b - \mathbf{A}x_0)$
.....
Для $j = \overline{1, m}$
$w_j := \mathbf{M}^{-1}\mathbf{A}v_j$
.....
увеличить j
.....
$x_m := x_0 + \sum_{i=1}^m y_i v_i$

Таким образом, вектор начальной невязки r_0 и векторы w_j находятся из решения систем $\mathbf{M}r_0 = b - \mathbf{A}x_0$ и $\mathbf{M}w_j = \mathbf{A}v_j$ соответственно. Напомним, что для матрицы $\mathbf{M}$ в §2.1 было сформулировано требование легкой обратимости. Для LU-предобусловливания решение таких систем сводится к решению двух треугольных систем, для SSOR-предобусловливания — к решению двух треугольных систем и умножения на диагональную матрицу и т.д.

Следовательно, алгоритм FOM с введенным в него предобусловливанием матрицей $\mathbf{M}$ можно записать как на рис. 5.2.

Подобным же образом предобусловливание может быть введено и в схему любого другого проекционного метода, требующего лишь возможности вычисления матрично-векторных произведений без пересчета элементов матрицы СЛАУ.

5.3. GMRES: Метод обобщенных минимальных невязок

Пусть теперь пространства $\mathcal{K}$ и $\mathcal{L}$ связаны соотношением $\mathcal{L} = \mathbf{A}\mathcal{K}$, причем в качестве $\mathcal{K}$ по-прежнему используется подпространство Крылова $K_m(v_1, \mathbf{A})$ с выбором $v_1 = r_0/\beta$, $\beta = \|r_0\|_2$. Для построения ортонормального базиса $\mathcal{K}$ снова воспользуемся ортогонализацией Арнольди.

Однако вместо проектирования (4.5) будем рассматривать эквивалентную (см. теорему 4.3.2) задачу минимизации функционала $\Phi_2(x) = \|r_x\|_2^2$ на пространстве $\mathcal{K}$:

$$y = \arg \min_y \|b - \mathbf{A}(x_0 + \mathbf{V}_m y)\|_2^2.$$

Вычислительная схема, построенная с использованием таких предпосылок, называется *методом обобщенных минимальных невязок (GMRES)*⁵.

⁵General Minimal RESiduals.

Выбрать $m < n$ и начальное приближение x_0

$p := m$

$r_0 := b - Ax_0; \beta := \|r_0\|_2$

Создать $((m + 1) \times m)$ -матрицу H

Создать $(m + 1)$ -вектор g

Начало

$H := 0; g := \beta e_1$

$v_1 := r_0/\beta$

Для $i = \overline{1, m}$

$w := Av_i$

Для $k = \overline{1, i}$

$h_{ki} := (w, v_k)$

$w := w - h_{ki}v_k$

увеличить k

$h_{i+1,i} := \|w\|_2$

$v_{i+1} := w/h_{i+1,i}$

Для $k = \overline{1, i-1}$

$(h_{1i}, \dots, h_{i+1,i})^T := G_k(h_{1i}, \dots, h_{i+1,i})^T$

увеличить k

Построить G_i так, что $[G_i h_{\cdot, i}]_{i+1} = 0$

$g := G_i g$

Если $|g_{i+1}| < \varepsilon$

то $p := i$

выйти из цикла по i

увеличить i

Решить треугольную СЛАУ $H_{1:p, 1:p} y = g_{1:p}$

$x_0 := x_0 + \sum_{i=1}^p y_i v_i$

Если $p < m$

то КОНЕЦ

$r_0 := b - Ax_0; \beta := \|r_0\|_2$

Продолжать, пока $\beta > \varepsilon$

Рис. 5.3. GMRES(m) с вращениями Гивенса

С использованием определения вектора v_1 и соотношений (4.19), (4.21) имеем

$$\begin{aligned}
r = b - \mathbf{A}(x_0 + \mathbf{V}_m y) &= r_0 - \mathbf{A}\mathbf{V}_m y = \\
&= \beta v_1 - \mathbf{V}_{m+1} \bar{\mathbf{H}}_m y = \\
&= \beta \mathbf{V}_{m+1} \mathbf{e}_1 - \mathbf{V}_{m+1} \bar{\mathbf{H}}_m y = \\
&= \mathbf{V}_{m+1} (\beta \mathbf{e}_1 - \bar{\mathbf{H}}_m y). \tag{5.4}
\end{aligned}$$

Поскольку матрица $\mathbf{V}_{m+1}$ составлена из ортонормированных векторов-столбцов, справедливо равенство

$$\Psi(y) = \Phi_2(x_0 + \mathbf{V}_m y) = \|\beta \mathbf{e}_1 - \bar{\mathbf{H}}_m y\|_2^2. \tag{5.5}$$

Таким образом, для нахождения коэффициентов линейного комбинирования векторов $\{v_i\}$ в GMRES необходимо решить СЛАУ

$$\bar{\mathbf{H}}_m y = \beta \mathbf{e}_1, \tag{5.6}$$

которая является переопределенной (так как матрица $\bar{\mathbf{H}}_m$ имеет размерность $(m+1) \times m$) и поэтому, как видно из (5.5), должна решаться в смысле наименьших квадратов.

