Analisi degli errori per un modello di code fluide markoviane

8 Giugno 2016

Questi appunti vanno visti solo come un'appendice e un aiuto per seguire l'omonimo seminario.

1 La funzione matriciale segno(A)

Data una matrice $A \in \mathbb{R}^{M\times M}$ senza autovalori immaginari puri, siano n i suoi autovalori con parte reale negativa e p quelli con parte reale positiva; sia infine S la matrice di similitudine di Jordan. Allora

$$\text{segno}(A) := S^{-1}\begin{bmatrix} -I_n & 0 \\ 0 & I_p \end{bmatrix} S. \tag{1}$$

è la funzione segno(A). Si può dimostrare che, se A ed E sono invertibili, il metodo

$$\begin{cases} Z_0 & = A \\ Z_{k+1} & = \frac{1}{2}(Z_k + EZ_k^{-1}E) \end{cases}$$

converge quadraticamente a segno(AE^{-1}). Nel listing 1 possiamo vedere il codice utilizzato.

2 Spectral divide and conquer

Abbiamo visto che lo snodo centrale per il problema di capacità infinita è trovare due matrici ortonormali Q, Z tali che

$$Q^T(\lambda E - A)Z = \begin{bmatrix} \lambda E_{11} - A_{11} & \lambda E_{12} - A_{12} \\ 0 & \lambda E_{22} - A_{22} \end{bmatrix} \tag{2}$$

e che gli spettri di $\lambda E_{11} - A_{11} \subset$ e di $\lambda E_{22} - A_{22}$ siano contenuti rispettivamente in $\mathbb{C}^{+0}$ e $\mathbb{C}^{-}$.

Per ottenere Q e Z aggiorniamo in primo luogo $\bar{A} = A + Ee\pi E/\pi Ee$, cosicché $\bar{A}$ sia invertibile. Sia $Z_\infty = \text{segno}(AE^{-1})$. Consideriamo le decomposizioni QR rank-revealing $(E - Z_\infty)^T\Pi = [Z_1, Z_2]R$ e $(\bar{A}Z_2, EZ_2)\Pi = QR$: allora le matrici ortonormali Q, $Z := [Z_2, Z_1]$ risolvono l'equazione (2). Nel listing 2 troviamo il codice relativo.

Listing 1: Calcolare la funzione segno.

```
function newtonSignIt(A,E;maxit=100)
    n = size(A,1);
    tol_scale = 1e-2;
    tol_cvg = 1e-15;
    X1 = A;
    sentinel = true;
    delta1 = Inf;
    iterazioni = 0;
    while sentinel
        iterazioni += 1;
        Y1 = inv(X1);
        X2 = (X1 + E*Y1*E)/2;
        delta2 = norm(X2 - X1)/norm(X2);
        if norm(X2 - X1,1) < tol_cvg*norm(X2,1)
            sentinel = false;
        elseif iterazioni > maxit
            sentinel = false;
            warn("Il risultato non e'
                una buona approssimazione");
        end
        delta1 = delta2;
        X1 = X2;
    end
    return X1
end
```

3 Additive decomposition

Siano A, E matrici con le consuete proprietà. Se la capacità del buffer è finita, dobbiamo trovare due matrici U, V tali che

$$U^{-1}EV = \begin{bmatrix} b & 0 & 0 \\ 0 & E_{11} & 0 \\ 0 & 0 & E_{22} \end{bmatrix}, \quad U^{-1}AV = \begin{bmatrix} 0 & 0 & 0 \\ 0 & A_{11} & 0 \\ 0 & 0 & A_{22} \end{bmatrix}. \tag{3}$$

con i blocchi di taglia rispettivamente $1, n, p$ e con gli spettri di $\lambda E_{11} - A_{11}$ e $\lambda E_{22} - A_{22}$ contenuti in $\mathbb{C}^-$ e $\mathbb{C}^+$.
Come passo intermedio, cerchiamo due matrici X, Y tali che

$$Y^{-1}EX = \begin{bmatrix} b & 0 \\ 0 & E_2 \end{bmatrix}, \qquad Y^{-1}AX = \begin{bmatrix} 0 & 0 \\ 0 & A_2 \end{bmatrix} \tag{4}$$

Listing 2: Risolvere il problema SDC.

```
function deflatingSubspace(A,E)
    n = size(A,1);
    #compute M_
    M_ = 0;
    for i = 1:n
        if E[i,i] < 0
            M_ += 1;
        end
    end
    #invariant vector
    pv = pvlu(A + eye(n));
    #rank-one update
    e = ones(n);
    A1 = A + E*e*pv'*E/dot(pv,E*e);
    X = newtonSignIt(A1,E);

    #calcolo di Z
    Z3 = qr((E - X)', Val{true})[1];
    Z1 = Z3[:,1:end-M_];
    Z2 = Z3[:,end-M_+1:end];

    #calcolo di Q
    Q = qr(hcat(A1*Z2, E*Z2), Val{true},
       thin=false)[1];
    Z = hcat(Z2,Z1);
    return Q, Z
end
```

