Programação para Alto Desempenho
Autovalor e autovetor

2021

1 O problema

Características importantes de matrizes, úteis em diversas aplicações, são seus autovalores e autovetores. Dada uma matriz $\mathbf{A}$ (quadrada, $N \times N$), se

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v},$$

onde λ é um escalar e $\mathbf{v}$ é um vetor, então dizemos que $\mathbf{v}$ é um *autovetor* da matriz $\mathbf{A}$, com *autovalor* λ. Neste caso, é fácil ver que qualquer vetor da forma $a\mathbf{v}$ onde a é um escalar é também um autovetor de $\mathbf{A}$ com o mesmo autovalor λ.[1] Por isso, em geral nos interessamos apenas pelos autovetores normalizados, isto é, aqueles para os quais $\|\mathbf{v}\| = 1$.[2] Uma matriz quadrada $N \times N$ tem N autovalores e correspondentes autovetores normalizados (os autovalores podem ser múltiplos, isto é, o mesmo autovalor para diversos autovetores). Vamos aqui considerar que os autovalores estão ordenados:

$$|\lambda_1| \geq |\lambda_2| \geq \cdots |\lambda_N|.$$

Existem métodos que permitem o cálculo preciso de todos os autovalores e autovetores de uma matriz dada. Infelizmente, o custo computacional é alto, o que torna inviável o emprego desses métodos para matrizes grandes. Mas muitas vezes estamos interessados apenas no maior autovalor e correspondente autovetor da matriz. Estes podem ser calculados aproximadamente de forma eficiente por métodos iterativos, como o método de Lanczos. Neste trabalho, iremos calcular o autovalor de maior valor absoluto e o correspondente autovetor de uma matriz dada, mas não usaremos o método de Lanczos, mas sim um método mais simples, denominado *power iteration*.

Seja a matriz $\mathbf{A}$, quadrada e $N \times N$, e λ_1 seu autovalor com maior valor absoluto, e $\mathbf{v}_1$ o correspondente autovetor (suposto normalizado). Portanto,

$$\mathbf{A}\mathbf{v}_1 = \lambda_1\mathbf{v}_1.$$

Este método se baseia no fato[3] de que os autovetores de uma matriz $N \times N$ formam uma base de $\mathbb{R}^N$, e portanto um vetor qualquer $\mathbf{x}$ em $\mathbb{R}^N$ pode ser expresso como uma combinação linear dos autovetores,

$$\mathbf{x} = \sum_{j=1}^{N} \alpha_j \mathbf{v}_j,$$

onde os α_j são escalares. Como ao multiplicar pela matriz cada autovetor é projetado na sua própria direção, com módulo alterado pelo correspondente autovalor, ao multiplicarmos um vetor qualquer por $\mathbf{A}$ este terá sua componente na direção do autovetor associado ao maior autovalor enfatizada, pois

$$\mathbf{A}\mathbf{x} = \mathbf{A}\sum_{j=1}^{N} \alpha_j \mathbf{v}_j = \sum_{j=1}^{N} \alpha_j \mathbf{A}\mathbf{v}_j = \sum_{j=1}^{N} \alpha_j \lambda_j \mathbf{v}_j,$$

[1] $\mathbf{A}a\mathbf{v} = a\mathbf{A}\mathbf{v} = a\lambda\mathbf{v} = \lambda a\mathbf{v}$.

[2] Vamos usar aqui a norma 2, que é a raiz quadrada da soma dos quadrados das componentes do vetor. Lembre-se que existem dois vetores normalizados na mesma direção, $\mathbf{v}$ e $-\mathbf{v}$.

[3] A prova pode ser encontrada em textos padrão de álgebra linear.

e como λ_1 é o maior autovalor, o componente do vetor $\mathbf{x}$ na direção de $\mathbf{v}_1$ cresce em magnitude em relação aos outros componentes. Se esse processo for realizado iterativamente (com continua normalização dos vetores resultantes) por um número apropriado de passos, o vetor resultante estará aproximadamente na direção de $\mathbf{v}_1$.

Temos então um algoritmo para encontrar o maior autovalor (em valor absoluto) e o correspondente autovetor, como segue:

1. Primeiro sorteamos um vetor aleatório de N elementos $\mathbf{r}_0$.
2. Calculamos $b_0 = \|\mathbf{r}_0\|$ e $\mathbf{x}_0 = \mathbf{r}_0/b_0$ (normalização).
3. Repetimos os cálculos: $\mathbf{r}_{i+1} = \mathbf{A}\mathbf{x}_i$, $b_{i+1} = \|\mathbf{r}_{i+1}\|$, $\mathbf{x}_{i+1} = \mathbf{r}_{i+1}/b_{i+1}$.

