Семинар 27

Сравнение методов диагонализации билинейной формы

Пусть $\beta\colon V \times V \to F$ – симметричная билинейная форма и $Q_\beta\colon V \to F$ – соответствующая ей квадратичная форма. Пусть мы выбрали некоторый базис $e_1, \ldots, e_n \in V$. Тогда наши формы превратились в отображения $\beta\colon F^n \times F^n \to F$ по правилу $(x, y) \mapsto x^t B y$ и $Q_\beta\colon F^n \to F$ по правилу $x \mapsto x^t B x$.

Давайте предположим, что дополнительно все угловые подматрицы B_k в матрице B невырождены. В этом случае существует два способа приведения матрицы B к диагональному виду: метод Якоби и метод Лагранжа. Давайте сравним их. Для наглядности я разберу пример $n = 4$.

Метод Якоби

Метод говорит, что надо сделать замену базиса

$$\begin{aligned}
e_1' &= e_1\\
e_2' &= e_2 - \frac{\beta(e_2, e_1')}{\beta(e_1', e_1')} e_1'\\
e_3' &= e_3 - \frac{\beta(e_3, e_1')}{\beta(e_1', e_1')} e_1' - \frac{\beta(e_3, e_2')}{\beta(e_2', e_2')} e_2'\\
e_4' &= e_4 - \frac{\beta(e_4, e_1')}{\beta(e_1', e_1')} e_1' - \frac{\beta(e_4, e_2')}{\beta(e_2', e_2')} e_2' - \frac{\beta(e_4, e_3')}{\beta(e_3', e_3')} e_3'
\end{aligned}$$

Давайте проследим как будет меняться матрица билинейной формы, когда мы по очереди выбираем базис.

$$\begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & * & * \\ 0 & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & 0 & * \\ 0 & * & 0 & * \\ 0 & 0 & * & * \\ * & * & * & * \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & 0 & 0 \\ 0 & * & 0 & 0 \\ 0 & 0 & * & 0 \\ 0 & 0 & 0 & * \end{pmatrix}$$

Эти матрицы написаны в базисах (e_1', e_2, e_3, e_4), (e_1', e_2', e_3, e_4), (e_1', e_2', e_3', e_4), (e_1', e_2', e_3', e_4').

В результате работы алгоритма Якоби мы матрицу B приводим к диагональному виду D с помощью верхне треугольной матрицы C с единицами на диагонали, то есть $C^t B C = D$. Если положить $L = C^{-1}$, это означает, что у нас есть разложение $B = L^t D L$. А это есть LU-разложение, которое единственное.

Метод Лагранжа

Этот метод работает на языке замены координат. Пусть у нас

$$Q_\beta(x) = \sum_{ij} b_{ij} x_i x_j = b_{11} x_1^2 + \sum_{j=2}^n b_{1j} x_j x_1 + Q'(x_2, \ldots, x_n)$$

Тогда выделим полный квадрат из первых двух слагаемых

$$Q_\beta(x) = b_{11} \left(x_1^2 + 2 \sum_{j=2}^n \frac{b_{1j}}{2b_{11}} x_j x_1 + \left(\sum_{j=2}^n \frac{b_{1j}}{2b_{11}} x_j \right)^2 \right) - b_{11} \left(\sum_{j=2}^n \frac{b_{1j}}{2b_{11}} x_j \right)^2 + Q'(x_2, \ldots, x_n)$$

Тогда

$$Q_\beta(x) = b_{11} \left(x_1 + \sum_{j=2}^n \frac{b_{1j}}{2b_{11}} x_j \right)^2 + Q''(x_2, \ldots, x_n)$$

