UTILIZAREA PACHETULUI DE PROGRAME jSNARK PENTRU RECONSTRUCȚIA ALGEBRICĂ A IMAGINILOR

Constantin Popa **Ioana Pomparău**

cpopa@univ-ovidius.ro ioanapomparau@gmail.com

Universitatea "Ovidius" din Constanța

Rezumat: Cercetările privind reconstrucția de imagini în tomografia computerizată începute în anii '50 au continuat cu lucrările lui G.T. Herman şi R. Gordon din anii '70, în care sunt propuse atât tehnici de generare a unor imagini model, cât şi algoritmi eficienți de reconstrucție. În acest context a fost elaborată prima versiune a pachetului de programe SNARK, cu dezvoltările ulterioare, SNARK09 şi jSNARK. În prima parte a lucrării se descrie modul de utilizare a pachetului jSNARK pentru generarea de imagini model (fantome). Imaginea şi datele de proiecție astfel obținute sunt exportate şi utilizate în mediul de programare MATLAB pentru efectuarea unor experimente de reconstrucție cu algoritmi iterativi de tip Kaczmarz.

Cuvinte cheie: reconstrucția algebrică a imaginilor; pachetul de programe jSNARK; algoritmul Kaczmarz.

Abstract: Research on the computerized tomography image reconstruction, started in the 50s, continued with the work of G.T. Herman and R. Gordon in the 70s. They proposed techniques for generating model images and also efficient algorithms for reconstruction. In this respect, a first version of the SNARK software package was developed. The current versions are SNARK09 and jSNARK. In the first part of this paper we describe how to use the jSNARK package to generate model images (ghosts). Image and projection data thus obtained are exported and used in the MATLAB programming environment to perform reconstruction experiments with Kaczmarz type iterative algorithms.

Key words: algebraic image reconstruction; jSNARK software package; Kaczmarz algorithm.

1. Generalități referitoare la reconstrucția de imagini din proiecții şi aplicația jSNARK

Reconstrucția de imagini din proiecții reprezintă o problemă actuală, iar cercetătorii în domeniu au propus numeroşi algoritmi pentru a o rezolva (vezi [2, 4]). Pentru a evalua şi compara performanța diverselor metode se recomandă efectuarea de experimente numerice exhaustive. Aplicația jSNARK a fost dezvoltată pentru a oferi posibilitatea generării de fantome, imagini reprezentând secțiuni transversale anatomice, precum şi date de proiecție, simulate într-un mod realist. De asemenea, implementează algoritmi clasici şi permite scrierea de programe definite de utilizator.

Prima variantă de SNARK a fost dezvoltată de Richard Gordon în 1970. Următoarele două versiuni, SNARK77 şi SNARK89 au fost scrise în limbajul de programare FORTRAN, iar la SNARK93 s-a folosit FORTRAN77. Începând cu SNARK05 s-a trecut la implementare în C++. Versiunile actuale, pe care le vom denumi generic SNARK, sunt SNARK09 (vezi documentația [1]), în C++ şi jSNARK (vezi documentația [5]), scrisă în limbajul JAVA.

Prin reconstrucție de imagini din proiecții înțelegem obținerea imaginii unei distribuții bidimensionale folosind aproximări ale integralelor sale linie de-a lungul unui număr finit de drepte ale căror locații sunt cunoscute (pentru mai multe detalii vezi [2]).

În SNARK [2], o imagine este determinată de o regiune pătrată, cu centrul în originea sistemului cartezian şi de o funcție de două variabile, care are valoarea zero în afara acestei regiuni. Vom numi densitate valoarea funcției într-un punct *(x, y)*. În tomografia computerizată, densitatea unei imagini într-un punct *(x, y)* reprezintă coeficientul de atenuare pentru fascicolul

de raze X ce trece prin țesuturile de la punctul (x, y). Aceşti coeficienți ai diferitelor țesuturi sunt în general cunoscuți (vezi [2, Tabelul 4.1]). Se presupune că aerul are densitate nulă.

Se realizează o discretizare a regiunii imaginii folosind un grid cu p elemente şi se obțin $n = p \times p$ pătrate, denumite pixeli. Se consideră o scanare cu m raze de la surse la receptori. Obținem o matrice de scanare de dimensiune $m \times n$ şi un vector de măsurători b din R^m. În Figura 1 este ilustrat modul în care se calculează valorile componentelor matricei A. Rezolvând sistemul de ecuații liniare $Ax = b$, se calculează o aproximare a imaginii scanate.

