The 7th Balkan Conference on Operational Research

"BACOR 05"

Constanta, May 2005, Romania

IMPROVING IMAGE RECONSTRUCTION WITH EVOLUTIONARY ALGORITHMS*

ANDREI BĂUTU
Naval Academy "Mircea cel Batran", Constanta, Romania

ELENA BĂUTU
CONSTANTIN POPA
Faculty of Mathematics and Informatics, "Ovidius" University, Constanța, Romania

HENRI LUCHIAN
Faculty of Computer Science, "Al. I. Cuza" University, Iași, Romania

---

Abstract
We present two hybrid algorithms for image reconstruction based on Particle Swarm Optimization (PSO) and Kaczmarz's algorithm. We compared these hybrids with Kaczmarz's original algorithm, pure PSO and Genetic Algorithms. We found that both our hybrid algorithms offer good solutions and one of them offers a nearly perfect solution. The article is structured in six sections. The first section contains a short description of the problem we tackle. Sections two, three and four describe the algorithms we compared. Section five contains results of our comparison, and section six contains ideas about future work.

Keywords*: image reconstruction, genetic algorithm, particle swarm optimization*

---

* This research was supported by the PNCDI INFOSOC Grant 131/2004

1. IMAGE RECONSTRUCTION

The problem we tackle comes from the field of Computerized Tomography (CT), and has large applicability in various fields like medicine, transportation, geology, or remote vision. Consider a planar cross-section of a body that is being scanned with some type of ray (or beam). For exemple, X-rays or infrared beams are used in medicine imaging, while electromagnetic beams are used in electromagnetic geotomography. The body's attenuation of rays in every point of the cross-section has to be reconstructed. We'll call image (or picture) this unknown function of two variables with real nonnegative values.

The fundamental model in the finite series-expansion approach may be formulated as follows: a cartesian grid of square picture elements, called pixels, is introduced into the region of interest so that it covers the whole picture that has to be reconstructed. The pixels are numbered from left to right and top to bottom, from *1* to *n* (see 1.1). We assume that the ray attenuation takes a constant uniform value x_j throughout the j^{th} pixel, for *j=1,2,…,n*. We also consider that each ray is a straight line between a source and a detector. Ray sources are located on the left hand side of the cross-section, and ray detectors are on the right hand side. Each source-detector pair denoteds a ray. Let m be the number of rays.

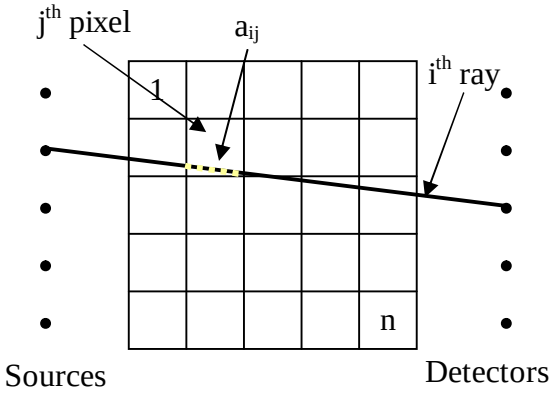

Figure 1.1 CT Discretization

For any *i=1,2,…,m* and *j=1,2,…,n*, we denote by a_{ij} the length of the segment defined by the intersection of the i^{th} ray with the j^{th} pixel. We assume that a_{ij} represents the weight of the contribution of the j^{th} pixel to the total attenuation along the i^{th} ray. The physical measurement of the total attenuation along the i^{th} ray, denoted by b_i, is the line integral of the unknown attenuation function along the the ray. Therefore, for in this discretized model, each line integral is a finite sum and we get the system of linear equations that describes the model