Сделаем замену

$$\begin{cases} \bar{x}_1 = x_1 + \sum_{j=2}^n \frac{b_{1j}}{2b_{11}} x_j \\ \bar{x}_2 = x_2 \\ \quad \ldots \\ \bar{x}_n = x_n \end{cases} \quad \text{то есть} \quad \begin{pmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \end{pmatrix} = \begin{pmatrix} 1 & \frac{b_{12}}{2b_{11}} & \cdots & \frac{b_{1n}}{2b_{11}} \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Обозначим эту матрицу перехода за C. Теперь вспомним, что после замены $Q(\bar{x}) = \bar{x}^t \bar{B} \bar{x}$. Тот факт, что $Q(\bar{x}) = b_{11}\bar{x}_1^2 + Q''(\bar{x}_2, \ldots, \bar{x}_n)$ означает, что матрица $\bar{B}$ имеет вид

$$\bar{B} = \begin{pmatrix} b_{11} & 0 & \ldots & 0 \\ 0 & * & \ldots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \ldots & * \end{pmatrix}$$

Кроме того, у нас верно равенство $B = C^t \bar{B} C$. Так как C – верхне треугольная с единицами на диагонали, все угловые миноры матрицы B совпадают с угловыми минорами матрицы $\bar{B}$. А значит в матрице $\bar{B}$ и в оставшейся в ней угловой подматрице все угловые миноры тоже не вырождены. То есть в этом случае в алгоритме Лагранжа работает только первый шаг.

Давайте поймем как меняется матрица 4 на 4 под действием алгоритма Лагранжа.

$$\begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & 0 & 0 \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & 0 & 0 \\ 0 & * & 0 & 0 \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & 0 & 0 \\ 0 & * & 0 & 0 \\ 0 & 0 & * & 0 \\ 0 & 0 & 0 & * \end{pmatrix}$$

В результате работы алгоритма Лагранжа мы матрицу B тоже приводим к диагональному виду D с помощью верхне треугольной матрицы C с единицами на диагонали, то есть $C^t B C = D$. Если положить $L = C^{-1}$, это означает, что у нас есть разложение $B = L^t D L$. А это тоже есть LU-разложение, которое единственное. А значит, в результате алгоритмы Якоби и Лагранжа приводят нашу матрицу B одним и тем же преобразованием к одному и тому же диагональному виду. Единственная разница между ними – они идут разными путями.

LU-разложение и диагонализация

Таким образом вместо диагонализации матрицы B можно рассматривать задачу нахождения LU-разложения $B = L^t D L$, где D – диагональная, а L – верхнетреугольная с единицами на диагонали.

Алгоритм получения такого разложения я описал в листочке с алгоритмами к семинару. А здесь я лишь кратко напомню, что надо попытаться привести матрицу B к верхне треугольному виду U с помощью элементарных преобразований I-типа, когда разрешено прибавлять более высокую строку с коэффициентом к более низкой. Такие преобразования как раз и соответствуют умножению на нижне треугольную матрицу R с единицами на диагонали слева. Предположим, что привести не удалось, это значит, что у нас случилось что-то вроде ситуации ниже

$$\begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & 0 & * & * \\ & & * & * & * \\ & & * & * & * \end{pmatrix}$$

В данном случае – это означает, что 3-ий угловой минор вырожден (так как умножение на нижне треугольную матрицу с единицами на диагонали с левой стороны не меняет угловые миноры). Таким образом мы можем определить выполнены ли все изначальные условия на миноры, которые нам необходимы для работы метода.

В противном случае, мы приводим матрицу B к матрице вида

$$U = \begin{pmatrix} d_1 & * & \ldots & * \\ & d_2 & \ldots & * \\ & & \ddots & \vdots \\ & & & d_n \end{pmatrix}$$

Тогда диагональная матрица D восстанавливается как диагональ матрицы U, то есть

$$D = \begin{pmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{pmatrix}$$

А матрица $L = D^{-1}U$, то есть в матрице U надо каждую строчку поделить на соответствующий диагональный элемент.

Замечание

- Если вдруг во время работы метода вы выяснили, что его угловые миноры обнуляются, то можно поступить следующим образом. Берете случайную невырожденную матрицу C и производите замену C^tBC. Тогда у новой матрицы с вероятностью единица все угловые миноры не вырождены.
- На самом деле есть более крутой способ на лету модифицировать алгоритм, чтобы не пересчитывать заново и не полагаться на случайность.