Замечание. Соотношение (5.6), на котором основан метод, может быть получено и с помощью непосредственного применения проекционного подхода (4.1)–(4.5). Однако формула (4.5) в этом случае требует более сложной интерпретации [1].

Матрица системы (5.6) имеет верхнюю хессенбергову форму. Поэтому (5.6) проще всего решать с помощью предварительного приведения к верхнетреугольному виду; последняя строка $\bar{\mathbf{H}}_m$ при этом обнуляется. Однако в отличие от FOM, где похожая на (5.6) система является квадратной, в GMRES для приведения к треугольному виду используются ортогональные преобразования⁶.

С учетом верхней хессенберговой формы наиболее простым вариантом оказываются вращения Гивенса [2, 5, 15]. Всего необходимо m последовательно применяемых вращений $G_1, G_2, \dots, G_m$, причем G_k строится так, чтобы его применение к $(h_{kk}, h_{k+1,k})^T$ обнуляло второй компонент этого вектора.

Обозначим $\mathbf{Q}_m = G_m G_{m-1} \dots G_1$. Будучи произведением матриц вращений, $\mathbf{Q}_m$ сама является ортогональной матрицей. Последовательное применение вращений приводит к переходу от (5.6) к

$$\bar{\mathbf{R}}_m y = \bar{g}, \tag{5.7}$$

⁶См. доказательство теоремы 5.3.1. Ортогональность преобразований гарантирует неувеличение нормы невязки, которая при этом может быть легко оценена подобно тому, как это делается в FOM.

где $\bar{\mathbf{R}}_m = \mathbf{Q}_m \bar{\mathbf{H}}_m$, $\bar{g} = \mathbf{Q}_m(\beta \mathbf{e}_1)$. По аналогии с матрицей $\mathbf{H}_m$, обозначим за $\mathbf{R}_m$ матрицу $\bar{\mathbf{R}}_m$ без последней строки (которая по построению вращений является нулевой), а за g — вектор $\bar{g}$ без последнего коэффициента.

Тогда имеет место следующая теорема:

Теорема 5.3.1 Вектор y , доставляющий минимум функционалу $\Psi(y)$ в методе GMRES, определяется соотношением $y = \mathbf{R}_m^{-1}g$. При этом норма невязки r_m , соответствующей решению $x_m = x_0 + \mathbf{V}_m y$, равна

$$\|b - \mathbf{A}x_m\|_2 = |g_{m+1}|. \quad (5.8)$$

Доказательство. Первая часть утверждения теоремы следует из соотношения (5.4). С учетом ортогональности матрицы $\mathbf{Q}_m$ имеем

$$\begin{aligned} \|r_m\|_2^2 &= \|\beta \mathbf{e}_1 - \bar{\mathbf{H}}_m y\|_2^2 = \|\mathbf{Q}_m(\beta \mathbf{e}_1 - \bar{\mathbf{H}}_m y)\|_2^2 = \|\bar{g} - \bar{\mathbf{R}}_m y\|_2^2 = \\ &= |g_{m+1}|^2 + \|g - \mathbf{R}_m y\|_2^2. \end{aligned}$$

В правой части первое слагаемое не зависит от y , а следовательно, минимум всей суммы достигается при $y = \mathbf{R}_m^{-1}$.

Для доказательства второй части теоремы воспользуемся тем свойством ортогональных матриц, что $\mathbf{Q}_m^T \mathbf{Q}_m = \mathbf{I}$. То же соотношение (5.4) можно переписать в виде

$$\begin{aligned} r_m &= \mathbf{V}_{m+1}(\beta \mathbf{e}_1 - \bar{\mathbf{H}}_m y) = \mathbf{V}_{m+1} \mathbf{Q}_m^T \mathbf{Q}_m(\beta \mathbf{e}_1 - \bar{\mathbf{H}}_m y) = \\ &= \mathbf{V}_{m+1} \mathbf{Q}_m^T(\bar{g} - \bar{\mathbf{R}}_m y). \end{aligned}$$

Последний $(m+1)$ -й компонент вектора $\bar{\mathbf{R}}_m y$ равен нулю, а первые m , как только что было доказано, в точности совпадают с первыми m компонентами вектора $\bar{g}$. Поэтому $\bar{g} - \bar{\mathbf{R}}_m y = g_{m+1} \mathbf{e}_{m+1}$ и

$$r_m = g_{m+1} \mathbf{V}_{m+1} \mathbf{Q}_m^T \mathbf{e}_{m+1},$$

откуда и следует утверждение (5.8). $\square$

Как и в случае FOM, основной проблемой в GMRES является выбор размерности m пространства $\mathcal{K}$; хранение базисных векторов этого пространства определяет основные затраты памяти при программной реализации метода, составляющие $O(mn)$ вещественных чисел.

Наиболее распространенной модификацией GMRES является *перезапускаемый GMRES*⁷ или GMRES(m). Выбирается некоторая размерность $m < n$, определяемая, по существу, лишь доступным объемом памяти, и после обновления решения $x_m = x_0 + \mathbf{V}_m y$ оно принимается за новое начальное приближение, после чего процесс повторяется.

