Non-stationary iterative methods for solving macroeconomic numeric models

Conf.dr. Bogdan OANCEA
Universitatea Artifex din Bucureşti
e-mail: oanceab@ie.ase.ro

Macroeconometric modeling was influenced by the development of new and efficient computational techniques. Rational Expectations models, a particular class of macroeconometric models, give raise to very large systems of equations, the solution of which requires heavy computations. Therefore, such models are an interesting testing ground for the numerical methods addressed in this research. The most difficult problem is to obtain the solution of the linear system that arises during the Newton step. As an alternative to the direct methods, we propose non-stationary iterative methods, also called Krylov methods, to solve these models. Numerical experiments conducted by authors confirm the interesting features of these methods: low computational complexity and storage requirements.
Keywords: non-stationary methods, rational expectation models.

Introducere

Modelele macroeconometrice de previziune reprezintă o clasă specială de modele matematice care generează sisteme de ecuaţii liniare sau neliniare de foarte mari dimensiuni. Un caz special de modele macroeconometrice îl constituie modelele bazate pe teoria aşteptărilor raţionale introdusă de Lucas [5].
Utilizând notaţia convenţională [3], un model dinamic cu aşteptări raţionale se poate exprima astfel:

$$f_i(y_t, y_{t-1}, \dots, y_{t-r}, y_{t+1|t-1}, \dots, y_{t+h|t-1}, z_t)=0 \quad i = 1,\dots n \qquad (1)$$

unde $y_{t+j|t-1}$ valoarea aşteptată pentru y_{t+j} condiţionat de informaţia disponibilă la sfârşitul perioadei *t-1* iar z_t reprezintă variabila exogenă. Pentru o soluţie consistentă, valoarea viitoare aşteptată $y_{t+j|t-1}$ trebuie să coincidă cu previziunea pentru perioada următoare atunci când rezolvăm sistemul condiţionat de informaţiile disponibile la sfârşitul perioadei *t-1*. Aceste valori aşteptate sunt astfel legate în timp iar rezolvarea modelului pentru fiecare y_t condiţionat de o valoare iniţială necesită cunoaşterea fiecărui $y_{t+j|0}$ pentru $j= 1,2, \dots T-t$ şi o valoare de final $y_{T+j|0}$ $j = 1,2, \dots, h$.
Metoda clasică de rezolvare a unui astfel de model este metoda căii extinse propuse de Fair and Taylor [3]. Această metodă utilizează iteraţii de tip Gauss-Seidel pentru fiecare perioadă de timp, pentru un orizont de previziune dat. Avantajul metodei este acela că are o complexitate redusă dar prezintă dezavantajul unei rate a convergenţei reduse.
O altă abordare în rezolvarea modelelor cu aşteptări raţionale este să scriem ecuaţiile (1) pentru fiecare perioadă de timp *t*, caz în care va rezulta un sistem neliniar de mari dimensiuni. Pentru astfel de sisteme propunem aplicarea metodei Newton de rezolvare a sistemelor neliniare împreună cu tehnici Krîlov de rezolvare a sistemelor liniare. Avantajul major al metodei Newton constă în convergenţa sa pătratică. Dezavantajul metodei constă în rezolvarea unui sistem liniar la fiecare pas al algoritmului. De asemenea, tot la fiecare iteraţie trebuie calculată şi matricea Jacobinană.
Metoda Newton aplicată la modelul (1) constă în următorul algoritm:

```
Fie solutia initială y(0)
For k = 0,1,2, ... until convergence
     Calculează b(k) = - h(y(k),z)
     Calculează J(k) = ∂h(y(k),z)/∂y'
     Rezolvă J(k)s(k) = b(k)
     y(k+1) = y(k) + s(k)
end
```

Fig.1. Algoritmul Newton pentru sisteme neliniare

În situaţia în care sistemul liniar *J(k)s(k) = b(k)* este de mari dimensiuni, situaţie care apare des în practică, metodele directe de rezolvare a sistemelor liniare nu sunt eficiente datorită complexităţii ridicate şi a cerinţelor de memorie foarte mari.