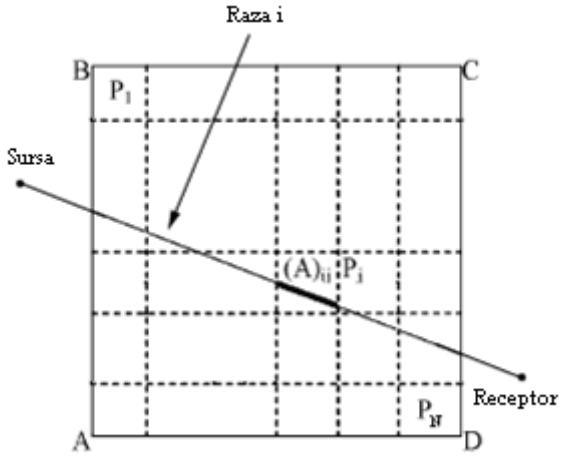

Figura 1.

Pentru execuția unui program SNARK se pot defini următoarele date de intrare [5, 1]:

- Date geometrice:
  - dimensiunea discretizării;
  - geometria razelor, i.e. paralele sau divergente;
  - numărul de direcții de proiecție;
  - lista direcțiilor de proiecție.
- Imagine de test (opțional):
  - imaginea fie este rezultată dintr-o rulare anterioară, fie este specificată prin descrierea de figuri geometrice.
- Date de proiecție:
  - acestea sunt calculate experimental sau sunt generate de aplicație.

Se pot colecta următoarele categorii de date de ieşire:

- Fantomele generate precum şi eventuale reconstrucții rezultate în urma aplicării unor algoritmi (proprii SNARK sau definiți de utilizator)
- Evaluare a diferenței dintre reconstrucțiile obținute
- Timpul de execuție al maşinii de calcul pentru fiecare instrucțiune şi iterație a unui algoritm.

Putem avea până la trei etape în execuția unui program SNARK:

1. Faza de generare a datelor;
2. Faza de inițializare şi reconstrucție;
3. Faza de analiză.

Fiecare din etapele de mai sus foloseşte date de intrare şi produce date de ieşire. Unele dintre datele de ieşire corespunzătoare unei faze de execuție vor reprezenta date de intrare ale unei faze următoare.

În acest articol ne vom concentra pe studiul primei etape de execuție. Ne interesează să generăm şi să exportăm fantome şi date proiecție, pentru a le utiliza ulterior în alte medii de programare. În Secțiunea 2 vom descrie şi exemplifica modalitatea de creare a datelor de test, iar în Secțiunea 3 le vom folosi pentru a construi în MATLAB o aproximare a imaginii inițiale, folosind algoritmul Kaczmarz (vezi, e.g, [3]).

2. Crearea de fantome şi de date de proiecție utilizând aplicația jSNARK

În pachetul de programe jSNARK, imaginile de test, pe care le vom numi fantome, sunt o combinație de forme geometrice (obiecte) de tipul elipsă, dreptunghi, triunghi isoscel, sector de cerc sau segment de cerc (aria cuprinsă între o coardă şi arcul pe care îl întinde). Fiecare obiect va avea o altă rată de absorbție a razelor incidente, simulând diferite țesuturi. Utilizatorul va specifica şi poziția, orientarea şi dimensiunea formelor geometrice care compun o fantomă. Densitatea la un punct dat va fi calculată ca suma densităților obiectelor care trec prin acel punct. Se pot defini până la 7 nivele de energie corespunzătoare unui spectru de energie monocromatic sau policromatic. În Figura 2 este ilustrată o captură de ecran a interfeței grafice a aplicației jSNARK.