⁷Restarted GMRES.

Такое обновление после построения m базисных векторов и нахождения коэффициентов их комбинирования называется *GMRES-циклом*, тогда как построение одного очередного базисного вектора считается *GMRES-итерацией*.

Внутри цикла реализуется проверка условия завершения в соответствии с (5.8); такая проверка легко реализуется, если совмещать построение базиса с помощью процедуры Арнольди и применение вращений Гивенса к уже найденным элементам матрицы $\bar{\mathbf{H}}_m$.

Соответствующий алгоритм показан на рис.5.3.

Очевидно, что при программной реализации нет необходимости в непосредственном хранении матриц вращения G_i , так как каждая такая матрица полностью определяется двумя числами — косинусом и синусом угла поворота — и индексами компонент вектора, к которым применяется вращение.

Предобусловливание GMRES осуществляется точно так же, как предобусловливание FOM.

5.4. BCG: Метод бисопряженных градиентов

В §4.6 была описана биортогонализация Ланцоша, которая в отличие от ортогонализации Арнольди, использует для построения базиса экономичные трехчленные формулы. Выберем в качестве пространства $\mathcal{K}$ пространство Крылова $K_m(r_0, \mathbf{A})$, а в качестве $\mathcal{L}$ — пространство $K_m(\tilde{r}_0, \mathbf{A}^T)$, где вектор $\tilde{r}_0$ удовлетворяет условию $(r_0, \tilde{r}_0) \neq 0$.

Тогда в соответствии с (4.26), если в методе FOM заменить процедуру Арнольди процедурой Ланцоша, решение будет уточняться по формуле

$$x_m = x_0 + \beta \mathbf{V}_m \mathbf{T}_m^{-1} \mathbf{e}_1, \quad (5.9)$$

где $\beta = (r_0, \tilde{r}_0)$. Решение СЛАУ с трехдиагональной матрицей, очевидно, удобнее всего искать с помощью метода прогонки [4].

Описанная модификация FOM носит название *двойственного метода Ланцоша*. Однако подобная схема все еще требует одновременного хранения всех базисных векторов либо их перевычисления после нахождения коэффициентов. Поэтому на практике обычно используется другой подход.

Запишем LU-разложение для матрицы $\mathbf{T}_m$

$$\mathbf{T}_m = \mathbf{L}_m \mathbf{U}_m,$$

⁸Метод называется двойственным, так как с минимальными изменениями, практически не усложняющими его схему, может быть использован для одновременного решения (1) и системы с транспонированной матрицей $\mathbf{A}^T z = b$.

Выбрать начальное приближение x_0
 $r_0 := b - \mathbf{A}x_0$
Выбрать вектор $\tilde{r}_0$, удовлетворяющий условию $(r_0, \tilde{r}_0) \neq 0$
 $p_0 := r_0$
 $\tilde{p}_0 := \tilde{r}_0$
Для $j = 1, 2, \dots$
 $\alpha_j := (r_j, \tilde{r}_j) / (\mathbf{A}p_j, \tilde{p}_j)$
 $x_{j+1} := x_j + \alpha_j p_j$
 $r_{j+1} := r_j - \alpha_j \mathbf{A}p_j$
 $\tilde{r}_{j+1} := \tilde{r}_j - \alpha_j \mathbf{A}^T \tilde{p}_j$
 $\beta_j := (r_{j+1}, \tilde{r}_{j+1}) / (r_j, \tilde{r}_j)$
Если $\|r_{j+1}\|_2 < \varepsilon$
то КОНЕЦ (решение достигнуто)
Если $\beta_j = 0$
то КОНЕЦ (при данном $\tilde{r}_0$ метод не сходится)
 $p_{j+1} := r_{j+1} + \beta_j p_j$
 $\tilde{p}_{j+1} := \tilde{r}_{j+1} + \beta_j \tilde{p}_j$
увеличить j

Рис. 5.4. BCG

и определим матрицу $\mathbf{P}_m$ следующим образом:

$$\mathbf{P}_m = \mathbf{V}_m \mathbf{U}_m^{-1}. \quad (5.10)$$

Подставим это выражение в (4.22)

$$\begin{aligned} x_m &= x_0 + \beta \mathbf{V}_m \mathbf{U}_m^{-1} \mathbf{L}_m^{-1} \mathbf{e}_1 = \\ &= x_0 + \beta \mathbf{P}_m \mathbf{L}_m^{-1} \mathbf{e}_1. \end{aligned} \quad (5.11)$$

Рассмотрим теперь свойства матрицы $\mathbf{P}_m$. По аналогии с (5.10) запишем

$$\tilde{\mathbf{P}}_m = \mathbf{W}_m (\mathbf{L}_m^T)^{-1}, \quad (5.12)$$

Тогда имеет место

$$(\tilde{\mathbf{P}}_m)^T \mathbf{A} \mathbf{P}_m = \mathbf{L}_m^{-1} \mathbf{W}_m^T \mathbf{A} \mathbf{V}_m \mathbf{U}_m^{-1} = \mathbf{L}_m^{-1} \mathbf{T}_m \mathbf{U}_m^{-1} = \mathbf{I}_m \mathbf{D}_m, \quad (5.13)$$