Якоби++

Давайте модифицируем алгоритм Якоби так, чтобы он работал всегда. Нам, правда, придется еще чуть-чуть ограничить себя, когда будем делать преобразования строк. Давайте теперь разрешим делать следующее: на k-ом шаге будем вычитать k-ю строку из всех более низких. Я поясню, в начале нам дана произвольная матрица

$$\begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{pmatrix}$$

На первом шаге нам разрешено вычитать только первую строку из всех более низких строк. Таким образом получаем

$$\begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & * & * & * & * \\ & * & * & * & * \\ & * & * & * & * \end{pmatrix}$$

На втором шаге спускаемся ниже и нам разрешено вычитать только вторую строку из всех более низких

$$\begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & * & * & * \\ & & * & * & * \\ & & * & * & * \end{pmatrix}$$

И так далее. Проблема будет, если мы встретим на k-ом шаге ноль в k-ой позиции на диагонали. Например на 3-ем шаге произошло следующее

$$\begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & 0 & * & * \\ & & * & * & * \\ & & * & * & * \end{pmatrix}$$

Тут два вариант: либо под этим нулем все нули, либо под ним есть ненулевой элемент. Если под ним все нули, то пропускаем и идем дальше, то есть в ситуации

$$\begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & & * & * \\ & & & * & * \\ & & & * & * \end{pmatrix}$$

Переходим к 4-ому шагу. Если же есть не нулевой элемент как на картинке

$$\begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & 0 & * & * \\ & & * & * & * \\ & & * & * & * \end{pmatrix}$$

то нужно выполнить специальный шаг. Пусть ненулевой элемент встретился в j-ой строке. Тогда прибавим j-ю строку наверх к текущей k-ой (на рисунке 3-ей) и прибавим j-й столбец к k-ому. Утверждается, что после этих манипуляций k-ый диагональный элемент будет ненулевым. Теперь надо продолжить алгоритм по предыдущему шаблону вычитая k-ю строку из более низких.

Хороший вопрос: а с чего вдруг все это работает? Нам надо объяснить, почему работает специальный шаг. Пусть мы дошли до ситуации

$$B_1 = \begin{pmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & 0 & * & * \\ & & * & * & * \\ & & * & * & * \end{pmatrix} = \begin{pmatrix} U & R \\ 0 & D \end{pmatrix} \quad \text{где} \quad D = \begin{pmatrix} 0 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

Это значит, что мы перешли от матрицы B к матрице B_1 прибавляя только 1 и 2 строку к более низким. Пусть C – матрица соответствующих преобразований, тогда $B_1 = CB$. Теперь давайте проделаем симметрично те же самые преобразования над столбцами, то есть перейдем к $B_1C^t = CBC^t$, тогда получим

$$B_1C^t = \begin{pmatrix} * & & & & \\ & * & & & \\ & & 0 & * & * \\ & & * & * & * \\ & & * & * & * \end{pmatrix} = \begin{pmatrix} S & 0 \\ 0 & D \end{pmatrix} \quad \text{где} \quad D = \begin{pmatrix} 0 & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

где матрица S диагональная, а матрица D будет той же самой, что и в матрице B_1, так как в первом и втором столбце снизу были только нули и они не задели матрицу D. Теперь мы видим, что часть сигнатуры B сидит в матрице S, а другая в матрице D. Потому мы должны определить сигнатуру D. Для этого мы можем сделать один шаг симметричного Гаусса, чтобы сделать первый элемент D ненулевым, а потом продолжить обычного Якоби. То есть прибавив некую s-ю строку D к первой и s-ый столбец D к первому, мы делаем

$$D = \begin{pmatrix} 0 & * & * \\ * & * & * \\ * & * & * \end{pmatrix} \mapsto D' = \begin{pmatrix} * & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

И потом продолжаем с этой подматрицей обычного Якоби. Заметим, что все действия, которые мы проделали с матрицей D тут равносильны описанным действиям из алгоритма, когда матрица D находится внутри B_1. И при этом в результате этих действий мы определим сигнатуру блока D, который является диагональным блоком в CBC^t.