Ca o alternativă la metodele directe se pot utiliza metodele iterative care calculează doar o aproximaţie a soluţiei, dar acest lucru nu influenţează convergenţa metodei Newton. O problemă practică dificilă este alegerea unui nivel de precizie al soluţiei care să garanteze convergenţa rapidă a metodei Newton. Vom defini $r(k) = b(k) - J(k)s(k)$ ca fiind eroarea soluţiei aproximative a sistemului liniar la iteraţia Newton cu numărul *k*. Se poate arăta [6] că metoda Newton este convergentă local dacă $\|r(k)\|/\|b(k)\|$ este un şir de valori mai mici decât 1.

Metode iterative de tip Krîlov

Spre deosebire de metodele iterative clasice, precum Gauss-Seidel sau Jacobi, metodele de tip Krîlov utilizează informaţii care se modifică de la o iteraţie la alta. Pentru un sistem liniar de forma *Ax = b* metodele Krîlov calculează aproximaţia *x(i)* în felul următor [9] :

$$x(i)=x(i-1)+d(i) \qquad i = 1,2,...$$

Pentru a calcula vectorul de actualizare *d(i)* sunt necesare doar produse scalare de vectori, operaţii de tip *saxpy* şi produse matrice-vector, ceea ce înseamnă că aceste metode au o complexitate scăzută şi devin atractive în special pentru sisteme de mari dimensiuni.

Una dintre cele mai cunoscute metode de tip Krîlov este metoda *gradientului conjugat* care se aplică pentru sisteme cu matrice simetrice pozitiv definite. Ideea de bază a acestei metode este aceea de a actualiza *x(i)* în aşa fel încât să se asigure o descreştere maximă pentru funcţia obiectiv $\frac{1}{2}x'Ax - x'b$ menţinând în acelaşi timp vectorii direcţie *d(i)* A-ortogonali. Metoda gradientului conjugat are o complexitate redusă, ea poate fi implementată doar cu o singură înmulţire matrice-vector la fiecare iteraţie. În aritmetică exactă metoda gradientului conjugat calculează soluţia sistemului în cel mult *n* iteraţii, *n* fiind dimensiunea matricei sistemului. O descriere amănunţită a acestei metode poate fi consultată în [4].

O altă metodă din clasa metodelor Krîlov este metoda "*Generalized Minimal Residuals"* (GMRES)[9]. Pseudo-codul pentru metoda GMRES este prezentat în figura 2.

```
Fie soluţia iniţială x(0), calculează vectorul
reziduu r = b - Ax(0)

ρ = ||r||2, v(1) = r/ ρ, β = ρ
for k = 1,2,... until convergence
     for j = 1,2, ... k,
          h(j,k) = (Av(k))'v(j)
     end
     v(k+1) = Av(k) - Σ_{j=1}^{k} h(j,k)v(j)
            (ortogonalizare Gram-Schmidt)
     h(k+1,k) = ||v(k+1)||2
     v(k+1,k) = v(k+1)/h(k+1,k)
end
```

Fig.2. Metoda GMRES.

Partea dificilă a acestei metode constă în a menţine ortogonalitatea vectorilor *v(j)*. Pentru aceasta, se utilizează o ortogonalizare Gram-Schmidt la fiecare iteraţie. Se observă ca la fiecare iteraţie metoda GMRES necesită memorarea şi calculul unui volum mai mare de informaţii. Pentru a evita această dificultate, metoda poate fi restartată după un număr de *m* iteraţii. Rezultatele obţinute până la iteraţia *m* vor constitui noua soluţie iniţială. Deoarece volumul de calcule şi de memorie utilizată creşte la fiecare iteraţie până la

restartare, parametrul *m* trebuie ales în funcție de problema particulară care este rezolvată, printr-un compromis între viteza de convergență pe de o parte respectiv complexitatea şi volumul de memorie necesar pe de altă parte. Dacă nu se utilizează restartarea, în aritmetică exactă, soluția este obținută în cel mult *n* iterații.