где $\mathbf{D}_m$ есть диагональная матрица с элементами

$$d_{ii} = (v_i, w_i).$$

Если обозначить образующие $\mathbf{P}_m$ столбцы векторами p_k , а столбцы $\tilde{\mathbf{P}}_m$ — векторами $\tilde{p}_k$, $k = \overline{1, m}$, то (5.13) означает, что

$$i \neq j \implies \tilde{p}_i^T \mathbf{A} p_j = (\mathbf{A} p_i, \tilde{p}_j) = 0. \quad (5.14)$$

Определение 5.4.1 Системы векторов $\{\tilde{p}_i\}_{i=1}^m$ и $\{p_i\}_{i=1}^m$, удовлетворяющие условию (5.14), называются *A*-бисопряженными или, если не возникает неоднозначности, просто бисопряженными.

Очевидно, что бисопряженность эквивалентна биортогональности относительно скалярного *A*-произведения, а потому для нахождения векторов $\tilde{p}$ и p вместо соотношений (5.10) и (5.12) можно пользоваться биортогонализацией Ланцоша (4.22)–(4.25) с использованием в ней вместо скалярных произведений (2) скалярных *A*-произведений (3).

Остается заметить, что присутствующее в (5.11) выражение $\mathbf{L}_m^{-1}\mathbf{e}_1$ представляет собой первый столбец матрицы $\mathbf{L}_m^{-1}$, элементы которого могут быть найдены по формуле (A.1). Нетрудно видеть, что эти элементы связаны рекуррентным соотношением

$$[\mathbf{L}_m^{-1}]_{i+1,1} = -\gamma_{i+1}[\mathbf{L}_m^{-1}]_{i,1}.$$

На рис.5.4 приведен алгоритм, построенный в соответствии с описанными соображениями. Такая вычислительная схема носит название *метода бисопряженных градиентов*⁹ или *BCG*¹⁰.

В качестве вектора $\tilde{r}_0$ на практике обычно выбирается $\tilde{r}_0 = r_0$ или $\tilde{r}_0 = \mathbf{e}_k$, где индекс k таков, что $[r_0]_k \neq 0$. Как и двойственный метод Ланцоша, BCG может быть легко приспособлен для решения системы $\mathbf{A}^T z = \tilde{b}$ одновременно с (1).

⁹Объяснение присутствия в названии слова «градиент» см. в §6.2.

¹⁰Bi-conjugate Gradients. Иногда также употребляется аббревиатура BiCG.

Глава 6

Симметричный случай

Все ранее описанные методы при построении не опирались на какие-либо предположения о симметричности матрицы A , а потому без специальных изменений могут применены к симметричным СЛАУ. Однако свойство симметрии матрицы позволяет упростить вычислительную схему, что ведет к меньшим требованиям к вычислительным ресурсам при программной реализации.

Как будет показано ниже, основные упрощения могут быть внесены в процедуру построения базиса пространства Крылова.

6.1. Трехчленные соотношения для невязок. Метод Ланцоша

Заметим, что при симметричности матрицы A пространства Крылова $K_m(v_1, A)$ и $K_m(w_1, A^T)$ совпадают, если $v_1 = w_1$. Следовательно, в этом случае биортогонализация Ланцоша (§4.6) приведет к построению одинаковых систем векторов $\{v_i\}_{i=1}^m$, $\{w_i\}_{i=1}^m$. Условие биортогональности тогда сведется к

$$i \neq j \implies (v_i, v_j) = (w_i, w_j) = 0, \quad (6.1)$$

т.е. будет построен ортогональный базис пространства Крылова.

Сравнивая формулы ортогонализации Арнольди из §4.5 и формулы (4.22)–(4.25), нетрудно заметить, что они совпадают с тем лишь отличием что в (4.22)–(4.23) векторы не нормируются. Таким образом, в симметричном случае ортогонализация Арнольди сводится к трехчленному рекуррентному соотношению

$$\beta_{j+1}v_{j+1} = Av_j - \alpha_jv_j - \beta_jv_{j-1}, \quad (6.2)$$

называемому ортогонализацией Ланцоша. Коэффициенты α_i , β_i образуют трехдиагональную матрицу T_m , для которой справедливо со-

```

 $r_0 := b - \mathbf{A}x_0$ 
 $\beta := \|r_0\|_2; \quad v_1 := r_0/\beta$ 
 $\beta_1 := 0; \quad v_0 := \mathbf{0}$ 
Для  $j = \overline{1, m}$ 
     $w_j := \mathbf{A}v_j - \beta_j v_{j-1}$ 
     $\alpha_j := (w_j, v_j)$ 
     $\beta_{j+1} := \|w_j\|_2$ 
    Если  $\beta_{j+1} = 0$ 
        то  $m := j$ 
    Прервать цикл по  $j$ 
     $v_{j+1} := w_j/\beta_{j+1}$ 
увеличить  $j$ 
 $\mathbf{T}_m := \text{tridiag}(\beta_i, \alpha_i, \beta_{i+1})_{i=1}^m$ 
Найти  $y$  из трехдиагональной системы  $\mathbf{T}_m y = \beta \mathbf{e}_1$ 
 $x_m := x_0 + \sum_{i=1}^m y_i v_i$ 

