TP-Projet 2
Subspace iteration methods

Ayoub Najmeddine
Lounes Naji

Département Sciences Du Numérique - Première année
2020-2021

Table des matières

1 Limitation de la méthode de la puissance itérée

Question 1 :

Type de la matrice	Taille de la matrice	Temps de calcul par POWER METHODE en (s)	Temps de calcul par la fonction EIG en (s)
1	200	1.630e+00	0.000e+00
1	500	2.750e+00	6.000e-02
2	200	2.000e-02	0.000e+00
2	500	6.000e-02	4.000e-02
3	200	5.000e-02	2.000e-02
3	500	1.300e-01	6.000e-02
4	200	1.280e+00	2.000e-02
1	500	2.850e+00	4.000e-02

Pour différents types et tailles de matrice, on constate que la fonction *eig* de matlab est plus efficae que la méthode de la puissance itérée en terme de rapidité de calcul.

Question 2 :

```
function [ W, V, n_ev, itv , flag ] = power_v12( A, m, percentage , eps , maxit )

    n = size(A,1);

    % initialisation des resultats
    W = [];
    V = [];
    it = [];
    n_ev = 0;

    % trace de A
    tA = trace(A);

    % somme des valeurs propres
    eig_sum = 0.0;

    % indicateur de la convergence (pourcentage atteint)
    convg = 0;

    % numero du couple propre courant
    k = 0;

    while (~convg && k < m)

        k = k + 1;

        % methode de la puissance iteree
        v = randn(n,1);
        v = mgs(v);
        z = A*v;
        beta = v'*z;
```

```
        % conv = || beta * v - A*v|||/|beta| < eps
        % voir section 2.1.2 du sujet
        norme = norm(beta*v - z, 2)/norm(beta,2);
        nb_it = 1;

        while(norme > eps && nb_it < maxit)
          y = A*v;
          v = y / norm(y,2);
          beta_old = beta;
          beta = v'*y;
          norme = norm(beta - beta_old)/abs(beta_old);
          nb_it = nb_it + 1;
        end

        % la calcul de ce couple propre a echoue => echec global
        if(nb_it == maxit)
          flag = -3;
          % on sort de la fonction en plein milieu
          % ce n'est pas tr  s bien structure
          % pardon aux enseignants de PIM
          return;
        end

        % on sauvegarde le couple propre
        W(k) = beta;
        V(:,k) = v;
        itv(k) = nb_it;
        eig_sum = eig_sum + beta;

        % deflation
        A = A - beta* (v*v');

        % est-ce qu'on a atteint le pourcentage
        convg = eig_sum/tA > percentage;

    end

    % on a atteint le pourcentage
    if (convg)
      n_ev = k;
      flag = 0;
      W = W';
    else
      % ce n'est pas le cas
      flag = 1;
    end

end
```

Question 3 :

La méthode de Déflation calcul à chaque itération la valeur propre la plus dominante est son vecteur propre associé, ce qui est trés couteux.

2 Extension de la méthode de déflation pour calculer les couples propres dominants

2.1 subspace iter v0 : une méthode basique pour calculer un couple propre dominant

2.1.1 Orthonormalisation

2.1.2 Critère d'arrêt

2.1.3 Le quotient de Rayleigh

Question 5 :

La matrice H est une matrice carré de taille $m \times m$, Le choix de la décomposition spectrale de A est due a sa taille $(n < m)$. Car cette méthode a pour but d'optimiser l'espace mémoire utilisé et donc améliorer la méthode de puissance itérée.

2.2 subspace iter v1 : version améliorée utilisant le quotient de Rayleigh

2.3 subspace iter v2 et subspace iter v3 : vers une résolution plus efficace

2.3.1 Approche par blocs (subspace iter v2)

Question 8 :

On peut montrer par récurrence que le coût en terme de flops du calcul de $A^p = (p-1) \cdot n^3$. Pour la matrice $A^p \cdot V$ le coût en terme de flops du calcul est $(p-1) \cdot n^3 + n^2 \cdot m$. Pour réduire le coût en terme de flops du calcul de $A^p \cdot V$ on calcule $Y = A \cdot V$ puis $V = Y$ et on la répète p fois. Comme A est de taille $n \times n$ et $A \cdot V$ et de taille $n \times m$, ce qui donne un coût égal à $p \cdot n^2 \cdot m$. Avec cette méthode on aura besoin de moins d'espace de stockage.

2.3.2 Méthode de déflation (subspace iter v3)

Question 9 :

(Cf **subspace_iter_v2.m**)

3 Éxpérimentation numérique

Question 10 :

L'augmentation de la valeur de p permet d'accélérer le sous programme, et la convergence des petites valeurs propres, en revanche p ne devrait pas être trop grand pour ne pas ralentir le calcul de A^p.

Question 11 :

Pour **subspace_iter_v1**, on effectue la projection de *Rayleigh* sur tous les vecteurs de V. sans tenir compte de la convergence de ces vecteurs. Ainsi, les vecteurs calculés au début ne sont pas aussi précis que ceux calculés plus tard.

Question 14 :

La différence entre les 4 types de matrices réside dans leurs spectres. En effet, les valeurs propres des matrices de type 1 et 4 sont uniformément distribuées, celles des matrices de type 3 sont de plus en plus espacées et celles des matrices de type 2 sont aléatoirement distribuées.