Depois de um certo número de repetições, teremos que $\mathbf{x}_i$ converge para $\mathbf{v}_1$ e b_i converge para $|\lambda_1|$. Considere agora os seguinte pontos:

- O método descrito funciona se duas condições forem satisfeitas:
    1. $|\lambda_1|$ tem que ser suficientemente maior do que o módulo de qualquer outro autovalor. Se isso não ocorrer, dependendo do vetor aleatório inicial e da diferença entre valores absolutos dos dois maiores autovalores λ_1 e λ_2, podemos ter uma convergência errônea (para um outro autovetor com autovalor grande, ao invés de $\mathbf{v}_1$) ou muito lenta. Um método conhecido como *subspace iteration* permite lidar com esse caso. Aqui vamos apenas ignorar esse problema, considerando que para nossas necessidades basta achar um autovalor grande e seu correspondente autovetor.
    2. O vetor aleatório inicial $\mathbf{r}_0$ precisa ter uma componente não-nula na direção de $\mathbf{v}_1$, para que essa componente possa ser enfatizada pelo processo iterativo. Se a componente na direção de $\mathbf{v}_1$ for nula, então o processo irá convergir para outro autovetor com autovalor grande para o qual o vetor inicial tem componente não-nula. A probabilidade de um vetor aleatório ter uma componente nula na direção de $\mathbf{v}_1$ é em princípio zero. Entretanto, devido a aproximações numéricas, na prática existe uma probabilidade não-nula disso acontecer. Como essa probabilidade é sempre pequena, vamos apenas ignorar esse problema neste trabalho.
- Para evitar lidar com números complexos, as matrizes estudadas serão todas simétricas de elementos reais, caso em que os autovalores e as componentes dos autovetores são reais.
- Para calcular o valor de $\mathbf{r}_0$, iniciamos cada um dos seus elementos com um número aleatório entre -1 e 1.
- O método acima dá apenas o valor absoluto de λ_1. Para encontrar o sinal, podemos comparar o sinal de uma componente de $\mathbf{x}_i$ com a mesma componente de $\mathbf{A}\mathbf{x}_i$. Se houver mudança de sinal, o autovalor é negativo. Isso só precisa ser feito no final do processo iterativo. O melhor é usar uma componente que seja suficientemente diferente de zero, por exemplo a componente com o maior valor absoluto (para evitar problemas de arredondamento numérico).
- Precisamos de um critério de término das iterações. Como b_i está convergindo para $|\lambda_1|$, podemos terminar as iterações assim que a diferença de valores entre b_i e b_{i+1} for suficientemente pequena (em comparação com uma precisão desejada). Usaremos a diferença relativa $|b_{i+1} - b_i|/b_{i+1}$. No entanto, para evitar problemas nas primeiras iterações (detecção de uma convergência que ainda não existe), forçaremos a execução de pelo menos 3 iterações.
- Como saber se o resultado (após a convergência) faz sentido? Para isso, verificamos se $\|\mathbf{A}\mathbf{v}_1 - \lambda_1\mathbf{v}_1\|/|\lambda_1|$ é da ordem da precisão especificada.

2 Descrição do programa

Dada uma matriz **A** como a descrita acima, o programa irá calcular λ_1 e $\mathbf{v}_1$ como definidos acima. Consideramos que **A** é uma matriz simétrica **esparsa** de grandes dimensões (N é muito grande).

As entradas para o programa serão uma descrição da matriz, a precisão a ser usada para teste de convergência e o nome do arquivo onde escrever os resultados. A descrição da matriz será fornecida em um arquivo, cujo nome é o primeiro parâmetro na linha de comando. O segundo parâmetro na linha de comando é o valor da precisão (um valor de ponto flutuante pequeno e estritamente positivo). O nome do arquivo de saída é o terceiro parâmetro da linha de comando.

O arquivo que descreve a matriz tem o seguinte formato: Na primeira linha, ele tem o valor de N. Na segunda linha, ele tem o número M de elementos que serão fornecidos (igual ao número de linhas seguintes). Cada uma das M linhas seguintes fornecerá três valores separados por espaços em branco: dois inteiros i e j seguidos de um número y de ponto flutuante (usar `double` ao fazer a leitura e nos cálculos). Isso indica que os elementos a_{ij} e a_{ji} da matriz são diferentes de zero e têm o valor y: $a_{ij} = a_{ji} = y$. Cada par i, j será fornecido apenas uma vez, ou como i seguido de j ou como j seguido de i. Teremos sempre $y > 0$. Todos os elementos da matriz que não tiverem valores fornecidos no arquivo terão valor assumido zero.

A saída do programa consiste em um arquivo, cujo nome é o terceiro parâmetro da linha de comando, e que contém em sua primeira linha o autovalor calculado, na segunda linha o tamanho da matriz N e nas N linhas seguintes os valores das componentes do autovetor, em ordem. Além de gerar esse arquivo, o programa também deve mostrar na tela o valor de $\|\mathbf{A}\mathbf{v}_1 - \lambda_1\mathbf{v}_1\|/|\lambda_1|$ para as aproximações de autovalor e autovetor calculadas e o tempo tomado para a execução de todos os cálculos (sem considerar tempos de leitura e escrita).

Para facilitar o trabalho de correção, coloque todo o código fonte em um único arquivo.

Entregar um arquivo (`tar.gz` ou `zip`) com o código fonte e um relatório (no formato PDF) que descreve as decisões de implementação e faz uma análise de desempenho do programa para os diversos arquivos de teste.

Dica O fato das matrizes serem grandes e esparsas (a maioria dos elementos da matriz é zero), faz com que o uso de representação densa de matrizes, onde guardamos todos os elementos da matriz na memória, inviável e ineficiente. Portanto, um ponto importante no desenvolvimento do código é encontrar uma representação esparsa da matriz que seja eficiente para as suas necessidades específicas.