Tot din categoria metodelor Krîlov face parte metoda "*BiConjugate Gradient"* (BiCG)[4]. Metoda BiCG generează două secvențe ortogonale de vectorii reziduu *r* şi secvențe de vectori direcție A-ortogonali. Actualizările vectorilor reziduu şi a vectorului direcție sunt similare metodei gradientului conjugat dar sunt realizate utilizând matricea A şi transpusa ei. O versiune îmbunătățită a acestei metode, denumită "*BiConjugate Gradient Stabilized"*(BiCGSTAB)[9] este prezentată în figura 3.

```
Fie soluția inițială x(0), calculează vectorul
reziduu r = b – Ax(0)

ρ0 = 1, ρ1 = r(0)'r(0), α = 1, ώ = 1, p = 0, v = 0
for  k = 1,2, ...  until  convergence
     β = (ρk/ ρk-1)(α/ώ)
     p = r + β(p- ώv)
     v = Ap
     α = ρk/(r(0)'v)
     s = r – αv
     t = As
     ώ = (t's)(t't)
     x(k) = x(k-1) + αp + ώs
     r = s – ώt
```

Fig.3. Metoda BiCGSTAB

La fiecare iterație a metodei BiCGSTAB avem 6 operații *saxpy*, 4 produse scalare şi 2 înmulțiri matrice-vector. Necesarul de memorie constă în stocarea matricei A şi a 7 vectori de dimensiune *n*. Totuşi, numărul de operații în virgulă mobilă pe iterație nu poate fi folosit pentru a compara direct performanțele algoritmilor BiCGSTAB şi GMRES deoarece GMRES necesită mai puține iterații pentru a ajunge la soluție.

Este binecunoscut faptul că precondiționarea sistemului influențează drastic convergența metodelor Krîlov [9]. Precondiționarea transformă sistemul liniar inițial în alt sistem cu aceeaşi soluție dar pentru care convergența metodelor iterative este mult mai rapidă. O matrice M cu ajutorul căreia putem obține o astfel de transformare se numeşte matrice de *precondiționare* iar sistemul obținut în urma precondiționării este de forma:

$$M^{-1}Ax = M^{-1}b \qquad (2)$$

Pentru ca precondiționarea să fie eficientă trebuie ca M^{-1} să fie uşor de calculat. În experimentele noastre am utilizat o precondiționare de tip ILU.

Rezultate experimentale

Pentru a efectua un studiu comparativ al performanțelor metodelor iterative de tp Krîlov, am considerat rezolvarea unor sisteme neliniare cu un număr de variabile cuprins între 10000 şi 38000. Am experimentat utilizarea a două metode iterative de rezolvare a sistemelor liniare ce apar la fiecare iterație Newton: GMRES şi BiCGSTAB. Limita de eroare în obținerea soluției a fost fixată la 10^{-4} pentru toate metodele.

Am implementat algoritmii descrişi în această lucrare în limbajul C++ iar programele au fost rulate sub sistemul de operare Linux. Metode GMRES a utilizat o restartare după 40 de iterații. Din rezultatele obținute constatăm că ambele metode sunt performante, cu un plus pentru metoda BiCGSTAB care necesită mai puțină memorie şi realizează mai puține operații în virgulă mobilă.

În tabelul 1 este prezentat numărul de operații în virgulă mobilă efectuat la fiecare iterație precum şi necesarul de memorie.