```

Рис. 6.1. Метод Ланцоша

отношение (аналог (4.20))

$$\mathbf{V}_m^T \mathbf{A} \mathbf{V}_m = \mathbf{T}_m. \quad (6.3)$$

Этот же факт можно получить и другим способом. Воспользуемся соотношением (4.20). Если $\mathbf{A} = \mathbf{A}^T$, то $\mathbf{V}_m^T \mathbf{A} \mathbf{V}_m = \mathbf{V}_m^T \mathbf{A}^T \mathbf{V}_m = (\mathbf{V}_m^T \mathbf{A} \mathbf{V}_m)^T$. Тогда

$$\mathbf{H}_m^T = (\mathbf{V}_m^T \mathbf{A} \mathbf{V}_m)^T = \mathbf{V}_m^T \mathbf{A} \mathbf{V}_m = \mathbf{H}_m.$$

Очевидно, что матрица в форме Хессенберга может быть симметричной лишь тогда, когда она является трехдиагональной.

Рассмотренный в §5.1 метод FOM может быть упрощен для симметричного случая, если процедуру Арнольди в нем заменить на (6.2); для решения трехдиагональной СЛАУ при этом наиболее естественно использовать метод прогонки [4, 3]. Такая вычислительная схема, называемая *методом Ланцоша*, приведена на рис. 6.1.

6.2. CG: Метод сопряженных градиентов

Как уже отмечалось, в случае симметричной матрицы пространства Крылова $K_m(r_0, \mathbf{A})$ и $K_m(r_0, \mathbf{A}^T)$ совпадают. Это приводит к тому, что

Выбрать начальное приближение x_0
 $r_0 := b - Ax_0$
 $p_0 := r_0$
Для $j = 1, 2, \dots$
 $\alpha_j := (r_j, r_j) / (Ap_j, p_j)$
 $x_{j+1} := x_j + \alpha_j p_j$
 $r_{j+1} := r_j - \alpha_j Ap_j$
 $\beta_j := (r_{j+1}, r_{j+1}) / (r_j, r_j)$
Если $\|r_{j+1}\|_2 < \varepsilon$
то КОНЕЦ (решение достигнуто)
 $p_{j+1} := r_{j+1} + \beta_j p_j$
увеличить j

Рис. 6.2. CG

в методе бисопряженных градиентов (рис. 5.4) будут иметь место равенства

$$\tilde{r}_i \equiv r_i, \quad \tilde{p}_i \equiv p_i$$

при выборе $\tilde{r}_0 = r_0$.

Очевидное упрощение вычислительной схемы сводится к тому, что отпадает необходимость в хранении и обработке векторов $\tilde{r}_i$ и $\tilde{p}_i$; соответствующий алгоритм приведен на рис. 6.2.

Соотношение (5.14) при этом превращается в условие

$$i \neq j \implies p_i^T Ap_j = 0. \quad (6.4)$$

Определение 6.2.1 Векторы $\{p_i\}_{i=1}^m$, удовлетворяющие условию (6.4), называются **A-сопряженными** или, если не возникает неоднозначности, просто **сопряженными**.

Соответственно, метод рис. 6.2 называется **методом сопряженных градиентов** или **CG**¹.

Если в дополнение к симметричности потребовать, чтобы матрица **A** была положительно определенной, то при $p_i \neq 0$ всегда $p_i^T Ap_i \neq 0$ и сходимость метода гарантирована.

Присутствие в названии слова «градиент» обусловлено следующей причиной. Поправки к решению вычисляются вдоль векторов p_i , которые генерируются из последовательности невязок процессом их приведения к сопряженной системе².

При этом CG относится к классу тех проекционных методов, для которых $\mathcal{K} = \mathcal{L} (= K_m(r_0, A))$. Теорема 4.3.1 утверждает, что задача

¹Conjugate Gradient Method.

²Т.е. A-ортогонализацией в скалярном произведении (3).

проектирования в этом случае эквивалентна минимизации функционала $\Phi_1(x) = \|x - x_*\|_2^2$, а в примере 1 из §4.2 было показано, что направление его наискорейшего убывания $-\nabla\Phi_1$ совпадает с направлением невязки.

Метод бисопряженных градиентов исторически появился позднее, как обобщение CG на несимметричный случай, что и отразилось на его названии, так как вместо ортогонализации (6.2) в нем используется биортогонализация (4.22)–(4.25).

Замечание. В [14] показан пример подхода к выводу метода сопряженных градиентов, в котором не используется идеология LU-разложения.

6.3. О связи симметричного и несимметричного случая

Рассмотрение симметричных СЛАУ завершим табл. 6.1, кратко обобщающей рассуждения двух предыдущих параграфов.