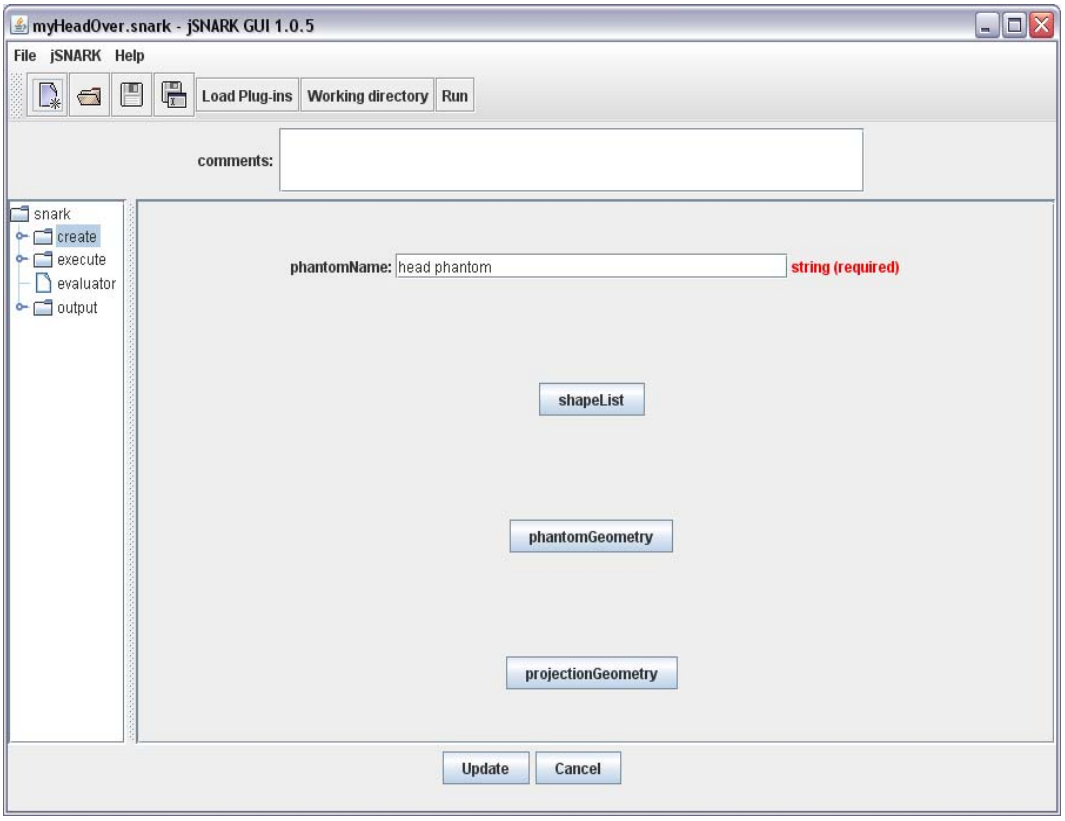

Figura 2.

Construim un studiu de caz, o fantomă reprezentând secțiunea transversală a unui cap de om în care apare un meningiom. Vezi Figurile 3–4. Definim şi datele de proiecție în cadrul secțiunii „projection Geometry”.

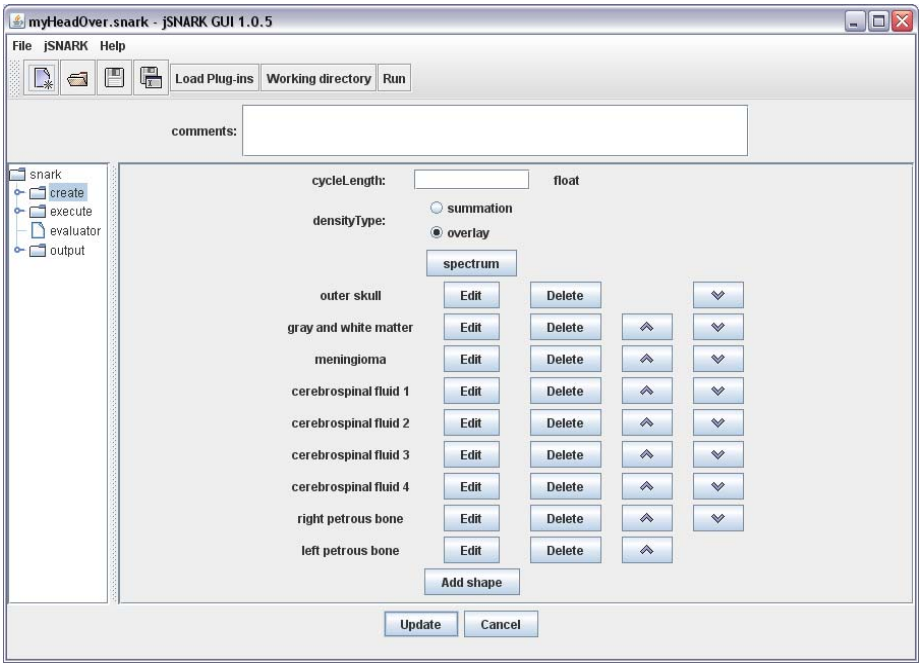

Figura 3.