Tabelul 1. Numărul de MFLOP/iteraţie şi necesarul de memorie

GMRES(40)			BiCGSTAB		
Dimensiune	MFLOP	Necesar Memorie (Mb)	Size	MFLOP	Necesar Memorie (Mb)
10000	2100	3.15	10000	723	1.44
14000	4800	4.90	14000	2880	1.92
18000	14100	6.25	18000	12800	3.07
22000	29500	8.10	20000	27500	3.51
26000	58000	9.21	26000	52000	4.60
30000	125000	11.10	30000	112000	5.66
34000	140000	12.4	34000	120000	7.22
38000	200000	14.0	38000	175000	8.20

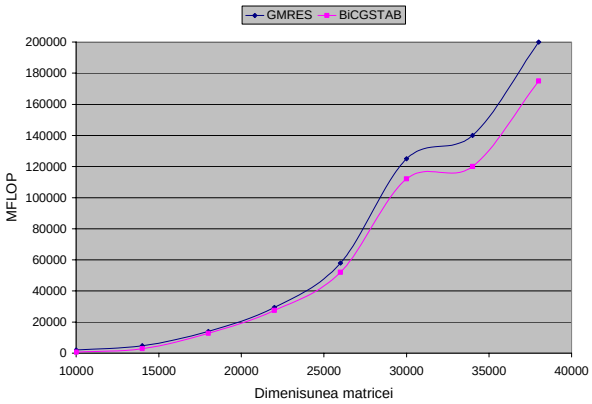

Fig.4. Numărul de operaţii în virgula mobilă (MFLOP) pe iteraţii pentru cele două metode.

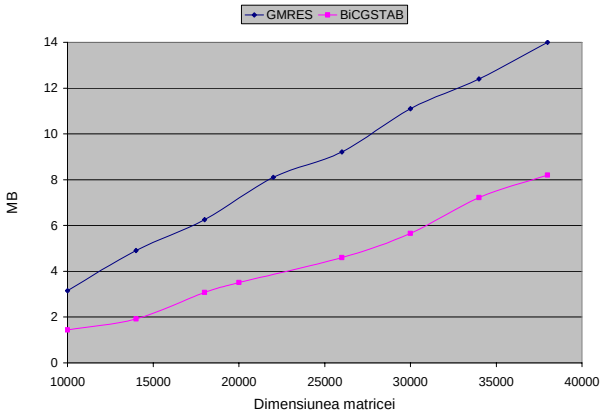

Fig.5. Necesarul de memorie pe iteraţie.

Concluzii

Algoritmul Newton de rezolvare a sistemelor neliniare combinat cu o metodă iterativă de tip Krîlov pentru rezolvarea sistemului liniar ce apare la fiecare iteraţie Newton reprezintă o alternativă de luat în seamă la metodele directe bazate pe factorizare LU. Atât complexitatea cât şi necesarul de memorie sunt mai mici în cazul metodelor iterative de tip Krîlov în comparaţie cu factorizarea LU. Singura problemă care apare în această situaţie este legată de rata de convergenţă a metodelor iterative care uneori poate fi destul de redusă.

Bibliografie

1.R. Barrett et. al., *Templates for the solution of linear systems: building blocks for iterative methods,* SIAM, 1994

2.P.N. Brown, Y. Saad, *Hybrid Krylov methods for nonlinear systems of equations,* SIAM Journal for Scientific and Statistical Computing, 11(3), 1990

3.P. Fisher, *Rational Expectations in Macroeconometric Models*, Kluwer Academic Publisher, Dordrecht, 1992.

4.G.H. Golub, C.F. van Loan, Matrix *Computations*, Johns Hopkins Series in Mathematical Sciences, The Johns Hopkins University Press, 1996.

5.R.E. Lucas, *Jr. Econometric Policy Evaluation : A Critique* , The Philips Curve and Labor Markets, vol 1, K. Brunner, A. H. Meltzer editors, North Holand, 1976.

6.B. Oancea, *Metode numerice de calcul paralel pentru modelele matematice din economie*, Teză de doctorat, ASE, Bucureşti, 2002

7.B. Oancea, *Implementation of Parallel Algorithms for Dense Matrix Computations,* Proceedings of the 7th International Conference on Informatics in Economy, Bucureşti, mai 2005.

8.B. Oancea, *Implementation of a Linear Algebra Library in Java,* Proceedings of "The Central and East European Conference in Business Information Systems", Universitatea Babes-Bolyai, Cluj-Napoca, mai, 2004.

9.Y. Saad, *Iterative methods for sparse linear systems,* PWS Publishing Company, 1996.