Замечание. Метод Ланцоша и метод сопряженных градиентов эквивалентны алгебраически. Однако их реализации в конечной арифметике будут давать, вообще говоря, разные результаты.

Несимметричный случай	Симметричный случай
Ортогонализация Арнольди	Ортогонализация Ланцоша
	Трехчленное соотношение
Матрица в форме Хессенберга	Трехдиагональная матрица
Биортогонализация Ланцоша	Ортогонализация Ланцоша
LU-разложение	LL^T или LDL^T -разложение
Метод FOM	Метод Ланцоша
	Метод CG
Метод BCG с $\tilde{r}_0 = r_0$	Метод CG

Таблица 6.1. Связь симметричного и несимметричного случая

Заключение

Поскольку решение СЛАУ является стандартным этапом численного моделирования, для его реализации разработано большое количество программного обеспечения, большинство которого является бесплатным и, ведя свою историю еще от появления первых электронно-вычислительных машин, стало стандартом de-facto.

Для систем с плотными матрицами следует упомянуть такие пакеты, как BLAS, LINPACK, EISPACK и LAPACK; для СЛАУ с разреженными матрицами существует пакет SPARSKIT. Перечисленные пакеты разработаны на языке FORTRAN и хотя не реализуют собственно методы решения СЛАУ, но содержат богатый набор подпрограмм, описывающих всевозможные операции с матрицами и векторами, часто встречающиеся в схемах методов. Наличие таких библиотек и их высокая оптимизированность значительно упрощает и делает более эффективной разработку программ-решателей.

Кроме того, авторами данного пособия разработан программный пакет LinPar, содержащий готовые реализации ранее описанных методов CG, BCG, GMRES(m) и методов класса А.А.Абрамова (ориентированных на СЛАУ с плотными матрицами) для платформы Intel-DOS/Windows.

Internet-адреса, по которым доступно перечисленное программное обеспечение, приведены в табл. 6.2.

Пакеты	Internet-адреса
BLAS, LINPACK, EISPACK, LAPACK	http://www.netlib.org
SPARSKIT	http://www.cs.umn.edu/Research/arpa/SPARSKIT/
LinPar	http://www.ict.nsc.ru/linpar/

Таблица 6.2. Программное обеспечение линейной алгебры

Приложение А

LU-факторизация трехдиагональных матриц

Решение вспомогательных СЛАУ с трехдиагональными матрицами требуется в тех методах, которые основаны на трехчленных соотношениях для построения базиса (см. ВСС, §5.4 и СС, §6.2). Как оказывается, LU-разложение таких матриц осуществляется очень простым способом, что позволяет обойтись без излишнего накопления данных.

Имеет место следующая теорема:

Теорема А.0.1 Матрицы L_m и U_m , образующие LU-разложение трехдиагональной матрицы

$$T_m = \begin{pmatrix} a_1 & b_2 & & & \mathbf{0} \\ c_2 & a_2 & b_3 & & \\ & c_3 & a_3 & \ddots & \\ & & \ddots & \ddots & b_m \\ \mathbf{0} & & & c_m & a_m \end{pmatrix},$$

являются нижнедвухдиагональной и верхнедвухдиагональной соответственно.

Доказательство. Утверждение теоремы доказывается по индукции. Для матрицы третьего порядка T_3 его легко проверить непосредственно.

Предположим теперь, что для T_m утверждение справедливо и она представима в виде произведения $T_m = L_m U_m$ ниже-двухдиагональной L_m и выше-двухдиагональной U_m . Заметим, что m -ый столбец L_m по предположению индукции имеет лишь один — последний — ненулевой элемент, и то же самое справедливо для m -ой строки U_m .

Необходимо показать, что существуют такие коэффициенты α, β, γ , для которых

$$\begin{pmatrix} \mathbf{T}_m & \mathbf{e}_m b_m \\ \mathbf{e}_m^T c_m & a_m \end{pmatrix} = \begin{pmatrix} \mathbf{L}_m & \mathbf{0} \\ \gamma \mathbf{e}_m^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{U}_m & \beta \mathbf{e}_m \\ \mathbf{0} & \alpha \end{pmatrix}.$$

Выполнив умножение в правой части, получим блочно-матричное уравнение

$$\begin{pmatrix} \mathbf{T}_m & \mathbf{e}_m b_m \\ \mathbf{e}_m^T c_m & a_m \end{pmatrix} = \begin{pmatrix} \mathbf{L}_m \mathbf{U}_m & \beta \mathbf{L}_m \mathbf{e}_m \\ \gamma \mathbf{e}_m^T \mathbf{U}_m & \alpha + \gamma \beta \end{pmatrix},$$

которое сводится к трем скалярным:

$$\begin{cases} \beta [\mathbf{L}_m]_{mm} = b_m \\ \gamma [\mathbf{U}_m]_{mm} = c_m \\ \alpha + \gamma \beta = a_m \end{cases}$$

имеющим очевидное решение $\beta = b_m [\mathbf{L}_m]_{mm}^{-1}$, $\gamma = c_m [\mathbf{U}_m]_{mm}^{-1}$, $\alpha = a_m - b_m c_m / ([\mathbf{L}_m]_{mm} [\mathbf{U}_m]_{mm})$. Тем самым теорема доказана. $\square$

Замечание. В [3] доказывается и более общий результат, заключающийся в том, что ленточный характер матриц сохраняется при LU-разложении для любой ширины ленты, в том числе для лент, имеющих разную ширину вверх и вниз относительно главной диагонали. Этот факт может оказываться полезным, например, для метода ЮМ (см. §5.1).