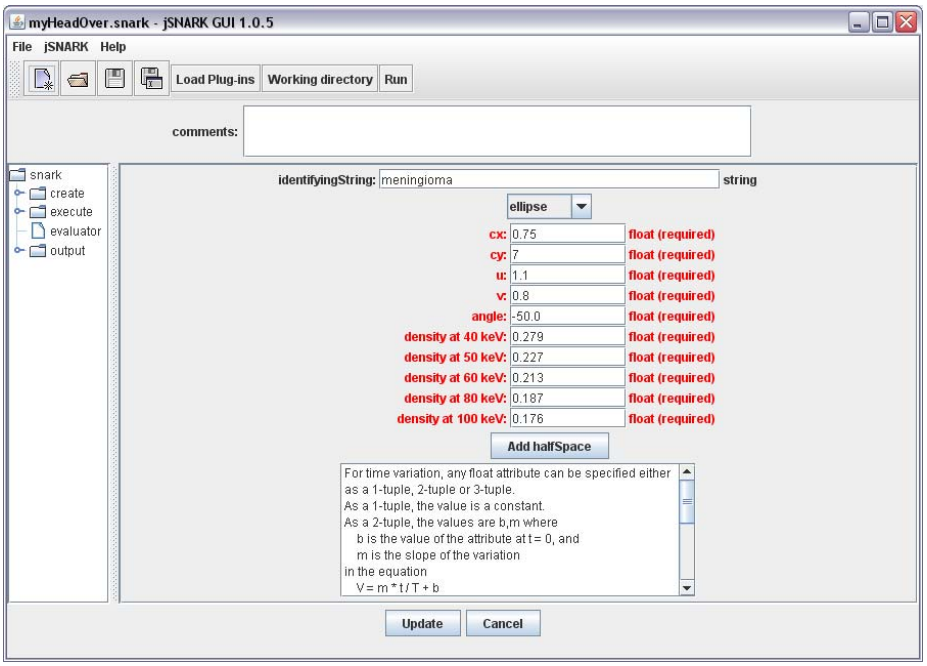

Figura 4.

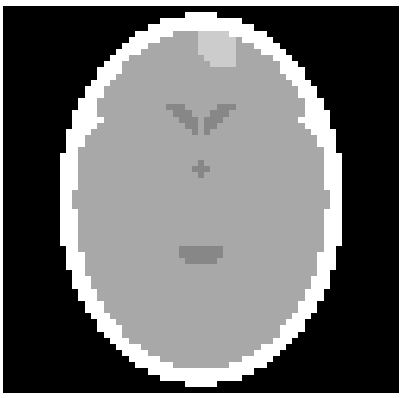

Figura 5.

În urma unei execuții jSNARK obținem imaginea PNG din Figura 5. Această imagine are dimensiunea de *63 × 63* pixeli. Fiecare pixel are o valoare întreagă cuprinsă între 0 şi 255.

3. Algoritmul Kaczmarz pentru reconstrucția de imagini din proiecții

În această secțiune vom prezenta pe scurt algoritmul propus de S. Kaczmarz în 1937 (vezi pentru detalii [4]). Acest algoritm iterativ, utilizat în reconstrucția algebrică a imaginilor, foloseşte proiecții succesive pe hiperplanele determinate de ecuațiile unui sistem liniar de forma $Ax = b$, unde A este o matrice de dimensiuni $m \times n$ şi b este un vector din R^m. Notând cu A_i linia i din matricea A (presupusă nenulă) şi cu b_i componenta i din termenul liber b, algoritmul Kaczmarz se scrie astfel.

Inițializare: Fie x^0 din R^n.

Pas iterativ: Pentru orice $k \geq 0$, se calculează

$$x^{k+1} = P_{H_1} \circ P_{H_2} \circ ... \circ P_{H_m}(x^k), \tag{1}$$

unde $H_i = \left\{x \in R^n, \langle x, A_i \rangle = b_i\right\}$ şi $P_{H_i}(x) = x - \dfrac{\langle x, A_i \rangle - b_i}{\|A_i\|^2} A_i$, pentru orice $i \in \{1, 2, ..., n\}$.

Dacă folosim $x^0 = 0$, şirul generat de algoritmul Kaczmarz converge la soluția de normă minimă, x_{LS}, a sistemului $Ax = b$. Aceasta este în general diferită de soluția exactă (cea pe care vrem să o reconstruim), diferența fiind determinată de componenta din spațiul nul al matricei A a imaginii exacte (vezi pentru detalii [4]).