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \,,\ i = 1,2,...,m \,, \tag{1.1}$$

which we can rewrite in matrix form as

$$Ax = b \tag{1.2}$$

where $A=(a_{ij})$ is an $m\times n$ matrix, $x=(x_1,...,x_n)^T$, and $b=(b_1,...,b_m)^T$. Here, *A* is called projection matrix, *x* is the image vector, and *b* is the measurements vector. Frequently, due to measurement errors, the measurements are subject to perturbations, so *b* does not belong to $R(A)$[1]. We can reconsider the problem (1.2) as a liniar least squares problem: find $x \in \mathrm{R}^n$ such as

$$\|Ax-b\|^2 = \min! \tag{1.3}$$

So, one needs to find $x^* \in R^n$ such that:

$$\|Ax^*-b\| = \min\{\|Ax-b\|, x\in R^n\} \tag{1.4}$$

We denote by $LSS(A,b)$ the set of all liniar least squares solutions of the equation (1.4). It is well known (see [1]) that $LSS(A,b)$ contains a unique element of minimal norm denoted by x_{LS}.

2. KACZMARZ'S ALGORITHM

We will briefly describe Kaczmarz's algorithm (KA) introduced in [3]. Let $a_i \neq 0$ be the i^{th} row of the matrix A, and b_i the i^{th} component of the vector b, for $i=1,2,...,m$. Let $f_i(b,\cdot)$, $F(b,\cdot): R^n \to R^n$ be the applications defined by

$$f_i(b,x) = x - \frac{\langle x, a_i\rangle - b_i}{\|a_i\|^2} a_i \tag{2.1}$$

$$F(b,x) = (f_1 \circ \ldots \circ f_m)(b,x) \tag{2.2}$$

KA may be stated as follows: let $x^0 \in R^n$ (arbitrary) and $x^k \in R^n$ an approximation. The next approximation is computed by

$$x^{k+1} = F(b, x^k),\ k \geq 0 \tag{2.3}$$

Kaczmarz proved in [3] that $\lim_{k\to\infty} x^k = x_{LS}$ for square nonsingular systems as in (1.2), and Tanabe proved in [7] that KA can be used for arbitrary least-squares problems of the form (1.3). However, x_{LS} is obtained as a limit point with Tanabe's method if and only if $x^0 \in R(A^T)$.

Extensions of KA and other algorithms have been proposed in [2], [4], [5], and [6], but all these algorithms have the same weak point: they obtain an approximation of the solution using another approximation. Thus, accumulation of approximations (i.e. errors) appear. Moreover, due to the lack of information, the usual initial approximation is a blank image ($x^0 = 0$).

---

[1] R(A) is the range of matrix A

3. EVOLUTIONARY ALGORITHMS

In general, an evolutionary algorithm is a stochastic, directed search algorithm based on principles of evolution theory applied to computer systems. The algorithm maintains a set of possible solutions and evolves them in a controlled randomly fashion in order to obtain a solution for a problem. Due to their evolutionary nature, there is no need for specific information, or superior theoretical knowledge about the optimization problem at hand.

The evolutionary process of the solutions is based on evolutionary operators. In order for evolution to occur, solutions must pass the sieve of selection. Each solution is assigned a measure of its performance in solving the problem objective, called fitness; the process is called evaluation. Then, solutions are selected according to their fitness. The selection process simulates the "survival of the fittest" paradigm from nature. The better fitted the solution, the higher the probability of leaving more offspring in the next generation of solutions.

We used evolutionary algorithms to improve results of KA image reconstruction problems. The fitness function, common to all the algorithms, is based on the problem objective function, thus trying to minimize the absolute errors between the computed and the measured results. We defined the fitness function as

$$fitness(x) = \left(1 + \|Ax - b\|^2\right)^{-1} \tag{2.4}$$

where x is a vector of $m \times n$ values from $[0,1]$ that represents the current image.

3.1 GENETIC ALGORITHM

The first evolutionary algorithm we used is a genetic algorithm (GA). In this context, a solution is called *chromosome*, and encodes an image as a vector of n double precision floating point numbers between 0 and 1, which are called genes. Actually, the value of the i^{th} gene represents the absorption of the i^{th} pixel in the image, and decodes into the colour of that pixel.