Коэффициенты самих матриц $\mathbf{L}_m$ и $\mathbf{U}_m$ с учетом простоты их вида можно найти непосредственно из выполнения умножения $\mathbf{L}_m \mathbf{U}_m$ и сравнения результата с матрицей $\mathbf{T}_m$. Так, если отнести диагональ к матрице $\mathbf{U}$, то разложение принимает вид

$$\mathbf{L}_m = \begin{pmatrix} 1 & & & \mathbf{0} \\ \gamma_2 & 1 & & \\ & \ddots & \ddots & \\ \mathbf{0} & & \gamma_m & 1 \end{pmatrix}; \quad \mathbf{U}_m = \begin{pmatrix} \alpha_1 & \beta_2 & & \mathbf{0} \\ & \alpha_2 & \ddots & \\ & & \ddots & \beta_m \\ \mathbf{0} & & & \alpha_m \end{pmatrix}.$$

Выполнив умножение, получим

$$\mathbf{L}_m \mathbf{U}_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \alpha_1 \gamma_2 & \alpha_2 + \beta_2 \gamma_2 & \beta_3 & & \\ & \alpha_2 \gamma_3 & \alpha_3 + \beta_3 \gamma_3 & \ddots & \\ & & \ddots & \ddots & \beta_m \\ & & & \alpha_{m-1} \gamma_m & \alpha_m + \beta_m \gamma_m \end{pmatrix}.$$

Сопоставление этой матрицы с $\mathbf{T}_m$ дает следующие рекуррентные формулы:

$$\begin{aligned}\alpha_1 &= a_1; \\ \beta_i &\equiv b_i; \\ \gamma_{i+1} &= c_{i+1}/\alpha_i; \\ \alpha_{i+1} &= a_{i+1} - b_{i+1}\gamma_{i+1}.\end{aligned}$$

При построении методов ВСГ и СГ используются матрицы $\mathbf{L}_m^{-1}$ и $\mathbf{U}_m^{-1}$, которые легко находятся аналитически

$$[\mathbf{L}_m^{-1}]_{ij} = \begin{cases} 0, & j > i \\ 1, & j = i \\ (-1)^{i+j} \prod_{k=j+1}^i \gamma_k, & j < i \end{cases} \quad (\text{A.1})$$

$$[\mathbf{U}_m^{-1}]_{ij} = \begin{cases} 0, & j < i \\ 1/\alpha_i, & j = i \\ \frac{(-1)^{i+j}}{\alpha_i} \prod_{k=i+1}^j \frac{\beta_k}{\alpha_k}, & j > i \end{cases} \quad (\text{A.2})$$

Приложение В

Методы, не использующие транспонирование. TFQMR

В §5.4 был описан метод бисопряженных градиентов BCG, имеющий по сравнению с FOM и GMRES гораздо более простую вычислительную схему и меньшие требования к памяти.

Однако BCG (как и любой другой метод, использующий операции с транспонированной матрицей $\mathbf{A}$) плохо поддается реализации на многопроцессорных вычислительных системах с распределенной памятью. Все более широкое применение таких систем привело к разработке целого класса методов, в которых операция транспонирования не используется¹.

Алгебраически это может быть достигнуто за счет изменения специальным образом полинома p_m , которому (см. §4.4) удовлетворяет последовательность невязок в методах, использующих подпространства Крылова:

$$r_m = p_m(\mathbf{A})r_0 \quad (\text{B.1})$$

Так, на использовании вместо $p_m(\mathbf{A})$ полинома $p_m^2(\mathbf{A})$ основан метод CGS [13]; метод BCGSTAB вместо (B.1) использует соотношение $r_m = p_m(\mathbf{A})q_m(\mathbf{A})r_0$, где q_m — специальным образом строящийся полином, такой что произведение $p_m q_m$ не содержит нечетных степеней. Построение подобных методов, как правило, оказывается значительно более сложным.

Из числа методов, свободных от транспонирования, в настоящее время широко применяется TFQMR², алгоритм которого показан на рис. В.1.

¹Англ. transpose-free methods.