O îmbunătățire a reconstrucției se obține introducând în algoritmul Kaczmarz o etapă de corecție bazată pe constrângerea componentelor aproximațiilor, de forma unei funcții $C : R^n \to [a, b] := [a_1, b_1] \times [a_2, b_2] \times ... \times [a_n, b_n] \subset R^n$ definită prin:

$$(Cx)_i = \begin{cases} x_i, & pt.\ x_i \in [a_i, b_i] \\ a_i, & pt.\ x_i < a_i \\ b_i, & pt.\ x_i > b_i \end{cases}, (\forall)\ i \in \{1, 2, ..., n\}. \tag{2}$$

Prezentăm mai jos algoritmul Kacmzarz care utilizează o constrângere de forma dată de ecuația (2).

Algoritmul Kaczmarz cu constrângeri

Inițializare: Fie x^0 din R^n.

Pas iterativ: Pentru orice $k \geq 0$, se calculează

$$x^{k+1} = C\left(P_{H_1} \circ P_{H_2} \circ ... \circ P_{H_m}(x^k)\right). \tag{3}$$

4. Rezultate numerice

În această secțiune vom prezenta experimente de reconstrucție relativ la imaginea generată în Secțiunea 2. Datele obținute prin utilizarea pachetului jSNARK: imaginea exactă (vezi Figura 5), matricea de scanare A şi termenul liber b, sunt exportate pentru a putea fi prelucrate în

mediul de programare MATLAB. Acesta se foloseşte ulterior la rezolvarea sistemului $Ax = b$ pentru reconstrucţia imaginii exacte. Împărţim valoarea fiecărui pixel al acesteia la 255 şi obţinem imaginea din Figura 6**,** ilustrată în MATLAB, cu proprietatea că fiecare componentă aparţine intervalului [0, 1].

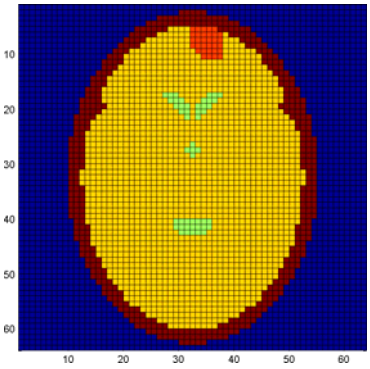

Figura 6. Imagine exactă

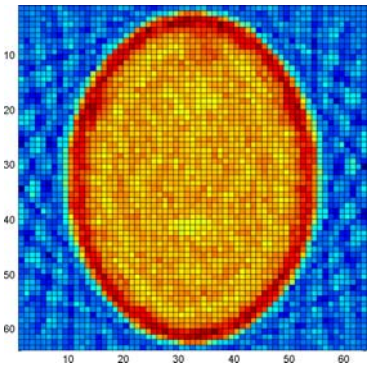

Figura 7. Reconstrucţie – Kaczmarz clasic

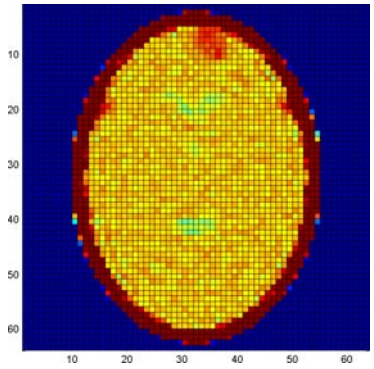

Figura 8. Reconstrucţie – Kaczmarz cu constrângeri

Pentru o aproximaţie iniţială $x^0 = 0$, aplicăm algoritmul Kaczmarz, respectiv algoritmul Kaczmarz cu o constrângere definită prin (2) cu $[a,b]=[0,1]^n$. Pentru 500 de iteraţii rezultă cele două reconstrucţii din Figura 7, respectiv Figura 8. Se observă că în urma utilizării unei constrângeri se obţine o aproximaţie mai bună a imaginii exacte.

BIBLIOGRAFIE

1. **DAVIDI, R.; HERMAN, G. T.; KLUKOWSKA, J.:** SNARK09: a Programming System for the Reconstruction of 2D Images from 1D projections, 2012.

2. **HERMAN, G. T.:** The Fundamentals of Computerized Tomography: Image Reconstruction from Projections, Second Edition, Springer – Verlag, London, UK, 2009.

3. **KACZMARZ, S.:** Angenäherte auflösung von systemen linearer gleichungen. În: Bulletin International de l'Academie Polonaise des Sciences et des Lettres, vol. 53, 1937, pp. 355 – 357.

4. **POPA, C.:** Projection Algorithms – Classical Results and Developments. Applications to Image Reconstruction. Lambert Academic Publishing – AV Akademikerverlag Gmbh & Co. KG, Saarbrücken, Germany, 2012.

5. **ROWLAND, S. W.:** jSNARK v1.0.4: a Programming System for the Reconstruction of Images from Projections.