On each iteration of the algorithm there is a fixed size population of *pop_size* chromosomes. The population evolves by means of three genetic operators: mutation, crossover and selection. For a chromosome, the mutation operator modifies each gene with *mut_rate* probability by changing the value of a randomly selected gene. The crossover operator works on a pair of chromosomes by swapping substrings of genes. Each chromosome is selected for crossover with *cross_rate* probability, and a random crossover point is selected. Genes preceding that point are swapped between chromosomes, and the resulting chromosomes are called offspring. The selection operator is implemented with Monte Carlo selection scheme. The probability that a chromosome will survive in the next generation is proportionate to its fitness. The better fitted the chromosomes, the higher the probability of leaving more offspring in the next generation.

3.2 PARTICLE SWARM OPTIMIZATION

The second evolutionary algorithm we used is an n-dimensional extension of the PSO algorithm described in [8]. In this context, a solution is called *particle*, and encodes an image as a vector of tuples of double precision floating-point numbers between 0 and 1. The values in the i^{th} tuple represent the current position in the search space, the velocity and the best so far position of the particle in its i^{th} dimension. The position component in the i^{th} tuple represents the absorption value of the i^{th} pixel in the image, and decodes into the colour of that pixel. Each particle has a fixed set of neighbours which influence its search strategy (we considered only one neighbour in our implementation).

On each iteration of the algorithm we have a fixed-size set of *part_count* particles. Each particle "moves" through the search space based on its history (best so far position) and its neighbours. On each iteration, velocity and position components of particles are updated using the following formulas

$$v_{t+1}^i{}' = v_t^i \cdot inertia + rand() \cdot cognitive \cdot (b_t^i - p_t^i) + rand() \cdot social \cdot (n_t^i - p_t^i)$$
$$v_{t+1}^i = \min(v_{\max}, \max(-v_{\max}, v_{t+1}^i{}')) \quad (2.5)$$

$$p_{t+1}^i{}' = p_t^i + v_{t+1}^i$$
$$p_{t+1}^i = \min(1, \max(0, p_{t+1}^i{}')) \quad (2.6)$$

where, for iteration t and particle i, v_t^i denotes the speed, b_t^i denotes the best so far position, p_t^i is the current position and n_t^i – the neighbour's position. The parameters of the algorithm (*inertia*, *cognitive*, *social*, and v_{max}) control the bias between exploration and the exploitation in the search space. The *best so far* position is updated if a better position is located.

4. HYBRID ALGORITHMS

In this section, we describe the hybrid algorithms based on Kaczmarz's algorithm and particle swarm optimisation. The classical genetic algorithm performed quite badly compared to the PSO approach and Kaczmarz algorithm; hybridisation of GA with KA will be subject of further work.

First hybrid algorithm (FHA) has two distinct stages. During the first stage, we run the PSO algorithm described earlier. In the second stage, the algorithm uses the best solution from the first stage as the initial approximation of the solution in Kaczmarz's algorithm. The solution of the algorithm is the solution reached by KA in the second stage.
Second hybrid algorithm (SHA) runs KA and PSO algorithms in parallel. For each particle SHA alternates between PSO and KA iterations, doing one iteration of each of them. The solution of the algorithm is chosen from the set of best solutions reached by PSO.

EXPERIMENTS

We present our results for four image reconstruction experiments. The first three images are artificial test images. The last image is a scanned photo of a baby, which was preprocessed through a resample, a grayscale and a negative filter. Images and their properties (left to right) are presented below:

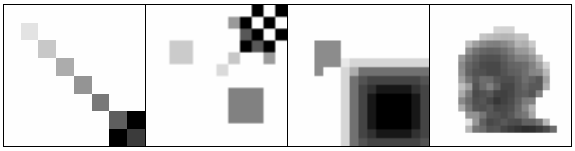

Figure 5.1 Test images (original source)

