МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
им. М. В. Ломоносова

Курсовая работа

Непрерываемый блочный алгоритм сопряженных градиентов

Выполнил: студент 3-го курса
кафедры ВТМ
Д. В. Печеный
Научный руководитель: Д. А. Желтков

Москва

2024

Оглавление

Введение

Алгоритм сопряженных градиентов (CG) - эффективный метод для решения задачи

$$Ax = b$$

где $A \in \mathbb{R}^{n \times n}, x, b \in \mathbb{R}^n$ - большая симметричная положительно-определенная разреженная матрица.
Блочный алгоритм сопряженных градиентов (BCG) это обобщение алгоритма CG, использующийся для решения задачи с многими правыми частями

$$AX = B$$

где $B, X \in \mathbb{R}^{n \times s}$. Этот алгоритм находит для себя множество применений, например, при обработке сигналов.

Цель курсовой работы реализовать данный алгоритм, а также провести тесты на различных матрицах с различным количеством правых частей

Глава 1

Блочный алгоритм сопряженных градиентов

Algorithm 1

1: $R_0 = B - AX_0$
2: $P_0 = R_0\gamma_0$
3: **for** i = 1,...,n **do**
4: $\quad \alpha_i = (P_i^T A P_i)^{-1}\gamma_i^T (R_{i+1}^T R_i)$
5: $\quad X_{i+1} = X_i + P_i\alpha_i$
6: $\quad R_{i+1} = R_i - AP_i\alpha_i$
7: $\quad$ *Если сошлось то стоп*
8: $\quad \beta_i = \gamma_i^{-1}(R_{i+1}^T R_i)^{-1}(R_{i+1}^T R_{i+1})$
9: $\quad P_i = (R_{i+1} + P_i\beta_i)\gamma_{i+1}$

$A \in \mathbb{R}^{n\times n}$ - Матрица системы, симметричная, положительно-определенная

$B \in \mathbb{R}^{n\times s}$ - Матрица с правыми частями

$X \in \mathbb{R}^{n\times s}$ - Матрица неизвестных

$R \in \mathbb{R}^{n\times s}$ - Матрица остатков

$P \in \mathbb{R}^{n\times s}$ - Матрица направления поиска

$\alpha, \beta, \gamma \in \mathbb{R}^{s\times s}$ - Невырожденные матрицы коэффициентов

Алгоритм 1 это алгоритм BCG, который был предложен О Лири, для решения систем с многими правыми частями.

Матрицы α_i выбираются с целью обеспечения ортогональности между матрицами остатков R_i и матрицами направления поиска P_j, полученные на преды-

дущих итерациях. То есть за счёт выбора α_i выполнено

$$R_i P_j = P_j R_i = 0 \quad \forall j < i$$

Матрицы β_i выбираются для обеспечения A-ортогональности между направлениями поиска. То есть за счет выбора β_i выполнено

$$P_i A P_j = 0 \quad \forall j \neq i$$

Матрицы γ_i выбираются для уменьшения ошибок округления и обеспечения численной стабильности

Заметим, что матрицы P_i и R_i имеют одинаковый ранг. Это означает что при уменьшении ранга одной матрицы, ранг уменьшится у другой. Этот недостаток является серьезной проблемой, ввиду невозможности вычисления обратной матрицы в строках 4, 8 для вычисления матриц коэффициентов α_i, β_i. И вследствие этого работа алгоритма прекращается.

Уменьшение ранга может возникнуть, например, в случае, если один столбец сойдётся до истинного решения быстрее другого, и тогда в матрице остатков R_i возникнет столбец, заполненный нулями, вследствие чего и уменьшится ранг.

Глава 2

Непрерываемый блочный алгоритм сопряженных градиентов

Algorithm 2

1: $R_0 = B - AX_0$
2: $P_0 = orth(R_0)$
3: **for** i = 1,...,maxit **do**
4: $\quad Q_i = AP_i$
5: $\quad \alpha_i = (P_i^T Q_i)^{-1}(P_i^T R_i)$
6: $\quad X_{i+1} = X_i + P_i\alpha_i$
7: $\quad R_{i+1} = R_i - AP_i\alpha_i$
8: $\quad$ *Если сошлось то стоп*
9: $\quad \beta_i = -(P_i^T Q_i)^{-1}(Q_i^T R_{i+1})$
10: $\quad P_i = orth(R_{i+1} + P_i\beta_i)$

Алгоритм 2 называется непрерываемым алгоритмом BCG (BFBCG). Он позволяет избежать уменьшения ранга за счет операции $\mathrm{orth}(\cdot)$, которая извлекает ортонормированный базис. Благодаря этому, в процессе выполнения алгоритма количество столбцов матриц P, α, β может уменьшится. Также, матрица γ из алгоритма 1 становится ненужной. При достижении сходимости какого либо столбца в матрице R можно просто выкидывать столбец который сошелся, а затем уменьшать параметр s. Утверждения $R_i P_j = P_j R_i = 0 \quad \forall j < i$ и $P_i A P_j = 0 \quad \forall j \neq i$ остаются верными.

Глава 3

Численные эксперименты

Самая дорогая операция здесь - умножение матрицы размера $n \times n$ на матрицу размера $n \times s$ в строке 4, поэтому сложность алгоритма равна $O(n^2 s)$. Ниже представлены некоторые эксперименты.

n (порядок A)	s (кол-во правых частей)	Кол-во итераций	Время выполнения (с)
100	10	19	0.001
1000	10	15	0.004
1000	20	22	0.009
1000	40	18	0.011
1000	80	14	0.025
1000	200	13	0.108
1000	600	7	0.346
1000	999	2	0.355
1000	1000	1	0.062
2000	1000	6	2.395
4000	1000	13	5.956
10000	1000	15	19.884
20000	1000	16	52.034

По наблюдениям видно, что при возрастании количества правых частей при фиксированном порядке матрицы, время выполнения программы растет, при этом количество итераций, необходимых для достижения нужной точности, убывает. При фиксированном количестве правых частей, увеличивая порядок матрицы A, возрастает как время выполнения, так и количество итераций.

Заключение

Алгоритм крутой всем советую

Литература

[1] O'Leary, D.P.: The block conjugate gradient algorithm and related methods. Linear Algebra Appl. 29, 293–322 (1980)

[2] Hao Ji, Yaohang Li: A breakdown-free block conjugate gradient method. BIT Numer Math (2017) 57:379–403