Siano ora Q, Z due matrici ortonormali tali che $Ze_1 = e$ e $Qe_1 = Ee$. È noto che

$$Q^T E Z = \begin{bmatrix} E_{11} & E_{12} \\ 0 & E_{22} \end{bmatrix}, \quad Q^T A Z = \begin{bmatrix} 0 & A_{12} \\ 0 & A_{22} \end{bmatrix},$$

con E_{11} scalare. Se le matrici X_1 e Y_1 risolvono $X_1 A_{22} = A_{12}$ e $E_{11} Y_1 = X_1 E_{22} - E_{12}$, allora

$$X := Z \begin{bmatrix} 1 & Y_1 \\ 0 & I \end{bmatrix}, \qquad Y := Q \begin{bmatrix} 1 & X_1 \\ 0 & I \end{bmatrix}$$

risolvono (4). Chiamiamo $Z_\infty = \text{segno}(A_2 E_2^{-1})$. Consideriamo la decomposizione QR rank-revealing $(E_2 + Z_\infty)\Pi = [Z_1, Z_2]R$, dove Z_2 ha n colonne; sia ora $[A_2 Z_2, E_2 Z_2]\Pi = [Q_1, Q_2]R$ l'altra decomposizione QR, con Q_1 avente n

colonne. Ripetiamo gli stessi passi su $E_2 - Z_\infty$, scegliendo questa volta p colonne e ottenendo le $\bar{Q}_1$ e $\bar{Z}_2$. Si può mostrare infine che

$$U := Y \begin{bmatrix} 1 & 0 \\ 0 & [Q_1, \bar{Q}_1] \end{bmatrix}, \qquad V := X \begin{bmatrix} 1 & 0 \\ 0 & [Z_2, \bar{Z}_2] \end{bmatrix} \tag{5}$$

risolvono (3). Troviamo il codice nel listing 3. Si noti che ritorniamo U^{-1} e che invertiamo algebricamente Y, tuttavia la maggior difficoltà numerica riguarda $[Q_1, \bar{Q}_1]$.

Listing 3: Risolvere il problema AD

```
function generalizedNewtonFin(A, E, n, p)

    M = size(A,1);
    aux1 = eye(M); aux1[:,1] = ones(M);
    ee = diag(E);
    aux2 = eye(M); aux2[:,1] = ee;
    Z = qr(aux1)[1]; Q = qr(aux2)[1];
    aux3 = Q'*E*Z; aux4 = Q'*A*Z;

    E12 = aux3[1,2:end]; E22 = aux3[2:end,2:end];
    A12 = aux4[1,2:end]; A22 = aux4[2:end,2:end];
    #X1*A22 = A12,  E11 = aux3[1,1],
    #-E11*Y1 = E12 - X1*E22
    X1 = (A22'\A12');
    Y1 = (E22'*X1 - E12')/aux3[1,1];
    X  = [1 Y1';zeros(M-1) eye(M-1)];
    Y  = [1 X1';zeros(M-1) eye(M-1)];
    #Invertiamo Y
    Yinv = [1 -X1'; zeros(M-1) eye(M-1)];
    aux5 = Yinv*aux3*X;  aux6 = Yinv*aux4*X;
    E2 = aux5[2:end,2:end]; A2 = aux6[2:end,2:end];
    #Ora calcoliamo la matrice segno di x*E2 - A2.
    Zinf = newtonSignIt(A2,E2);

    ZZ  = qr((E2 + Zinf)', Val{true})[1];
    ZZm = ZZ[:,end-n+1:end];
    QQ  = qr([A2*ZZm E2*ZZm], Val{true},
          thin=false)[1];
    QQm = QQ[:, 1:n];

    ZZ  = qr((E2 - Zinf)', Val{true})[1];
    ZZp = ZZ[:,end-p+1:end];
    QQ  = qr(hcat(A2*ZZp, E2*ZZp), Val{true},
          thin=false)[1];
    QQp = QQ[:, 1:p];

    Uinv =[[1 zeros(n+p)'];zeros(M-1)
          inv([QQm QQp])]*Yinv*Q';
    V = Z*X*[[1 zeros(n+p)'];zeros(M-1) [ZZm ZZp]];
    return Uinv, V
end
```