Table 5.1 Image colormap

Image	Size	Source	Unique colors
Test 1	8x8	Drawing	9
Test 2	12x12	Drawing	8
Test 3	16x16	Drawing	11
Test 4	20x20	Scanned photo	71

Table 5.2 Test images properties

For each image, we ran all five algorithms with various settings, but we present results only on 10^{th} and 100^{th} iteration for the following settings:

- KA: no settings required;
- GA: pop_size=60, mut_rate=5%, and cross_rate=70%;
- PSO: part_count=60, v_{max}=1.1, inertia=0.35, cognitive=1.2, and social=1.2;
- FHA: part_count=60, v_{max}=1.1, inertia=0.35, cognitive=1.2, and social=1.2;
- SHA: part_count=60, v_{max}=1.1, inertia=0.35, cognitive=1.2, and social=1.2.

1.1 RESULTS AFTER 10 ITERATIONS

After 10 iterations, both classic GA and PSO failed to obtain a "good" reconstruction even for the simple Test 1 image. While the GA's image is useless, the PSO's image began to shape up. This is however a normal behaviour since 10 iterations are not enough to allow these evolutionary algorithms to establish a good search direction.

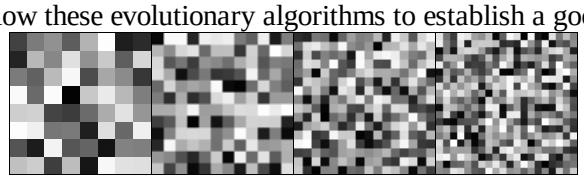

Figure 5.2 GA results after 10 iterations

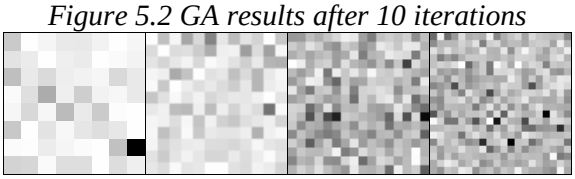

Figure 5.3 PSO results after 10 iterations

As it was expected, KA found some shapes, but images are still affected by noise, which is common in Kaczmarz image reconstructions. Also, in all the tests we noticed that KA has trouble reconstructing shapes near the edges of the images. We consider this to be normal behaviour, since these are the areas that are the least scanned by rays.

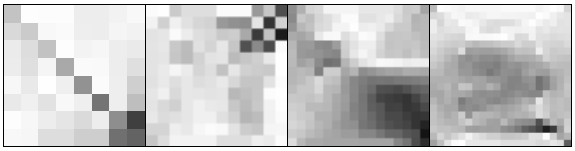

Figure 5.4 KA results after 10 iterations

Powered by KA, FHA found some shapes, too. However, the shapes were actually found in the KA stage. Moreover, because of the "bad" starting approximation generated with PSO during the first stage of FHA, reconstructed images are not as good as those of KA.

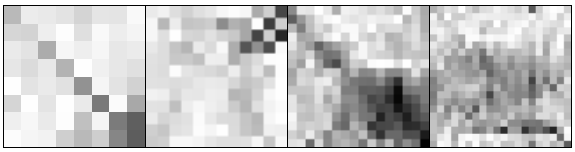

Figure 5.5 FHA results after 10 iterations

Results of SHA are surprisingly good. Test 1 image is almost perfect and other images have very well defined shapes. It seems that KA drives the reconstruction process towards the "good" image, while PSO helps eliminating the noise, thus filtering the image.

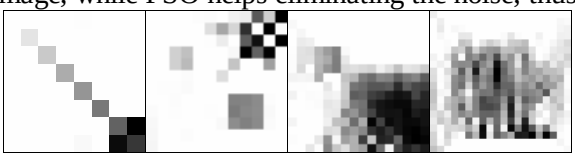

Figure 5.6 SHA results after 10 iterations

RESULTS AFTER 100 ITERATIONS

After 100 iterations, GA still has no good image. It is clear that this simple approach (using the chromosome encoding described in section 3.1) does not work for these general images. However, maybe a different encoding or modified, problem specific, operators may lead to better results. We will try to further investigate this aspect.