²Transpose-Free Quasi-Minimal Residuals.

```

r0 := r̃0 := b - Ax0
q0 := p-1 := d0 := 0
γ0 := η0 := 0
ρ-1 := 1, τ0 := ||r0||2
для n=0,1,2,... {
    ρn := (r̃0, rn), βn := ρn/ρn-1 un := rn + βnqn
    pn := un + βn(qn + βnpn-1), vn := Apn, σn := (r̃0, vn)
    αn := ρn/σn, qn+1 := un - αnvn, vn := αn(un + qn+1)
    rn+1 := rn - Avn
    если ||rn+1||2 < ε
        то КОНЕЦ
    для m от 2n + 1 до 2n + 2 {
        если (m + 1) четно
            то ωm+1 := √(||rn+1|| · ||rn||)
            иначе ωm+1 := ||rn+1||
        γm :=  $\frac{\omega_{m+1}}{\tau_{m-1}}$ , cm :=  $\frac{1}{\sqrt{1 + \gamma_m^2}}$ 
        τm := τm-1γmcm, ηm := cm2αn
        если m четно
            то ym := qn
            иначе ym := un
        dm := ym + γm-12  $\frac{\eta_{m-1}}{\alpha_n}$  dm-1
        xm := xm-1 + ηmdm
    }
}

```

Рис. В.1. TFQMR

Список литературы

- [1] Абаффи Й., Спедикато Э. **Математические методы для линейных и нелинейных уравнений: проекционные ABS-алгоритмы.** — М.: Мир, 1996.
- [2] Годунов С. К. **Современные аспекты линейной алгебры.** — Новосибирск: Научн. книга, 1997.
- [3] Голуб Дж., Ван Лоун Ч. **Матричные вычисления.** — М.: Мир, 1999.
- [4] Ильин В. П. **Методы неполной факторизации для решения линейных систем.** — М.: Физматлит, 1995.
- [5] Ортега Дж. **Введение в параллельные и векторные методы решения линейных систем.** — М.: Мир, 1991.
- [6] Писсанецки С. **Технология разреженных матриц.** — М.: Мир, 1988.
- [7] Фаддеев Д. К., Фаддеева В. Н. **Вычислительные методы линейной алгебры.** — М.: Физматгиз, 1963.
- [8] **Aspects of Computational Science.** — Stichting Nationale Computer Faciliteiten, The Netherlands, 1995.
- [9] Chow E., Saad Y. **ILUS: an Incomplete LU Preconditioner in Sparse Skyline Format.** — In: Int. J. for Num. Meth. in Fluids. — Vol. 25, 739–748 (1997).
- [10] Kadioglu M., Mudrick S. **On the Implementation of the GMRES(m) Method to Elliptic Equations in Meteorology.** — In: J. of Comput. Phys., 102, 348–359 (1992).
- [11] Kooper M. N. et al. **Application of the Implicitly Updated Arnoldi Method with a Complex Shift-Invert Strategy in MHD.** — In: J. of Comput. Phys., 118, 320–328 (1995).

- [12] Saad Y. **SPARSKIT: a basic tool kit for sparse matrix computations**³, 1994.
- [13] Saad Y. **Iterative Methods for Sparse Linear Systems**. — PWS Publishing Company, 1996.
- [14] Stewart G. W. **Son of Afternotes on Numerical Analysis**. — University of Maryland, 1996.
- [15] Stewart G. W. **A Survey of Matrix Algorithms. Vol.1: Basic Decompositions**. — University of Maryland, 1995.
- [Доп] Баландин М. Ю., Шурина Э. П. **Методы решения СЛАУ большой размерности: Учеб. пособие**. — Новосибирск: Изд-во НГТУ, 2000.

³Распространяется в электронной версии. Доступно по адресу <http://www.cs.umn.edu/Research/arpa/SPARSKIT/>

Содержание

Введение	2
Используемые обозначения и соглашения	3
1 Хранение и обработка разреженных матриц	5
1.1 Форматы хранения	5
1.2 Матрично-векторное умножение	8
1.3 Симметричность портрета и учет краевых условий	9
1.4 Прямой и обратный ход	10
2 Предобусловливание. Неполное LU-разложение	12
2.1 Предобусловливание	12
2.2 LU-факторизация	14
2.3 Симметричный случай	17
2.4 О программной реализации LU-предобусловливания	19
3 Классические итерационные методы и релаксация	21
3.1 Методы Якоби и Гаусса-Зейделя	21
3.2 Ускорение сходимости релаксационных методов	24
3.3 Связь с предобусловливанием	25
4 Проекционные методы. Подпространства Крылова	27
4.1 Общий подход к построению проекционных методов .	27
4.2 Случай одномерных подпространств	29
4.3 Два важных выбора подпространств	31
4.4 Подпространства Крылова	33
4.5 Базис подпространства Крылова. Ортогонализация Арнольди	34
4.6 Биортогонализация Ланцоша	37

5	Методы крыловского типа	40
5.1	FOM: Метод полной ортогонализации	40
5.2	Предобусловливание в схеме метода	43
5.3	GMRES: Метод обобщенных минимальных невязок	45
5.4	BCG: Метод бисопряженных градиентов	49
6	Симметричный случай	52
6.1	Трехчленные соотношения для невязок. Метод Ланцоша	52
6.2	CG: Метод сопряженных градиентов	53
6.3	О связи симметричного и несимметричного случая . .	55
	Заключение	55
A	LU-факторизация трехдиагональных матриц	57
B	Методы, не использующие транспонирование. TFQMR	60
	Список литературы	62