Случай вырожденной формы

До этого мы обсуждали методы, которые работают с матрицей, у которой все угловые миноры не равны нулю. С помощью случайной замены вида $B \mapsto C^tBC$ у любой невырожденной матрицы B можно добиться того, чтобы это условие на миноры выполнялось. Однако этот метод не годится на случай вырожденной матрицы. Давайте объясню, что делать в этом случае.

Напомним, что $\beta\colon F^n \times F^n \to F$. Пусть $U \subseteq F^n$ – подпространство такое, что $F^n = U \oplus \ker\beta$. Тогда выберем базис в U и в $\ker\beta$. В этом случае матрица для билинейной формы будет иметь блочный вид

$$\begin{pmatrix} B & 0 \\ 0 & 0 \end{pmatrix}$$

где B – матрица ограничения формы β на подпространство U. При этом если мы диагонализируем матрицу B, то это равносильно диагонализации матрицы формы β на всем пространстве.

Потому можно поступить так. Найти базис ядра $\ker\beta$, дополнить его до базиса всего пространства векторами $v_1, \ldots, v_s$. Составить новую матрицу $\beta(v_i, v_j)$ – она не вырожденная и к ней применимы методы описанные выше.

Основная проблема описанного выше метода – надо пересчитывать всю матрицу целиком. Давайте вспомним, что исходная матрица B записана в стандартном базисе. Если мы нашли базис ядра $w_1, \ldots, w_k$, то его можно дополнить до базиса всего пространства с помощью стандартного базиса. Кроме того, если мы искали базис ядра с помощью алгоритма Гаусса решая систему $Bx = 0$. То дополнить до базиса всего пространства можно векторами e_i, где i – номера свободных переменных. Но подматрицу $\beta(e_i, e_j)$ считать легко, так как это и есть коэффициенты матрицы B. То есть нам надо взять в матрице B все столбцы и строки с номерами свободных переменных и соответствующая подматрица – это и есть матрица ограничения на какое-то дополнение к ядру.

Определение сигнатуры формы

Пусть теперь у нас поле $F = \mathbb{R}$. Тогда любая билинейная форма приводится не просто к диагональному виду, а к диагональному виду, в котором все диагональные элементы из множества $\{-1, 0, 1\}$. Причем количество единиц, минус единиц и нулей не зависит от диагонального вида, все что может измениться – это их порядок на диагонали. Замечу, что количество нулей совпадает с размером ядра формы, а количество единиц и минус единиц вместе – это ранг формы. Количество единиц, минус единиц и нулей называется сигнатурой формы. Обычно рассматривают невырожденные формы (как к ним сводить я объяснил выше), потому в этом случае сигнатурой называют только количество единиц и минус единиц. Часто пишут сигнатура (p, q) имея в виду, что у нас p единиц и q минус единиц. Ниже приведена таблица с терминами.

Термин	**Обозначение**	Q_β	**Индексы**
Положительно определена	$\beta > 0$ или $Q_\beta > 0$	$\forall x \neq 0 \Rightarrow Q_\beta(x) > 0$	$\#1 = n$, $\#-1 = 0$, $\#0 = 0$
Отрицательно определена	$\beta < 0$ или $Q_\beta < 0$	$\forall x \neq 0 \Rightarrow Q_\beta(x) < 0$	$\#1 = 0$, $\#-1 = n$, $\#0 = 0$
Неотрицательно определена	$\beta \geqslant 0$ или $Q_\beta \geqslant 0$	$\forall x \Rightarrow Q_\beta(x) \geqslant 0$	$\#1 = p$, $\#-1 = 0$, $\#0 = k > 0$
Неположительно определена	$\beta \leqslant 0$ или $Q_\beta \leqslant 0$	$\forall x \Rightarrow Q_\beta(x) \leqslant 0$	$\#1 = 0$, $\#-1 = q$, $\#0 = k > 0$
Неопределена		$\exists x, y : Q_\beta(x) > 0$ и $Q_\beta(y) < 0$	$\#1 = p > 0$, $\#-1 = q > 0$, $\#0 = k$