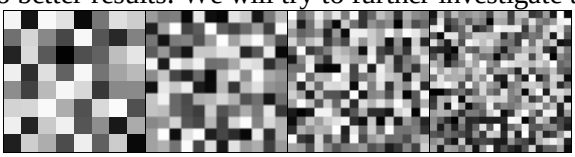

Figure 5.7 GA results after 100 iterations

PSO has improved its Test 1 and Test 2 results, but results are not very satisfactory and it still has no acceptable solution for Test 3 and Test 4 images.

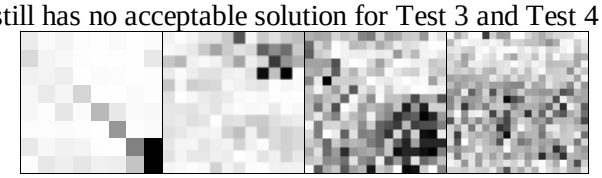

Figure 5.8 PSO results after 100 iterations

As we expected, KA has improved very little its results for Test 1 and Test 2, too, but the noise accumulated during computations is making further improvements difficult. For Test 3 and Test 4 KA has no satisfactory results, probably for the same reasons.

Basicly, KA reached a dead-end where the solution image improves very little or none at all.

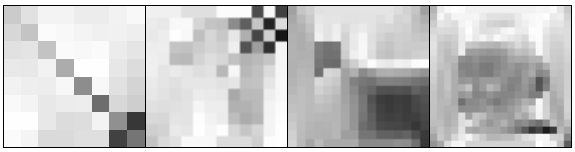

Figure 5.9 KA results after 100 iterations

FHA performed much better than PSO, and a little better than simple KA. All reconstructed images are affected by the noise from the KA in the second stage of the algorithm. Moreover, final results are almost independent of the settings of PSO or its solutions (which become KA's starting approximation).

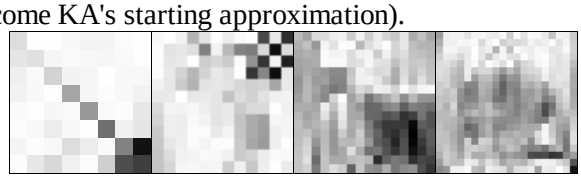

Figure 5.10 FHA results after 100 iterations

After 100 iterations, SHA has almost perfect solutions for Test 1 and Test 2. These images have almost invisible differences when compared to original images. Also, SHA found very good solutions for Test 3 and Test 4. All its solutions are much better than those offered by KA alone, or the other evolutionary algorithms we presented in this article.

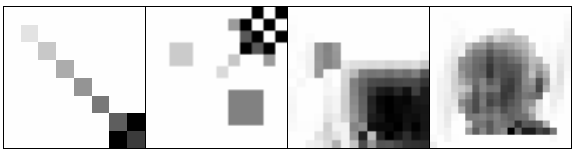

Figure 5.11 SHA results after 100 iterations

OVERAL PERFORMANCE

The following charts show the evolution during 100 iterations of the residual norm for KA, and the mean residual norm for PSO, FHA and SHA over 25 runs. We did not include on these charts the GA's mean residual norm as they were of no interest (the mean residual norm in the GA over 25 runs of the algorithm was almost constant).

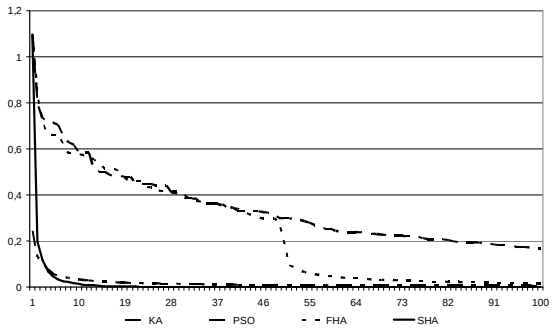

Figure 5.12 Test 1 - Residual norm over 100 iterations

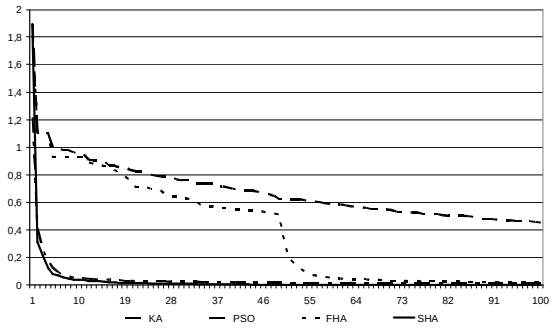

Figure 5.13 Test 2 - Residual norm over 100 iterations

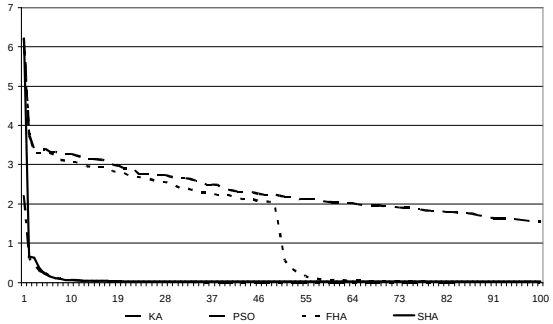

Figure 5.14 Test 3 - Residual norm over 100 iterations

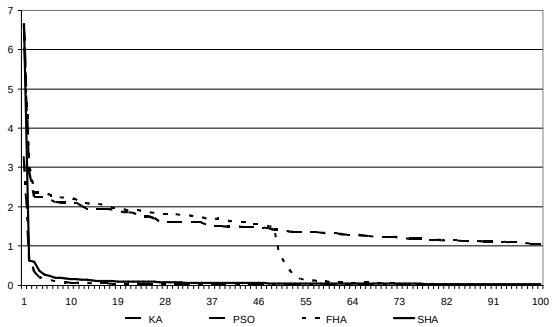

Figure 5.15 Test 4 - Residual norm over 100 iterations

CONCLUSIONS AND FUTURE WORK

Particle Swarm Optimisation is a powerful technique which can be used successfully to improve results of classical algorithms for image reconstruction. SHA outperforms particle swarm optimisation and Kaczmarz algorithm by far, providing results that are very close to the original images.

One future direction is the study of the influence that PSO parameters have on the quality of the solution of SHA. More tests need to be done on real-world data and we will try to develop other fitness functions that are more suitable for our problem, too. We will also test other hybridisations of numerical and evolutionary algorithms for image reconstruction.

BIBLIOGRAPHY

[1] Bjork A., "Numerical methods for least squares problems", SIAM Philadelphia, 1996;
[2] Bjork A., Elfving T., "Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations", BIT 19, 1979, pp. 145-163;
[3] Kaczmarz S., "Angenaherte Auflo-sung von Systemen linearer Gleichungen", Bull. Acad. Polonaise Sci. et Letters A, 1937, pp. 355-357;
[4] Popa C., "Iterative methods for lin-ear least-squares problems", Mathematical Monographs, 77, Western University of Timisoara, 2003;
[5] Popa C., "Least-squares solution of overdetermined inconsistent linear systems using Kaczmarz's relaxation", Intern. J. Comp. Math., 55, 1995, pp. 79-89;
[6] Pyle L. D., "A generalised inverse ε-algorithm for constructing intersection projection matrices with application" , Numer. Math. 10, 1967, pp. 86-102;
[7] Tanabe K.,"Projection method for solving a singular system of linear equations and its applications", Numer. Math., 17, 1971, pp. 203-214;
[8] Kennedy J., Eberhart R., "Particle Swarm Optimisation", Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942-1948.